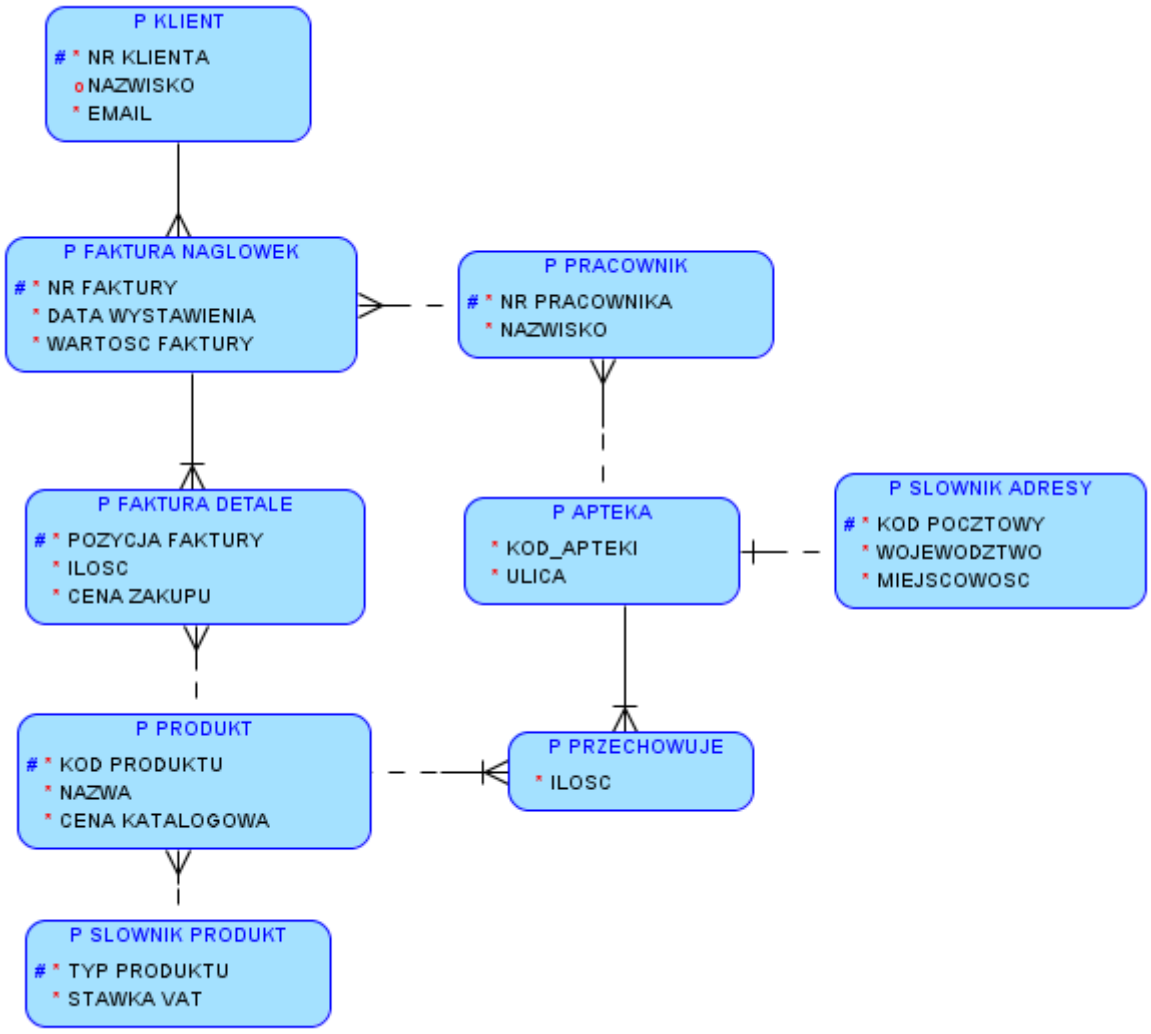
1. Analiza biznesowa projektowanej rzeczywistości:

Celem projektu jest implementacja bazy danych sprzedaży w pewnej sieci aptek. Zależy nam głównie na monitorowaniu transferu produktów, które dzielą się na 3 typy (leki, suplementy, kosmetyki [dane zawarte w słowniku]).

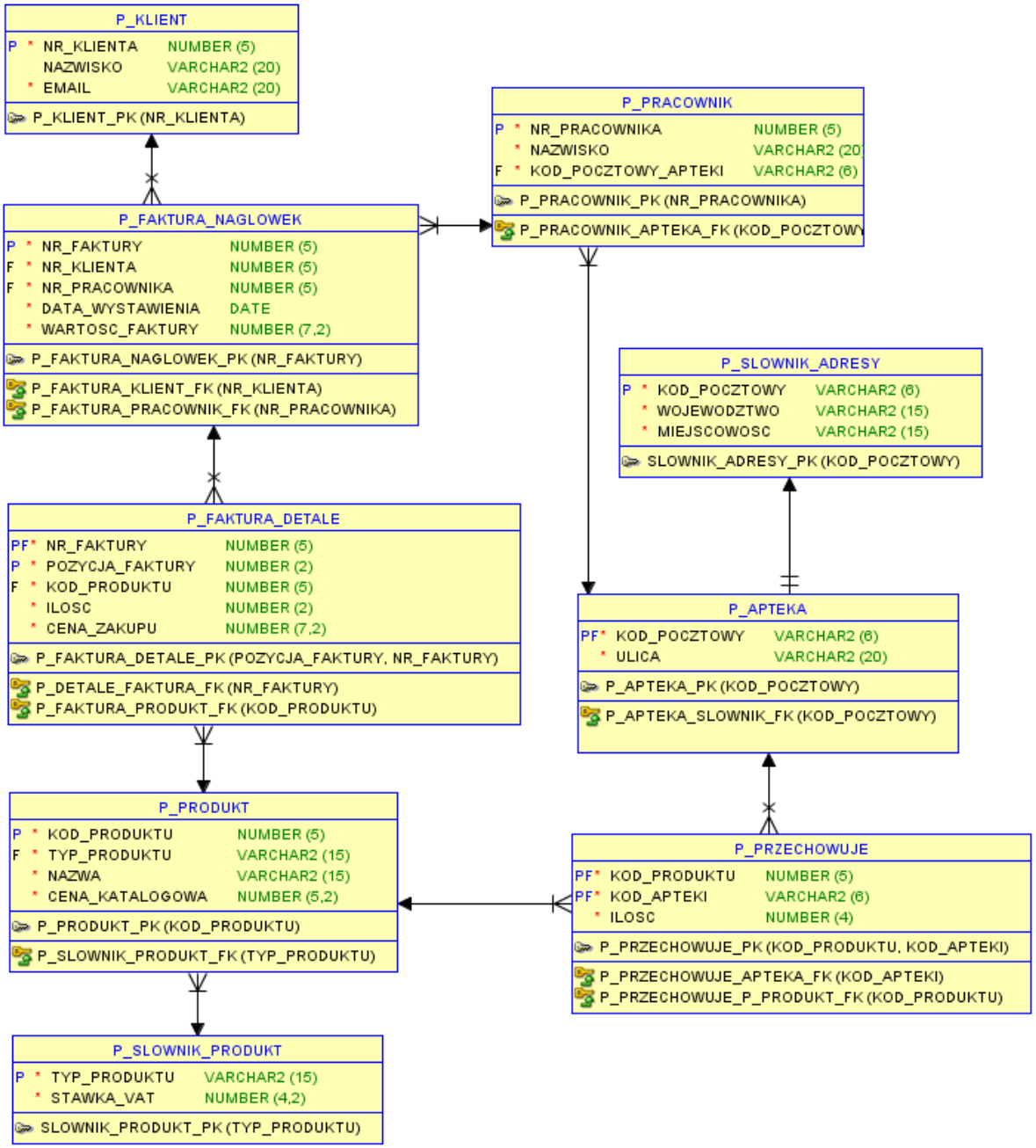
Nasza baza danych zawiera minimum informacji o pracownikach wystawiających faktury jak i o klientach.

Istotne dla nas są jedynie zyski generowane przez konkretne produkty, pracowników i apteki.

2. Model logiczny:



3. Model relacyjny:



4. Skrypt instalujący projekt.

Skrypt składa się dwóch części:
W pierwszej tworzone są wszystkie tabele, triggery, funkcje, procedury
W drugiej następuje wygenerowanie danych do tabel oraz 5 perspektyw pokazujących:

1. Dochód uzyskany przez wszystkie apteki w każdym miesiącu.
2. Dochód wygenerowany przez każdego z pracowników.
3. Dochód wygenerowany przez każdą z aptek.
4. Dochód pozyskany ze sprzedaży każdego z produktów.
5. Łączną wartość zakupów każdego z klientów.

```
-----------------------------------------------------------------
--    SKRYPT TWORZACY BAZE DANYCH ORAZ
--    UZUPELNIAJACY JA LOSOWYMI DANYMI I
--    NA ICH PODSTAWIE TWORZACY PERSPEKTYWY
-----------------------------------------------------------------
--
--    TWORZENIE TABEL ORAZ POWIAZAN:
--
-----------------------------------------------------------------

CREATE TABLE p_apteka (
  kod_pocztowy   VARCHAR2(6) NOT NULL,
  ulica         VARCHAR2(20) NOT NULL
);

ALTER TABLE p_apteka ADD CONSTRAINT p_apteka_pk PRIMARY KEY ( kod_pocztowy );

CREATE TABLE p_faktura_detale (
  nr_faktury       NUMBER(5) NOT NULL,
  pozycja_faktury   NUMBER(2) NOT NULL,
  kod_produktu      NUMBER(5) NOT NULL,
  ilosc           NUMBER(2) NOT NULL,
  cena_zakupu       NUMBER(7,2) NOT NULL
);

ALTER TABLE p_faktura_detale ADD CONSTRAINT p_faktura_detale_pk PRIMARY KEY (
pozycja_faktury,nr_faktury );

CREATE TABLE p_faktura_naglowek (
  nr_faktury        NUMBER(5) NOT NULL,
  nr_klienta        NUMBER(5) NOT NULL,
  nr_pracownika      NUMBER(5) NOT NULL,
  data_wystawienia   DATE NOT NULL,
  wartosc_faktury    NUMBER(7,2) NOT NULL
);

ALTER TABLE p_faktura_naglowek ADD CONSTRAINT p_faktura_naglowek_pk PRIMARY KEY ( nr_faktury );

CREATE TABLE p_klient (
  nr_klienta   NUMBER(5) NOT NULL,
  nazwisko     VARCHAR2(20),
  email       VARCHAR2(20) NOT NULL
);
```

```
ALTER TABLE p_klient ADD CONSTRAINT p_klient_pk PRIMARY KEY ( nr_klienta );

CREATE TABLE p_pracownik (
   nr_pracownika        NUMBER(5) NOT NULL,
   nazwisko             VARCHAR2(20) NOT NULL,
   kod_pocztowy_apteki   VARCHAR2(6) NOT NULL
);

ALTER TABLE p_pracownik ADD CONSTRAINT p_pracownik_pk PRIMARY KEY ( nr_pracownika );

CREATE TABLE p_produkt (
   kod_produktu      NUMBER(5) NOT NULL,
   typ_produktu      VARCHAR2(15) NOT NULL,
   nazwa             VARCHAR2(15) NOT NULL,
   cena_katalogowa   NUMBER(5,2) NOT NULL
);

ALTER TABLE p_produkt ADD CONSTRAINT p_produkt_pk PRIMARY KEY ( kod_produktu );

CREATE TABLE p_przechowuje (
   kod_produktu   NUMBER(5) NOT NULL,
   kod_apteki     VARCHAR2(6) NOT NULL,
   ilosc          NUMBER(4) NOT NULL
);

ALTER TABLE p_przechowuje ADD CONSTRAINT p_przechowuje_pk PRIMARY KEY ( kod_produktu,kod_apteki );

CREATE TABLE p_slownik_adresy (
   kod_pocztowy   VARCHAR2(6) NOT NULL,
   wojewodztwo    VARCHAR2(15) NOT NULL,
   miejscowosc    VARCHAR2(15) NOT NULL
);

ALTER TABLE p_slownik_adresy ADD CONSTRAINT slownik_adresy_pk PRIMARY KEY ( kod_pocztowy );

CREATE TABLE p_slownik_produkt (
   typ_produktu   VARCHAR2(15) NOT NULL,
   stawka_vat     NUMBER(4,2) NOT NULL
);

ALTER TABLE p_slownik_produkt ADD CONSTRAINT slownik_produkt_pk PRIMARY KEY ( typ_produktu );

ALTER TABLE p_apteka
   ADD CONSTRAINT p_apteka_slownik_fk FOREIGN KEY ( kod_pocztowy )
      REFERENCES p_slownik_adresy ( kod_pocztowy );

ALTER TABLE p_faktura_detale
   ADD CONSTRAINT p_detale_faktura_fk FOREIGN KEY ( nr_faktury )
      REFERENCES p_faktura_naglowek ( nr_faktury )
         ON DELETE CASCADE;

ALTER TABLE p_faktura_naglowek
```

```
   ADD CONSTRAINT p_faktura_klient_fk FOREIGN KEY ( nr_klienta )
      REFERENCES p_klient ( nr_klienta )
         ON DELETE CASCADE;

ALTER TABLE p_faktura_naglowek
   ADD CONSTRAINT p_faktura_pracownik_fk FOREIGN KEY ( nr_pracownika )
      REFERENCES p_pracownik ( nr_pracownika );

ALTER TABLE p_faktura_detale
   ADD CONSTRAINT p_faktura_produkt_fk FOREIGN KEY ( kod_produktu )
      REFERENCES p_produkt ( kod_produktu );

ALTER TABLE p_pracownik
   ADD CONSTRAINT p_pracownik_apteka_fk FOREIGN KEY ( kod_pocztowy_apteki )
      REFERENCES p_apteka ( kod_pocztowy );

ALTER TABLE p_przechowuje
   ADD CONSTRAINT p_przechowuje_apteka_fk FOREIGN KEY ( kod_apteki )
      REFERENCES p_apteka ( kod_pocztowy )
         ON DELETE CASCADE;

ALTER TABLE p_przechowuje
   ADD CONSTRAINT p_przechowuje_p_produkt_fk FOREIGN KEY ( kod_produktu )
      REFERENCES p_produkt ( kod_produktu );

ALTER TABLE p_produkt
   ADD CONSTRAINT p_slownik_produkt_fk FOREIGN KEY ( typ_produktu )
      REFERENCES p_slownik_produkt ( typ_produktu );

-----------------------------------------------------------------
--
--     WPROWADZENIE PODSTAWOWYCH DANYCH DO SLOWNIKOW:
--
-----------------------------------------------------------------
/
INSERT INTO p_slownik_adresy ( kod_pocztowy, wojewodztwo, miejscowosc)
VALUES ( '02-930','Mazowieckie','Warszawa' );

INSERT INTO p_slownik_adresy ( kod_pocztowy, wojewodztwo, miejscowosc)
VALUES ( '30-430','Malopolskie','Krakow' );

INSERT INTO p_slownik_adresy ( kod_pocztowy, wojewodztwo, miejscowosc)
VALUES ( '15-127','Podlaskie','Bialystok' );

INSERT INTO p_slownik_adresy ( kod_pocztowy, wojewodztwo, miejscowosc)
VALUES ( '45-021','Opolskie','Opole' );

INSERT INTO p_slownik_adresy ( kod_pocztowy, wojewodztwo, miejscowosc)
VALUES ( '60-104','Wielkopolskie','Poznan' );

INSERT INTO p_slownik_adresy ( kod_pocztowy, wojewodztwo, miejscowosc)
VALUES ( '76-271','Pomorskie','Ustka' );
```

```
INSERT INTO p_slownik_adresy ( kod_pocztowy, wojewodztwo, miejscowosc)
VALUES ( '54-054','Dolnoslaskie','Wroclaw' );

INSERT INTO p_slownik_produkt ( typ_produktu, stawka_vat)
VALUES ( 'Suplement','17' );

INSERT INTO p_slownik_produkt ( typ_produktu, stawka_vat)
VALUES ( 'Lek','12' );

INSERT INTO p_slownik_produkt ( typ_produktu, stawka_vat)
VALUES ( 'Kosmetyk','23' );

----------------------------------------------------------------
--
--     TWORZENIE TRIGGEROW
--
----------------------------------------------------------------
/
create or replace TRIGGER p_tr_faktura
BEFORE DELETE OR INSERT OR UPDATE ON p_faktura_detale
FOR EACH ROW
BEGIN
 if inserting then
   update p_faktura_naglowek
   set wartosc_faktury = wartosc_faktury + (:NEW.ilosc*:NEW.cena_zakupu)
   where nr_faktury = :NEW.nr_faktury;
   update p_przechowuje
   set ilosc = ilosc - :NEW.ilosc
   where kod_produktu = :NEW.kod_produktu;

 elsif updating then
   update p_faktura_naglowek
   set wartosc_faktury = wartosc_faktury + (:OLD.cena_zakupu*(:NEW.ilosc-:OLD.ilosc))
   where nr_faktury = :OLD.nr_faktury;
   update p_przechowuje
   set ilosc = ilosc + :OLD.ilosc - :NEW.ilosc
   where kod_produktu = :NEW.kod_produktu;

 elsif deleting then
   update p_faktura_naglowek
   set wartosc_faktury = wartosc_faktury - (:OLD.cena_zakupu*:OLD.ilosc)
   where nr_faktury = :OLD.nr_faktury;
   update p_przechowuje
   set ilosc = ilosc + :OLD.ilosc
   where kod_produktu = :NEW.kod_produktu;

 end if;
END;
/
----------------------------------------------------------------
--
--     TWORZENIE SEKWENCJI
--
```

```
-----------------------------------------------------------------

--      SEKWENCJA GENERUJACA NR FAKTURY
CREATE SEQUENCE  P_SEQ_NR_FAKTURY
MINVALUE 1 MAXVALUE 99999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20
NOORDER  NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
  /
--      SEKWENCJA GENERUJACA NR KLIENTA
CREATE SEQUENCE  P_SEQ_NR_KLIENTA
MINVALUE 1 MAXVALUE 9999 INCREMENT BY 1 START WITH 1 CACHE 20
NOORDER  NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
/
--      SEKWENCJA GENERUJACA NR PRACOWNIKA
CREATE SEQUENCE  P_SEQ_NR_PRACOWNIKA
MINVALUE 1 MAXVALUE 9999 INCREMENT BY 1 START WITH 1 CACHE 20
NOORDER  NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
/
--      SEKWENCJA GENERUJACA KOD PRODUKTU
CREATE SEQUENCE  P_SEQ_KOD_PRODUKTU
MINVALUE 1 MAXVALUE 9999 INCREMENT BY 1 START WITH 1 CACHE 20
NOORDER  NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
/
-----------------------------------------------------------------
--
--      TWORZENIE FUNKCJI POZYSKUJACYCH ODPOWIEDNIE DANE
--
-----------------------------------------------------------------

--      POZYSKANIE LOSOWEGO TYPU PRODUKTU Z P_SLOWNIK_PRODUKT
CREATE OR REPLACE FUNCTION p_fn_daj_typ_produktu RETURN VARCHAR2 AS
 v_typ VARCHAR2(15) ;
BEGIN
 SELECT * INTO v_typ FROM (
          SELECT typ_produktu FROM  p_slownik_produkt
                    ORDER BY dbms_random.value)
WHERE ROWNUM = 1;
 RETURN v_typ;
END p_fn_daj_typ_produktu;
/
--      POZYSKANIE LOSOWEGO KODU POCZTOWEGO Z P_SLOWNIK_ADRESY
CREATE OR REPLACE FUNCTION p_fn_daj_adres RETURN VARCHAR2 AS
 v_kod VARCHAR2(6) ;
BEGIN
 SELECT * INTO v_kod FROM (
          SELECT kod_pocztowy FROM  p_slownik_adresy
                    ORDER BY dbms_random.value)
WHERE ROWNUM = 1;
 RETURN v_kod;
END p_fn_daj_adres;
/
--      POZYSKANIE LOSOWEGO KODU PRODUKTU Z P_PRODUKT
CREATE OR REPLACE FUNCTION p_fn_daj_kod_produktu RETURN NUMBER AS
 v_kod NUMBER ;
```

```
BEGIN
 SELECT * INTO v_kod FROM (
        SELECT kod_produktu FROM  p_produkt
                ORDER BY dbms_random.value)
WHERE ROWNUM = 1;
 RETURN v_kod;
END p_fn_daj_kod_produktu;
/
--    POZYSKANIE LOSOWEGO NR KLIENTA Z P_KLIENT
CREATE OR REPLACE FUNCTION p_fn_daj_nr_klienta RETURN NUMBER AS
 v_numer NUMBER ;
BEGIN
 SELECT * INTO v_numer FROM (
        SELECT nr_klienta FROM  p_klient
                ORDER BY dbms_random.value)
WHERE ROWNUM = 1;
 RETURN v_numer;
END p_fn_daj_nr_klienta;
/
--    POZYSKANIE LOSOWEGO KODU POCZTOWEGO APTEKI Z P_APTEKA
CREATE OR REPLACE FUNCTION p_fn_daj_kod_apteki RETURN VARCHAR2 AS
 v_kod VARCHAR2(6) ;
BEGIN
 SELECT * INTO v_kod FROM (
        SELECT kod_pocztowy FROM  p_apteka
                ORDER BY dbms_random.value)
WHERE ROWNUM = 1;
 RETURN v_kod;
END p_fn_daj_kod_apteki;
/
--    POZYSKANIE LOSOWEGO NR PRACOWNIKA Z P_PRACOWNIK
CREATE OR REPLACE FUNCTION p_fn_daj_nr_pracownika RETURN NUMBER AS
 v_numer NUMBER ;
BEGIN
 SELECT * INTO v_numer FROM (
        SELECT nr_pracownika FROM  p_pracownik
                ORDER BY dbms_random.value)
WHERE ROWNUM = 1;
 RETURN v_numer;
END p_fn_daj_nr_pracownika;
/
--    POZYSKANIE LOSOWEGO NR FAKTURY Z P_FAKTURA_NAGLOWEK
CREATE OR REPLACE FUNCTION p_fn_daj_nr_faktury RETURN NUMBER AS
 v_numer NUMBER ;
BEGIN
 SELECT * INTO v_numer FROM (
        SELECT nr_faktury FROM  p_faktura_naglowek
                ORDER BY dbms_random.value)
WHERE ROWNUM = 1;
 RETURN v_numer;
END p_fn_daj_nr_faktury;
/
----------------------------------------------------------------
```

```
--
--      TWORZENIE PROCEDUR WPROWADZAJACYH DANE
--
-----------------------------------------------------------------

-----------------------------------------------------------------
--      RECZNE DODANIE NOWEGO PRODUKTU
CREATE OR REPLACE PROCEDURE p_pr_dodaj_produkt
(v_kod in p_produkt.kod_produktu%type,
v_typ in p_produkt.typ_produktu%type,
v_nazwa p_produkt.nazwa%type,
v_cena p_produkt.cena_katalogowa%type)
AS
v_tmp_nazwa number;
v_tmp_kod number;
BEGIN
 select count(*) into v_tmp_nazwa from p_produkt
  where nazwa = v_nazwa;
  select count(*) into v_tmp_kod from p_produkt
  where kod_produktu = v_kod;

  if v_tmp_nazwa = 0 and v_tmp_kod = 0 then
      insert into p_produkt (kod_produktu,typ_produktu,nazwa,cena_katalogowa)
       values( v_kod,v_typ,v_nazwa,v_cena);
  else
      if v_tmp_nazwa = 0 then
      dbms_output.put_line('produkt o kodzie: '+v_kod+' juz istnieje');
      elsif v_tmp_kod = 0 then
      dbms_output.put_line('produkt o nazwie: '+v_nazwa+' juz istnieje');
      else
      dbms_output.put_line('produkt o kodzie: '+v_kod+' i nazwie: '+v_nazwa+' juz istnieje');
      end if;
  end if;
END p_pr_dodaj_produkt;
/
-----------------------------------------------------------------

-----------------------------------------------------------------
--      RECZNE DODANIE NOWEGO KLIENTA
CREATE OR REPLACE PROCEDURE p_pr_dodaj_klienta
(v_nazwisko in p_klient.nazwisko%type,
v_email in p_klient.email%type)
AS
v_tmp_email number;
BEGIN
 select count(*) into v_tmp_email from p_klient
  where email = v_email;

  if v_tmp_email = 0 then
    insert into p_klient (nr_klienta,nazwisko,email)
    values( P_SEQ_NR_KLIENTA.NEXTVAL,v_nazwisko,v_email);
  else
    dbms_output.put_line('klient o emailu: '+v_email+' juz istnieje');
```

```
 end if;
END p_pr_dodaj_klienta;
/
-----------------------------------------------------------------

-----------------------------------------------------------------
--      RECZNE DODANIE NOWEGO PRACOWNIKA
CREATE OR REPLACE PROCEDURE p_pr_dodaj_pracownika
(v_nrpracownika in p_pracownik.nr_pracownika%type,
v_nazwisko in p_pracownik.nazwisko%type,
v_kodapteki in p_pracownik.kod_pocztowy_apteki%type)
AS
v_tmp_nr number;

BEGIN
 select count(*) into v_tmp_nr from p_pracownik
  where nr_pracownika = v_nrpracownika;

  if v_tmp_nr = 0 then
      insert into p_pracownik (nr_pracownika,nazwisko,kod_pocztowy_apteki)
       values( v_nrpracownika,v_nazwisko,v_kodapteki);
  else
      dbms_output.put_line('pracownik o nr: '+v_nrpracownika+' juz istnieje');
  end if;
END p_pr_dodaj_pracownika;
/
-----------------------------------------------------------------

-----------------------------------------------------------------
--      RECZNE DODANIE NOWEJ APTEKI
CREATE OR REPLACE PROCEDURE p_pr_dodaj_apteke
(v_kod in p_apteka.kod_pocztowy%type,
v_ulica in p_apteka.ulica%type)
AS
v_tmp_kod number;

BEGIN
 select count(*) into v_tmp_kod from p_apteka
  where kod_pocztowy = v_kod;

  if v_tmp_kod = 0 then
      insert into p_apteka (kod_pocztowy,ulica)
       values( v_kod,v_ulica);
  else
      dbms_output.put_line('apteka o kodzie: '+v_kod+' juz istnieje');
  end if;
END p_pr_dodaj_apteke;
/

-----------------------------------------------------------------
--      RECZNE DODANIE X DANEGO PRODUKTU DO DANEJ APTEKI
CREATE OR REPLACE PROCEDURE p_pr_dodaj_przechowuje
(v_kod_produktu in p_przechowuje.kod_produktu%type,
```

```
v_kod_apteki in p_przechowuje.kod_apteki%type,
v_ilosc in p_przechowuje.ilosc%type)
AS
v_tmp number;
BEGIN
 select count(*) into v_tmp from p_przechowuje
  where kod_apteki = v_kod_apteki and kod_produktu = v_kod_produktu;

  if v_tmp = 0 then
      insert into p_przechowuje (kod_produktu,kod_apteki,ilosc)
       values( v_kod_produktu,v_kod_apteki,v_ilosc);
  else
      update p_przechowuje
      set ilosc = ilosc + v_ilosc
      where kod_apteki = v_kod_apteki and kod_produktu = v_kod_produktu;
  end if;
END p_pr_dodaj_przechowuje;
/
---------------------------------------------------------------------

---------------------------------------------------------------------
--      RECZNE STWORZENIE NOWEJ PUSTEJ FAKTURY
CREATE OR REPLACE PROCEDURE p_pr_dodaj_fakture
(v_nr_klienta in p_faktura_naglowek.nr_klienta%type,
v_nr_pracownika in p_faktura_naglowek.nr_pracownika%type,
v_data in p_faktura_naglowek.data_wystawienia%type default sysdate)
AS
BEGIN
   insert into p_faktura_naglowek
   (nr_faktury,nr_klienta,nr_pracownika,data_wystawienia,wartosc_faktury)
   values( p_seq_nr_faktury.nextval,v_nr_klienta,v_nr_pracownika,v_data,0);
END p_pr_dodaj_fakture;
/
---------------------------------------------------------------------

---------------------------------------------------------------------
--      RECZNE DODANIE POZYCJI DO FAKTURY
CREATE OR REPLACE PROCEDURE p_pr_dodaj_pozycje
(v_nr_faktury in p_faktura_detale.nr_faktury%type,
v_kod_produktu in p_faktura_detale.kod_produktu%type,
v_ilosc in p_faktura_detale.ilosc%type)
AS
v_tmp number;
v_tmp_cena number;
v_tmp_pozycja number;
v_tmp_wartosc number;
BEGIN
 select count(*) into v_tmp from p_faktura_detale
  where nr_faktury = v_nr_faktury and kod_produktu = v_kod_produktu;
 select cena_katalogowa into v_tmp_cena from p_produkt
  where kod_produktu = v_kod_produktu;
 select count(*) into v_tmp_pozycja from p_faktura_detale
  where nr_faktury = v_nr_faktury;
```

```
   if v_tmp = 0 then
      insert into p_faktura_detale
      (nr_faktury,pozycja_faktury,kod_produktu,ilosc,cena_zakupu)
       values( v_nr_faktury,v_tmp_pozycja+1,v_kod_produktu,v_ilosc,v_tmp_cena);
  else
      update p_faktura_detale
      set ilosc = v_ilosc
      where nr_faktury = v_nr_faktury and kod_produktu = v_kod_produktu;
  end if;
END p_pr_dodaj_pozycje;
/
-----------------------------------------------------------------

-----------------------------------------------------------------
--      PSEUDOLOSOWE DODANIE NOWEGO PRODUKTU
CREATE OR REPLACE PROCEDURE p_pr_generuj_produkt AS
v_kod_produktu number;
v_tmp_kod number;
v_typ_produktu varchar2(15);
v_nazwa varchar2(15);
v_nazwanr number;
v_cena_kat number;
BEGIN
v_kod_produktu := p_seq_kod_produktu.nextval;
select count(*) into v_tmp_kod from p_produkt
  where kod_produktu = v_kod_produktu;
while v_tmp_kod != 0
loop
v_kod_produktu := p_seq_kod_produktu.nextval;
select count(*) into v_tmp_kod from p_produkt
  where kod_produktu = v_kod_produktu;
end loop;

v_typ_produktu := p_fn_daj_typ_produktu();

select count(*) into v_nazwanr from p_produkt;
v_nazwa := 'produkt '||to_char(v_nazwanr+1);

select round(dbms_random.value(10,40),2) into v_cena_kat from dual;

p_pr_dodaj_produkt(v_kod_produktu,v_typ_produktu,v_nazwa,v_cena_kat);

END p_pr_generuj_produkt;
/
-----------------------------------------------------------------

-----------------------------------------------------------------
--      PSEUDOLOSOWE DODANIE NOWEGO KLIENTA
CREATE OR REPLACE PROCEDURE p_pr_generuj_klienta AS
v_nazwisko varchar2(20);
v_email varchar2(20);
v_nr number;
```

```
BEGIN

select count(*) into v_nr from p_klient;
v_nazwisko := 'nazwisko '||to_char(v_nr+1);
v_email := 'email '||to_char(v_nr+1);

p_pr_dodaj_klienta(v_nazwisko,v_email);

END p_pr_generuj_klienta;
/
-----------------------------------------------------------------

-----------------------------------------------------------------
--      PSEUDOLOSOWE DODANIE NOWEGO PRACOWNIKA
CREATE OR REPLACE PROCEDURE p_pr_generuj_pracownika AS
v_nazwisko varchar2(20);
v_nr_pracownika number;
v_tmp_nr number;
v_kod_apteki varchar(6);
BEGIN
v_nr_pracownika := p_seq_nr_pracownika.nextval;
select count(*) into v_tmp_nr from p_pracownik
  where nr_pracownika = v_nr_pracownika;
while v_tmp_nr != 0
loop
v_nr_pracownika := p_seq_nr_pracownika.nextval;
select count(*) into v_tmp_nr from p_pracownik
  where nr_pracownika = v_nr_pracownika;
end loop;

v_nazwisko := 'nazwisko '||to_char(v_nr_pracownika);

v_kod_apteki := p_fn_daj_kod_apteki();

p_pr_dodaj_pracownika(v_nr_pracownika,v_nazwisko,v_kod_apteki);

END p_pr_generuj_pracownika;
/
-----------------------------------------------------------------

-----------------------------------------------------------------
--      PSEUDOLOSOWE DODANIE NOWEJ APTEKI
CREATE OR REPLACE PROCEDURE p_pr_generuj_apteke AS
v_ulica varchar2(20);
v_ulicanr number;
v_tmp_kod number;
v_kod_apteki varchar(6);
v_tries number;
BEGIN
v_kod_apteki := p_fn_daj_adres();
select count(*) into v_tmp_kod from p_apteka
  where kod_pocztowy = v_kod_apteki;
v_tries := 0;
```

```
while v_tmp_kod != 0 and v_tries < 20
loop
v_kod_apteki := p_fn_daj_adres();
select count(*) into v_tmp_kod from p_apteka
  where kod_pocztowy = v_kod_apteki;
v_tries := v_tries +1;
end loop;
if v_tries < 20 then
select count(*) into v_ulicanr from p_apteka;
v_ulica := 'ulica nr '||to_char(v_ulicanr+1);

p_pr_dodaj_apteke(v_kod_apteki,v_ulica);
end if;
END p_pr_generuj_apteke;
/
----------------------------------------------------------------------

----------------------------------------------------------------------
--      PSEUDOLOSOWE DODANIE ISTNIEJACYCH PRODUKTOW DO ISTNIEJACEJ APTEKI
CREATE OR REPLACE PROCEDURE p_pr_generuj_przechowuje AS
v_kod_produktu number;
v_ilosc number;
v_kod_apteki varchar(6);
BEGIN
v_kod_apteki := p_fn_daj_kod_apteki();
v_kod_produktu := p_fn_daj_kod_produktu();

select round(dbms_random.value(50,200)) into v_ilosc from dual;

p_pr_dodaj_przechowuje(v_kod_produktu,v_kod_apteki,v_ilosc);

END p_pr_generuj_przechowuje;
/
----------------------------------------------------------------------

----------------------------------------------------------------------
CREATE OR REPLACE PROCEDURE p_pr_generuj_pozycje
--      PSEUDOLOSOWE WYSTAWIENIE JEDNEJ POZYCJI FAKTURY NA PODST. ISTNIEJACYCH DANYCH
(v_nr_faktury in p_faktura_naglowek.nr_faktury%type default p_seq_nr_faktury.currval) AS
v_kod_produktu number;
v_ilosc number;

BEGIN
v_kod_produktu := p_fn_daj_kod_produktu();

select round(dbms_random.value(1,3)) into v_ilosc from dual;

p_pr_dodaj_pozycje(v_nr_faktury,v_kod_produktu,v_ilosc);

END p_pr_generuj_pozycje;
/
----------------------------------------------------------------------
```

```
-------------------------------------------------------------------
--      PSEUDOLOSOWE WYSTAWIENIE JEDNEJ FAKTURY NA PODST. ISTNIEJACYCH DANYCH
CREATE OR REPLACE PROCEDURE p_pr_generuj_fakture
(v_data in p_faktura_naglowek.data_wystawienia%type default sysdate) AS
v_nr_klienta number;
v_nr_pracownika number;
v_ilosc_pozycji number;
BEGIN
v_nr_klienta := p_fn_daj_nr_klienta();
v_nr_pracownika := p_fn_daj_nr_pracownika();
p_pr_dodaj_fakture(v_nr_klienta,v_nr_pracownika,v_data);

select round(dbms_random.value(1,5)) into v_ilosc_pozycji from dual;

for counter in 1..v_ilosc_pozycji
loop
p_pr_generuj_pozycje(p_seq_nr_faktury.currval);
end loop;

END p_pr_generuj_fakture;
/
-------------------------------------------------------------------

-------------------------------------------------------------------
--      PSEUDOLOSOWE WYSTAWIENIE LOSOWA ILOSC FAKTUR POMIEDY ZADANYMI DATAMI NA PODST.
ISTNIEJACYCH DANYCH
CREATE OR REPLACE PROCEDURE p_pr_generuj_wiecej_faktur
(v_poczatek in p_faktura_naglowek.data_wystawienia%type,
v_koniec in p_faktura_naglowek.data_wystawienia%type default sysdate) AS
v_ilosc_faktur_w_dniu number;
v_data date;
BEGIN
v_data := v_poczatek;
while v_koniec - v_data != 0
loop
select round(dbms_random.value(1,5)) into v_ilosc_faktur_w_dniu from dual;
for counter in 1..v_ilosc_faktur_w_dniu
loop
p_pr_generuj_fakture(v_data);
end loop;
v_data := v_data + 1;
end loop;
END p_pr_generuj_wiecej_faktur;
/
-------------------------------------------------------------------


-------------------------------------------------------------------
--      PSEUDOLOSOWE WYGENEROWANIE CALEJ BAZY DANYCH
CREATE OR REPLACE PROCEDURE p_pr_generuj_baze AS
v_ile number;
BEGIN
select round(dbms_random.value(3,5)) into v_ile from dual;
```

```
for counter in 1..v_ile
loop
p_pr_generuj_apteke();
end loop;

select round(dbms_random.value(15,25)) into v_ile from dual;
for counter in 1..v_ile
loop
p_pr_generuj_klienta();
end loop;

select round(dbms_random.value(9,15)) into v_ile from dual;
for counter in 1..v_ile
loop
p_pr_generuj_pracownika();
end loop;

select round(dbms_random.value(20,40)) into v_ile from dual;
for counter in 1..v_ile
loop
p_pr_generuj_produkt();
end loop;

select round(dbms_random.value(30,100)) into v_ile from dual;
for counter in 1..v_ile
loop
p_pr_generuj_przechowuje();
end loop;

p_pr_generuj_wiecej_faktur(to_date('2022/01/01','YYYY/MM/DD'),to_date('2024/12/31','YYYY/MM/DD'));

update p_przechowuje
set ilosc = (select round(dbms_random.value(10,50)) from dual) + kod_produktu;

END p_pr_generuj_baze;
/
-----------------------------------------------------------------


-----------------------------------------------------------------
--
--      KONIEC CZESCI DEKLARACYJNEJ
--
-----------------------------------------------------------------



-----------------------------------------------------------------
--      WYGENEROWNIE BAZY DANYCH
execute p_pr_generuj_baze;
/
-----------------------------------------------------------------



-----------------------------------------------------------------
```

```
--
--      TWORZENIE PERSPEKTYW
--
-----------------------------------------------------------------


-----------------------------------------------------------------
--      PRZYCHOD W POSZCZEGOLNYCH MIESIACACH
create or replace view p_przychod_miesieczny as
select extract(year from data_wystawienia) Rok,
(case extract(month from data_wystawienia)
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) Miesiac,
to_char(nvl(sum(wartosc_faktury),0)*100,999999999.99) Przychod
from p_faktura_naglowek
GROUP BY extract(year from data_wystawienia), extract(month from data_wystawienia)
order by extract(year from data_wystawienia), extract(month from data_wystawienia);
/
-----------------------------------------------------------------


-----------------------------------------------------------------
--      PRZYCHOD Z POSZCZEGOLNYCH PRODUKTOW
create or replace view p_przychod_z_produktow as
select pro.kod_produktu, pro.nazwa, pro.typ_produktu,
to_char(nvl(sum(det.ilosc*cena_zakupu),0)*100,999999999.99) Przychod
from p_faktura_detale det right join p_produkt pro on pro.KOD_PRODUKTU = det.KOD_PRODUKTU
group by pro.kod_produktu, pro.nazwa, pro.typ_produktu
order by Przychod desc;
/
-----------------------------------------------------------------


-----------------------------------------------------------------
--      PRZYCHOD GENEROWANY PRZEZ POSZCEGOLNYCH PRACOWNIKOW
create or replace view p_przychod_pracownikow as
select extract(year from data_wystawienia) Rok,pra.nr_pracownika, pra.nazwisko, pra.kod_pocztowy_apteki as
apteka,
to_char(nvl(sum(wartosc_faktury),0)*100,999999999.99) Przychod
from p_faktura_naglowek nag right join p_pracownik pra on
 pra.NR_PRACOWNIKA = nag.NR_PRACOWNIKA
group by extract(year from data_wystawienia), pra.nr_pracownika, pra.nazwisko, pra.kod_pocztowy_apteki
order by Rok, Przychod desc;
/
-----------------------------------------------------------------
```

```
-----------------------------------------------------------------
--      PRZYCHOD GENEROWANY PRZEZ POSZCZEGOLNYCH KLIENTOW
create or replace view p_przychod_z_klientow as
select kli.nr_klienta, kli.nazwisko,
to_char(nvl(sum(wartosc_faktury),0)*100,999999999.99) Przychod
from p_faktura_naglowek nag
right join p_klient kli on kli.nr_klienta = nag.nr_klienta
left join p_pracownik pra on pra.NR_PRACOWNIKA = nag.NR_PRACOWNIKA
group by kli.nr_klienta, kli.nazwisko
order by Przychod desc;
/
-----------------------------------------------------------------


-----------------------------------------------------------------
--      PRZYCHOD GENEROWANY PRZEZ POSZCZEGOLNE APTEKI
create or replace view p_przychod_z_aptek as
select extract(year from data_wystawienia) Rok, apt.kod_pocztowy, adr.wojewodztwo, adr.miejscowosc,
apt.ulica,
to_char(nvl(sum(wartosc_faktury),0)*100,999999999.99) Przychod
from p_faktura_naglowek nag
join p_pracownik pra on pra.NR_PRACOWNIKA = nag.NR_PRACOWNIKA
right join p_apteka apt on apt.kod_pocztowy = pra.kod_pocztowy_apteki
join p_slownik_adresy adr on adr.kod_pocztowy = apt.kod_pocztowy
group by extract(year from data_wystawienia), apt.kod_pocztowy, adr.wojewodztwo, adr.miejscowosc,
apt.ulica
order by Rok, Przychod desc;
/
-----------------------------------------------------------------
```

6. Skrypt deinstalujący projekt.

```
DROP TRIGGER p_tr_faktura;

DROP TABLE p_apteka CASCADE CONSTRAINTS;
DROP TABLE p_faktura_detale CASCADE CONSTRAINTS;
DROP TABLE p_faktura_naglowek CASCADE CONSTRAINTS;
DROP TABLE p_klient CASCADE CONSTRAINTS;
DROP TABLE p_pracownik CASCADE CONSTRAINTS;
DROP TABLE p_produkt CASCADE CONSTRAINTS;
DROP TABLE p_przechowuje CASCADE CONSTRAINTS;
DROP TABLE p_slownik_adresy CASCADE CONSTRAINTS;
DROP TABLE p_slownik_produkt CASCADE CONSTRAINTS;

DROP SEQUENCE P_SEQ_NR_FAKTURY;
DROP SEQUENCE P_SEQ_NR_KLIENTA;
DROP SEQUENCE P_SEQ_NR_PRACOWNIKA;
DROP SEQUENCE P_SEQ_KOD_PRODUKTU;

DROP FUNCTION p_fn_daj_typ_produktu;
DROP FUNCTION p_fn_daj_adres;
```

```
DROP FUNCTION p_fn_daj_kod_produktu;
DROP FUNCTION p_fn_daj_nr_klienta;
DROP FUNCTION p_fn_daj_kod_apteki;
DROP FUNCTION p_fn_daj_nr_pracownika;
DROP FUNCTION p_fn_daj_nr_faktury;

DROP PROCEDURE p_pr_dodaj_produkt;
DROP PROCEDURE p_pr_dodaj_klienta;
DROP PROCEDURE p_pr_dodaj_pracownika;
DROP PROCEDURE p_pr_dodaj_apteke;
DROP PROCEDURE p_pr_dodaj_przechowuje;
DROP PROCEDURE p_pr_dodaj_fakture;
DROP PROCEDURE p_pr_dodaj_pozycje;

DROP PROCEDURE p_pr_generuj_produkt;
DROP PROCEDURE p_pr_generuj_klienta;
DROP PROCEDURE p_pr_generuj_pracownika;
DROP PROCEDURE p_pr_generuj_apteke;
DROP PROCEDURE p_pr_generuj_przechowuje;
DROP PROCEDURE p_pr_generuj_pozycje;
DROP PROCEDURE p_pr_generuj_fakture;
DROP PROCEDURE p_pr_generuj_wiecej_faktur;
DROP PROCEDURE p_pr_generuj_baze;

DROP VIEW p_przychod_miesieczny;
DROP VIEW p_przychod_z_produktow;
DROP VIEW p_przychod_pracownikow;
DROP VIEW p_przychod_z_klientow;
DROP VIEW p_przychod_z_aptek;
```

7. Instrukcja instalacji projektu

W celu instalacji projektu wystarczy uruchomić skrypt instalacyjny.

8. Sprawdzanie poprawności instalacji

W celu sprawdzenia poprawności działania programu posłużyć się można dowolną funkcją dodającą lub generującą dane. Ja posłużyłam się procedurą generującą bazę, a więc i pośrednio każdą istniejąca w skrypcie. Wyniki widać na końcu dokumentacji, gdzie umieściłam przykładowe wykorzystanie perspektyw, wszystkie procedury oraz funkcje działają poprawnie.

Funkcje dodające chronią program przez wprowadzaniem tych samych danych wielokrotnie. Trigger aktualizuje zarówno stan ilość produktów w aptekach jak i wartość faktury.

Wszystkie funkcje są case-sensitive, tak więc wartość argumentów należy wprowadzać dokładnie. Również procedury „dodaj” nie zawierają dodatkowych zabezpieczeń przed wprowadzaniem niewłaściwych danych jedynie przed ich powielaniem.

9. Demonstracja użycia perspektyw:

1. Dochód uzyskany przez wszystkie apteki w każdym miesiącu.

```
  CREATE OR REPLACE FORCE EDITIONABLE VIEW "WCY24BDZ_11"."P_PRZYCHOD_MIESIECZNY" ("ROK",
"MIESIAC", "PRZYCHOD") AS
  select extract(year from data_wystawienia) Rok,
(case extract(month from data_wystawienia)
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) Miesiac,
to_char(nvl(sum(wartosc_faktury),0)*100,999999999.99) Przychod
from p_faktura_naglowek
GROUP BY extract(year from data_wystawienia), extract(month from data_wystawienia)
order by extract(year from data_wystawienia), extract(month from data_wystawienia);
```

ZAPYTANIE: Dochód miesięczny aptek z podziałem na miesiące o przychodzie powyżej średniego dochodu

```
SELECT ROK, MIESIAC, PRZYCHOD
FROM P_PRZYCHOD_MIESIECZNY
WHERE PRZYCHOD > (
   SELECT AVG(PRZYCHOD)
   FROM P_PRZYCHOD_MIESIECZNY
)
AND ROWNUM <= 10
ORDER BY ROK DESC, MIESIAC;
```

ROK	MIESIAC	PRZYCHOD
2023	Luty	13223,25
2023	Styczen	13877,99
2022	Czerwiec	12956,65
2022	Grudzien	13055,60
2022	Lipiec	13480,06
2022	Maj	13860,63
2022	Marzec	14703,40
2022	Pazdziernik	14817,79
2022	Sierpien	13353,98
2022	Wrzesien	14059,97

2. Dochód wygenerowany przez każdego z pracowników.

```
  CREATE OR REPLACE FORCE EDITIONABLE VIEW "WCY24BDZ_11"."P_PRZYCHOD_PRACOWNIKOW" ("ROK",
"NR_PRACOWNIKA", "NAZWISKO", "APTEKA", "PRZYCHOD") AS
  select extract(year from data_wystawienia) Rok,pra.nr_pracownika, pra.nazwisko, pra.kod_pocztowy_apteki
as apteka,
to_char(nvl(sum(wartosc_faktury),0)*100,999999999.99) Przychod
from p_faktura_naglowek nag right join p_pracownik pra on
 pra.NR_PRACOWNIKA = nag.NR_PRACOWNIKA
group by extract(year from data_wystawienia), pra.nr_pracownika, pra.nazwisko, pra.kod_pocztowy_apteki
order by Rok, Przychod desc;
```

ZAPYTANIE: Przychód pracowników w województwach, gdzie pracownik z najwyższym przychodem przekroczył 15000

```
SELECT ROK, NR_PRACOWNIKA, NAZWISKO, APTEKA, PRZYCHOD
FROM P_PRZYCHOD_PRACOWNIKOW
WHERE APTEKA IN (
   SELECT APTEKA
   FROM P_PRZYCHOD_PRACOWNIKOW
   GROUP BY APTEKA
   HAVING MAX(PRZYCHOD) > 15000
)
AND ROWNUM <= 10
ORDER BY PRZYCHOD DESC;
```

ROK	NR_PRACOWNIKA	NAZWISKO	APTEKA	PRZYCHOD
2022	5	nazwisko 5	30-430	18325,42
2022	3	nazwisko 3	15-127	16599,22
2022	4	nazwisko 4	15-127	16320,66
2022	8	nazwisko 8	02-930	15863,95
2022	6	nazwisko 6	30-430	15516,35
2022	9	nazwisko 9	02-930	15452,58
2022	1	nazwisko 1	30-430	15371,52
2022	7	nazwisko 7	02-930	15338,18
2022	10	nazwisko 10	15-127	14333,06
2022	2	nazwisko 2	02-930	14330,90

3. Dochód wygenerowany przez każdą z aptek w poszczególnych latach.

```
  CREATE OR REPLACE FORCE EDITIONABLE VIEW "WCY24BDZ_11"."P_PRZYCHOD_Z_APTEK" ("ROK",
"KOD_POCZTOWY", "WOJEWODZTWO", "MIEJSCOWOSC", "ULICA", "PRZYCHOD") AS
  select extract(year from data_wystawienia) Rok, apt.kod_pocztowy, adr.wojewodztwo, adr.miejscowosc,
apt.ulica,
to_char(nvl(sum(wartosc_faktury),0)*100,999999999.99) Przychod
from p_faktura_naglowek nag
join p_pracownik pra on pra.NR_PRACOWNIKA = nag.NR_PRACOWNIKA
right join p_apteka apt on apt.kod_pocztowy = pra.kod_pocztowy_apteki
join p_slownik_adresy adr on adr.kod_pocztowy = apt.kod_pocztowy
group by extract(year from data_wystawienia), apt.kod_pocztowy, adr.wojewodztwo, adr.miejscowosc,
apt.ulica
order by Rok, Przychod desc;
```

ZAPYTANIE: Apteki generujące największy dochód w województwach o łącznym przychodzie większym niż 100 000

```
SELECT ROK, KOD_POCZTOWY, WOJEWODZTWO, MIEJSCOWOSC, ULICA, PRZYCHOD
FROM P_PRZYCHOD_Z_APTEK
WHERE WOJEWODZTWO IN (
   SELECT WOJEWODZTWO
   FROM P_PRZYCHOD_Z_APTEK
   GROUP BY WOJEWODZTWO
   HAVING SUM(PRZYCHOD) > 100000
)
AND ROWNUM <= 5
ORDER BY PRZYCHOD DESC;
```

ROK	KOD_PO	WOJEWODZTWO	MIEJSCOWOSC	ULICA	PRZYCHOD
2023	02-930	Mazowieckie	Warszawa	ulica nr 2	67865,67
2022	02-930	Mazowieckie	Warszawa	ulica nr 2	60985,61
2022	30-430	Malopolskie	Krakow	ulica nr 3	49213,29
2022	15-127	Podlaskie	Bialystok	ulica nr 1	47252,94
2023	15-127	Podlaskie	Bialystok	ulica nr 1	44185,82

4. Dochód pozyskany ze sprzedaży każdego z produktów.

```
  CREATE OR REPLACE FORCE EDITIONABLE VIEW "WCY24BDZ_11"."P_PRZYCHOD_Z_PRODUKTOW"
("KOD_PRODUKTU", "NAZWA", "TYP_PRODUKTU", "PRZYCHOD") AS
  select pro.kod_produktu, pro.nazwa, pro.typ_produktu,
to_char(nvl(sum(det.ilosc*cena_zakupu),0)*100,999999999.99) Przychod
from p_faktura_detale det right join p_produkt pro on pro.KOD_PRODUKTU = det.KOD_PRODUKTU
group by pro.kod_produktu, pro.nazwa, pro.typ_produktu
order by Przychod desc;
```

ZAPYTANIE: Dochód z produktów w kategoriach, gdzie produkty o największym dochodzie stanowią więcej niż 10% całkowitego dochodu

```
SELECT KOD_PRODUKTU, NAZWA, TYP_PRODUKTU, PRZYCHOD
FROM P_PRZYCHOD_Z_PRODUKTOW
WHERE TYP_PRODUKTU IN (
   SELECT TYP_PRODUKTU
   FROM P_PRZYCHOD_Z_PRODUKTOW
   GROUP BY TYP_PRODUKTU
   HAVING MAX(PRZYCHOD) > SUM(PRZYCHOD) * 0.1
)
AND ROWNUM <= 10
ORDER BY PRZYCHOD DESC;
```

KOD_PRODUKTU	NAZWA	TYP_PRODUKTU	PRZYCHOD
11	produkt 11	Suplement	30148,20
16	produkt 16	Kosmetyk	29067,62
14	produkt 14	Suplement	27924,20
4	produkt 4	Suplement	26601,36
6	produkt 6	Lek	26586,39
17	produkt 17	Kosmetyk	25570,86
10	produkt 10	Kosmetyk	25564,20
19	produkt 19	Lek	25150,50
18	produkt 18	Kosmetyk	24354,88
20	produkt 20	Lek	23205,46

5. Łączną wartość zakupów każdego z klientów

```
  CREATE OR REPLACE FORCE EDITIONABLE VIEW "WCY24BDZ_11"."P_PRZYCHOD_Z_KLIENTOW"
("NR_KLIENTA", "NAZWISKO", "PRZYCHOD") AS
  select kli.nr_klienta, kli.nazwisko,
to_char(nvl(sum(wartosc_faktury),0)*100,999999999.99) Przychod
from p_faktura_naglowek nag
right join p_klient kli on kli.nr_klienta = nag.nr_klienta
left join p_pracownik pra on pra.NR_PRACOWNIKA = nag.NR_PRACOWNIKA
group by kli.nr_klienta, kli.nazwisko
order by Przychod desc;
```

ZAPYTANIE: Klienci, którzy dokonali największych zakupów w aptekach o łącznym przychodzie większym niż 5 000

```
SELECT NR_KLIENTA, NAZWISKO, PRZYCHOD
FROM P_PRZYCHOD_Z_KLIENTOW
WHERE NR_KLIENTA IN (
   SELECT DISTINCT NR_KLIENTA
   FROM P_FAKTURA_NAGLOWEK
   WHERE NR_PRACOWNIKA IN (
      SELECT NR_PRACOWNIKA
      FROM P_PRZYCHOD_PRACOWNIKOW
      WHERE APTEKA IN (
         SELECT KOD_POCZTOWY
         FROM P_PRZYCHOD_Z_APTEK
         GROUP BY KOD_POCZTOWY
         HAVING SUM(PRZYCHOD) > 5000
      )
   )
)
AND ROWNUM <= 10
ORDER BY PRZYCHOD DESC;
```

NR_KLIENTA	NAZWISKO	PRZYCHOD
5	nazwisko 5	23315,84
12	nazwisko 12	22173,07
14	nazwisko 14	21324,77
21	nazwisko 21	20638,45
2	nazwisko 2	20534,64
24	nazwisko 24	19473,92
13	nazwisko 13	18926,70
9	nazwisko 9	18763,16
18	nazwisko 18	18569,53
20	nazwisko 20	17465,02