WOJSKOWA AKADEMIA TECHNICZNA

Bazy danych

Dokumentacja projektu

Prowadzący – mgr inż. Józef Woźniak

Wykonał – Bartosz Sobieski

Grupa I7Y5S1

Spis treści

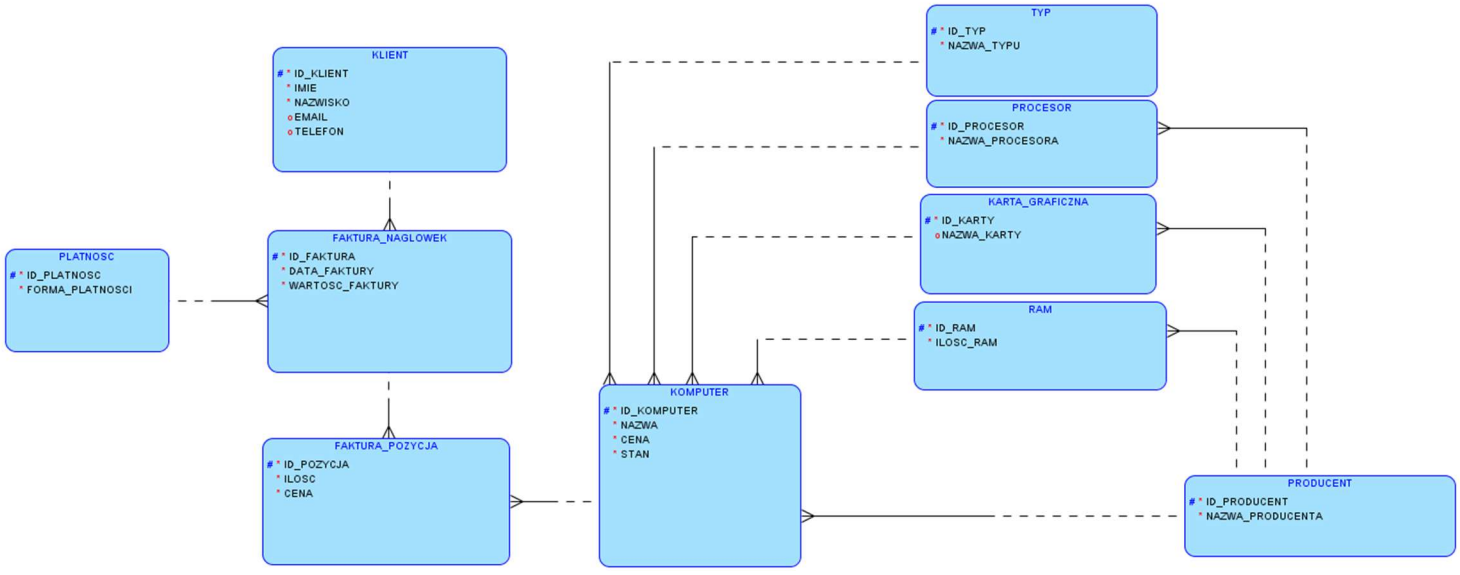
1. Analiza biznesowa projektowanej rzeczywistości

Model bazy danych stworzony na potrzeby projektu zrealizowany został na bazie sklepu ze sprzętem komputerowym. Model składa się z 10 tabel, w tym z 6 tabel słownikowych. W tabelach przechowywane są dane klientów, wystawionych faktur, sposobów płatności, komputerów oraz komponentów z których te komputery się składają. Zgodnie z założeniami projektu zapewniono fakturowanie wielopozycyjne.

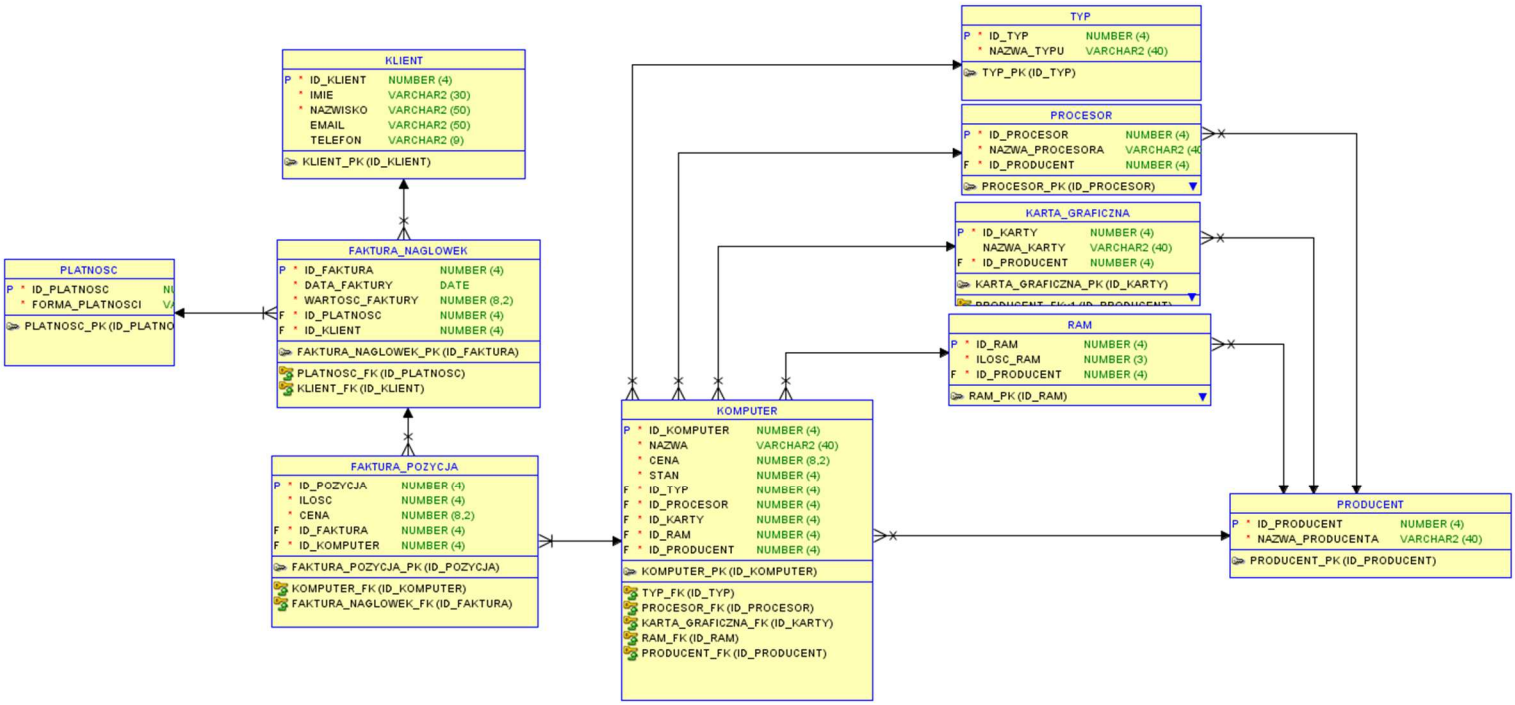
Utworzony model pozwala nam na między innymi na ewidencje klientów i sprzedawanych towarów, ale także na analizę zebranych danych by wyznaczyć przychody miesięczne, sprzedaż poszczególnych produktów itp.

2. Model logiczny i relacyjny bazy danych

2.1 Model logiczny



2.2 Model relacyjny



3. Oprogramowanie tworzące bazę danych oraz generator danych

Do zaprojektowania bazy danych użyto narzędzia Oracle Data Modeler. Następnie baza została wdrożona za pomocą Oracle SQL Developera, gdzie wprowadzono przykładowe dane, utworzono dodatkowe obiekty bazodanowe oraz generator faktur. Dane wytworzone przez generator a także utworzone wcześniej widoki zostały użyte w programie Jaspersoft iReport Designer do utworzenia przykładowych raportów.

3.1. Sekwencje zapewniające autonumerację kluczy głównych tabel

```
CREATE SEQUENCE seq_faktura_naglowek START WITH 1 INCREMENT BY 1
MAXVALUE 9999 MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_faktura_pozycja START WITH 1 INCREMENT BY 1
MAXVALUE 9999 MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_karta_graficzna START WITH 1 INCREMENT BY 1
MAXVALUE 9999 MINVALUE 1 NOCACHE;
```

```
CREATE SEQUENCE seq_klient START WITH 1 INCREMENT BY 1 MAXVALUE
9999 MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_komputer START WITH 1 INCREMENT BY 1 MAXVALUE 999
MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_platnosc START WITH 1 INCREMENT BY 1 MAXVALUE 9999
MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_procesor START WITH 1 INCREMENT BY 1 MAXVALUE 9999
MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_producent START WITH 1 INCREMENT BY 1 MAXVALUE 9999
MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_ram START WITH 1 INCREMENT BY 1 MAXVALUE 9999
MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_typ START WITH 1 INCREMENT BY 1 MAXVALUE 9999
MINVALUE 1 NOCACHE;
```

3.2. Wyzwalacze zapewniające numerację klucza głównego przy wpisywaniu danych do tabeli

```
CREATE OR REPLACE TRIGGER TR_INS_FAKTURA_NAGLOWEK
   BEFORE INSERT ON FAKTURA_NAGLOWEK
   FOR EACH ROW
begin
        :NEW.ID_FAKTURA:=SEQ_FAKTURA_NAGLOWEK.nextval;
end;

CREATE OR REPLACE TRIGGER TR_INS_KARTA_GRAFICZNA
   BEFORE INSERT ON KARTA_GRAFICZNA
   FOR EACH ROW
begin
:NEW.ID_KARTY:=SEQ_KARTA_GRAFICZNA.nextval;
end;

CREATE OR REPLACE TRIGGER TR_INS_KLIENT
   BEFORE INSERT ON KLIENT
   FOR EACH ROW
begin
:NEW.ID_KLIENT:=SEQ_KLIENT.nextval;
end;

CREATE OR REPLACE TRIGGER TR_INS_KOMPUTER
   BEFORE INSERT ON KOMPUTER
```

```
   FOR EACH ROW
begin
:NEW.ID_KOMPUTER:=SEQ_KOMPUTER.nextval;
end;

CREATE OR REPLACE TRIGGER TR_INS_PLATNOSC
   BEFORE INSERT ON PLATNOSC
   FOR EACH ROW
begin
:NEW.ID_PLATNOSC:=SEQ_PLATNOSC.nextval;
end;

CREATE OR REPLACE TRIGGER TR_INS_PROCESOR
   BEFORE INSERT ON PROCESOR
   FOR EACH ROW
begin
:NEW.ID_PROCESOR:=SEQ_PROCESOR.nextval;
end;

CREATE OR REPLACE TRIGGER TR_INS_PRODUCENT
   BEFORE INSERT ON PRODUCENT
   FOR EACH ROW
begin
:NEW.ID_PRODUCENT:=SEQ_PRODUCENT.nextval;
end;

CREATE OR REPLACE TRIGGER TR_INS_RAM
   BEFORE INSERT ON RAM
   FOR EACH ROW
begin
:NEW.ID_RAM:=SEQ_RAM.nextval;
end;

CREATE OR REPLACE TRIGGER TR_INS_TYP
   BEFORE INSERT ON TYP
   FOR EACH ROW
begin
:NEW.ID_TYP:=SEQ_TYP.nextval;
end;
```

3.3. Wyzwalacz tabeli Faktura_pozycja numerujący klucz główny oraz aktualizujący tabele Faktura_nagłówek przy operacji INSERT

```
CREATE OR REPLACE TRIGGER TR_INS_FAKTURA_POZYCJA
   BEFORE INSERT ON FAKTURA_POZYCJA
   FOR EACH ROW
```

```
DECLARE
  V_ilosc number;
begin
    :NEW.ID_POZYCJA:=SEQ_FAKTURA_POZYCJA.nextval;

    select CENA*:NEW.ILOSC into :NEW.CENA
    from komputer
    where komputer.id_komputer = :NEW.id_komputer;

    update komputer
    set stan=stan-:NEW.ilosc
    where komputer.id_komputer = :NEW.id_komputer;

    update faktura_naglowek
    set wartosc_faktury = wartosc_faktury + :NEW.cena
    where faktura_naglowek.id_faktura = :NEW.id_faktura;

end;
```

3.4. Funkcje

3.4.1. Funkcja zwracająca cenę danego komputera

```
CREATE OR REPLACE FUNCTION FN_DAJ_CENE (V_ID_KOMPUTER IN NUMBER)
RETURN NUMBER AS

V_CENA KOMPUTER.CENA%TYPE;

BEGIN

SELECT KOMPUTER.CENA INTO V_CENA

FROM KOMPUTER

WHERE ID_KOMPUTER = V_ID_KOMPUTER;


RETURN V_CENA;


END FN_DAJ_CENE;
```

3.4.2. Funkcja zwracająca stan danego komputera

```
CREATE OR REPLACE FUNCTION FN_DAJ_STAN(V_ID_KOMPUTER IN NUMBER)
RETURN NUMBER AS

V_STAN KOMPUTER.STAN%TYPE;

BEGIN

SELECT STAN INTO V_STAN

FROM KOMPUTER
```

```
WHERE ID_KOMPUTER=V_ID_KOMPUTER;
RETURN V_STAN;
END FN_DAJ_STAN;
```

3.4.3. Funkcja zwracająca obecny numer sekwencji tabeli Faktura_naglowek

```
CREATE OR REPLACE FUNCTION FN_DAJ_LAST RETURN NUMBER AS
V_LAST NUMBER;
BEGIN
   SELECT last_number INTO V_LAST
 FROM user_sequences
 WHERE sequence_name = 'SEQ_FAKTURA_NAGLOWEK';
RETURN V_LAST;
END FN_DAJ_LAST;
```

3.4.4. Funkcja zwracająca losową datę od dzisiejszej daty maksymalnie 90 dni wstecz

```
CREATE OR REPLACE FUNCTION FN_LOS_DATA RETURN DATE AS
DNI NUMBER;
BEGIN
DNI:=ROUND(DBMS_RANDOM.VALUE(1,90));
RETURN (SYSDATE - DNI);
END FN_LOS_DATA;
```

3.4.5. Funkcja zwracająca losową ilość z przedziału 1-4

```
CREATE OR REPLACE FUNCTION FN_LOS_ILOSC RETURN NUMBER AS
BEGIN
RETURN ROUND(DBMS_RANDOM.VALUE(1,4));
END;
```

3.4.6. Funkcja losująca losowego klienta

```
CREATE OR REPLACE FUNCTION FN_LOS_KLIENT RETURN NUMBER AS
 V_ID KLIENT.ID_KLIENT%TYPE;
BEGIN
```

```
 SELECT * INTO V_ID
 FROM(SELECT ID_KLIENT FROM KLIENT ORDER BY DBMS_RANDOM.VALUE)
 WHERE ROWNUM=1;


 RETURN V_ID;


END FN_LOS_KLIENT;
```

3.4.7. Funkcja losująca ilość pozycji z przedziału 1-4

```
CREATE OR REPLACE FUNCTION FN_LOS_POZYCJE RETURN NUMBER AS
BEGIN
RETURN ROUND(DBMS_RANDOM.VALUE(1,4));
END FN_LOS_POZYCJE;
```

3.4.8. Funkcja losująca komputer

```
create or replace FUNCTION FN_LOS_KOMPUTER
  RETURN NUMBER
AS
  V_ID KOMPUTER.ID_KOMPUTER%TYPE;
BEGIN
  SELECT *
  INTO V_ID
  FROM
    (SELECT ID_KOMPUTER FROM KOMPUTER ORDER BY DBMS_RANDOM.VALUE
    )
  WHERE ROWNUM=1;
  RETURN V_ID;
END FN_LOS_KOMPUTER;
```

3.4.9. Funkcja losująca sposób płatności

```
CREATE OR REPLACE FUNCTION FN_LOS_PLATNOSC RETURN NUMBER AS
   V_ID PLATNOSC.ID_PLATNOSC%TYPE;
BEGIN
```

```
 SELECT * INTO V_ID
 FROM(SELECT ID_PLATNOSC FROM PLATNOSC ORDER BY
DBMS_RANDOM.VALUE)
 WHERE ROWNUM=1;


 RETURN V_ID;
END FN_LOS_PLATNOSC;
```

3.4.10. Funkcja losująca kartę graficzną

```
create or replace FUNCTION FN_LOS_KARTA
 RETURN NUMBER
AS
 V_ID KARTA_GRAFICZNA.ID_KARTY%TYPE;
BEGIN
 SELECT *
 INTO V_ID
 FROM
  (SELECT ID_KARTY FROM KARTA_GRAFICZNA ORDER BY
DBMS_RANDOM.VALUE
  )
 WHERE ROWNUM=1;
 RETURN V_ID;
END FN_LOS_KARTA;
```

3.4.11. Funkcja losująca procesor

```
create or replace FUNCTION FN_LOS_PROCESOR
 RETURN NUMBER
AS
 V_ID PROCESOR.ID_PROCESOR%TYPE;
BEGIN
 SELECT *
 INTO V_ID
 FROM
```

```
  (SELECT ID_PROCESOR FROM PROCESOR ORDER BY DBMS_RANDOM.VALUE
  )
 WHERE ROWNUM=1;
 RETURN V_ID;
END FN_LOS_PROCESOR;
```

3.4.12. Funkcja losująca producenta

```
create or replace FUNCTION FN_LOS_PRODUCENT
 RETURN NUMBER
AS
 V_ID PRODUCENT.ID_PRODUCENT%TYPE;
BEGIN
 SELECT *
 INTO V_ID
 FROM
  (SELECT ID_PRODUCENT FROM PRODUCENT ORDER BY
DBMS_RANDOM.VALUE
  )
 WHERE ROWNUM=1;
 RETURN V_ID;
END FN_LOS_PRODUCENT;
```

3.4.13. Funkcja losująca RAM

```
create or replace FUNCTION FN_LOS_RAM
 RETURN NUMBER
AS
 V_ID RAM.ID_RAM%TYPE;
BEGIN
 SELECT *
 INTO V_ID
 FROM
  (SELECT ID_RAM FROM RAM ORDER BY DBMS_RANDOM.VALUE
  )
```

```
 WHERE ROWNUM=1;
 RETURN V_ID;
END FN_LOS_RAM;
```

3.4.14. Funkcja losująca typ

```
create or replace FUNCTION FN_LOS_TYP
 RETURN NUMBER
AS
 V_ID TYP.ID_TYP%TYPE;
BEGIN
 SELECT *
 INTO V_ID
 FROM
  (SELECT ID_TYP FROM TYP ORDER BY DBMS_RANDOM.VALUE
  )
 WHERE ROWNUM=1;
 RETURN V_ID;
END FN_LOS_TYP;
```

3.5. Procedury

3.5.1. Procedura dodająca nowy komputer złożony z losowych komponentów dostępnych w bazie. Procedura korzysta z wcześniej zadeklarowanych funkcji.

```
create or replace PROCEDURE PR_DODAJ_KOMPUTER (V_NAZWA IN
VARCHAR,V_CENA IN FLOAT,V_STAN IN NUMBER)AS
V_TYP NUMBER;
V_KARTA NUMBER;
V_PROCESOR NUMBER;
V_RAM NUMBER;
V_PRODUCENT NUMBER;
BEGIN
V_TYP := FN_LOS_TYP;
```

```
V_KARTA := FN_LOS_KARTA;

V_RAM := FN_LOS_RAM;

V_PRODUCENT := FN_LOS_PRODUCENT;

V_PROCESOR := FN_LOS_PROCESOR;


INSERT INTO
KOMPUTER(NAZWA,CENA,STAN,ID_TYP,ID_KARTY,ID_PROCESOR,ID_RAM,ID_PR
ODUCENT)
VALUES(V_NAZWA,V_CENA,V_STAN,V_TYP,V_KARTA,V_PROCESOR,V_RAM,V_P
RODUCENT);


END PR_DODAJ_KOMPUTER;
```

3.5.2. Procedura dodająca dla losowego klienta nową fakturę na losową ilość losowych przedmiotów. Procedura korzysta z wcześniej zadeklarowanych funkcji.

```
CREATE OR REPLACE PROCEDURE PR_DODAJ_FAKTURE AS

V_PLATNOSC NUMBER;

V_KLIENT NUMBER;

V_DATA DATE;

V_ILOSC NUMBER;

V_POZYCJE NUMBER;

V_KOMPUTER NUMBER;

V_CENA NUMBER(8,2);

V_STAN NUMBER;

V_FAKTURA NUMBER;

V_I NUMBER;

BEGIN

  COMMIT;

  V_PLATNOSC:=FN_LOS_PLATNOSC;

  V_KLIENT:=FN_LOS_KLIENT;

  V_DATA:=FN_LOS_DATA;
```

```
  V_POZYCJE:=FN_LOS_POZYCJE;
  V_I := 1;
  INSERT INTO
FAKTURA_NAGLOWEK(DATA_FAKTURY,WARTOSC_FAKTURY,ID_PLATNOSC,ID_K
LIENT) VALUES(V_DATA,0,V_PLATNOSC,V_KLIENT);
  V_FAKTURA:=FN_DAJ_LAST;
  V_FAKTURA:=V_FAKTURA-1;
  FOR V_I IN 1..V_POZYCJE
  LOOP
    V_KOMPUTER:=FN_LOS_KOMPUTER;
    V_ILOSC:=FN_LOS_ILOSC;
    V_CENA:=FN_DAJ_CENE(V_KOMPUTER);
    V_STAN:=FN_DAJ_STAN(V_KOMPUTER);
    IF V_STAN<V_ILOSC
    THEN
      ROLLBACK;
      dbms_output.put_line('Niewystarczajaca ilosc produktow');
      EXIT;
    ELSE
      INSERT INTO
FAKTURA_POZYCJA(ILOSC,CENA,ID_FAKTURA,ID_KOMPUTER)
VALUES(V_ILOSC,V_CENA,V_FAKTURA,V_KOMPUTER);
    END IF;
  END LOOP;
  COMMIT;
END PR_DODAJ_FAKTURE;
```

3.5.3. Procedura tworząca nowe faktury zależnie w liczbie zależnej od podanego do funkcji argumentu liczbowego. Procedura korzysta z wyżej zadeklarowanej procedury generowania pojedynczej faktury.

```
CREATE OR REPLACE PROCEDURE PR_DODAJ_WIELE_FAKTUR (V_ILOSC IN
NUMBER) AS
```

```
V_I NUMBER;
BEGIN
V_I:=1;
FOR V_I IN 1..V_ILOSC
LOOP
   PR_DODAJ_FAKTURE;
END LOOP;
END PR_DODAJ_WIELE_FAKTUR;
```

3.6. Widoki

3.6.1. Zestawienie wszystkich komputerów oraz zysków z ich sprzedaży

```
CREATE OR REPLACE VIEW KOMPUTERY_ZYSK AS
SELECT NAZWA_PRODUCENTA||' '||NAZWA AS
"Komputer",NAZWA_PROCESORA AS "Procesor",NAZWA_KARTY AS "Karta
Graficzna",RAM.ILOSC_RAM||' GB' AS "RAM",SUM(FAKTURA_POZYCJA.ILOSC)
AS "Ilosc sprzedanych",SUM(FAKTURA_POZYCJA.CENA) AS "Zysk_calkowity"
FROM KOMPUTER
INNER JOIN PRODUCENT ON PRODUCENT.ID_PRODUCENT =
KOMPUTER.ID_PRODUCENT
INNER JOIN FAKTURA_POZYCJA ON
FAKTURA_POZYCJA.ID_KOMPUTER=KOMPUTER.ID_KOMPUTER
INNER JOIN PROCESOR ON
PROCESOR.ID_PROCESOR=KOMPUTER.ID_PROCESOR
INNER JOIN KARTA_GRAFICZNA ON
KARTA_GRAFICZNA.ID_KARTY=KOMPUTER.ID_KARTY
INNER JOIN RAM ON RAM.ID_RAM=KOMPUTER.ID_RAM
GROUP BY NAZWA_PRODUCENTA||'
'||NAZWA,NAZWA_PROCESORA,NAZWA_KARTY,ILOSC_RAM
ORDER BY "Zysk_calkowity" DESC;
```

3.6.2. Zestawienie faktur zrealizowanych w przeciągu ostatniego miesiąca

```
CREATE OR REPLACE VIEW OSTATNI_MIESIAC AS
SELECT *
```

```
FROM FAKTURA_NAGLOWEK

WHERE DATA_FAKTURY BETWEEN SYSDATE - 30 AND SYSDATE

ORDER BY DATA_FAKTURY;
```

3.6.3. Zestawienie dwudziestu klientów których wydatki były największe

```
CREATE OR REPLACE VIEW TOP20_KLIENTOW AS

SELECT IMIE AS "Imie",NAZWISKO AS "Nazwisko",COUNT(*) AS "Ilosc zamowien",SUM(WARTOSC_FAKTURY) AS "Wydatki"

FROM KLIENT

INNER JOIN FAKTURA_NAGLOWEK ON FAKTURA_NAGLOWEK.ID_KLIENT = KLIENT.ID_KLIENT

WHERE ROWNUM <= 20

GROUP BY IMIE,NAZWISKO

ORDER BY "Wydatki" DESC;
```

4. Skrypt wdrożeniowy instalujący i deinstalujący projekt

4.1. Skrypt instalacyjny wraz z wprowadzaniem danych

```
CREATE SEQUENCE seq_faktura_naglowek START WITH 1 INCREMENT BY 1 MAXVALUE 9999 MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_faktura_pozycja START WITH 1 INCREMENT BY 1 MAXVALUE 9999 MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_karta_graficzna START WITH 1 INCREMENT BY 1 MAXVALUE 9999 MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_klient START WITH 1 INCREMENT BY 1 MAXVALUE 9999 MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_komputer START WITH 1 INCREMENT BY 1 MAXVALUE 999 MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_platnosc START WITH 1 INCREMENT BY 1 MAXVALUE 9999 MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_procesor START WITH 1 INCREMENT BY 1 MAXVALUE 9999 MINVALUE 1 NOCACHE;

CREATE SEQUENCE seq_producent START WITH 1 INCREMENT BY 1 MAXVALUE 9999 MINVALUE 1 NOCACHE;
```

```
CREATE SEQUENCE seq_ram START WITH 1 INCREMENT BY 1 MAXVALUE 9999 MINVALUE 1
NOCACHE;

CREATE SEQUENCE seq_typ START WITH 1 INCREMENT BY 1 MAXVALUE 9999 MINVALUE 1
NOCACHE;

CREATE TABLE komputer (
   id_komputer    NUMBER(4) NOT NULL,
   nazwa          VARCHAR2(40) NOT NULL,
   cena           NUMBER(8, 2) NOT NULL,
   stan           NUMBER(4) NOT NULL,
   id_typ         NUMBER(4) NOT NULL,
   id_procesor    NUMBER(4) NOT NULL,
   id_karty       NUMBER(4) NOT NULL,
   id_ram         NUMBER(4) NOT NULL,
   id_producent   NUMBER(4) NOT NULL
)
LOGGING;
ALTER TABLE komputer ADD CONSTRAINT komputer_pk PRIMARY KEY ( id_komputer );

CREATE OR REPLACE FUNCTION fn_daj_cene (
   v_id_komputer IN NUMBER
) RETURN NUMBER AS
   v_cena komputer.cena%TYPE;
BEGIN
   SELECT
      komputer.cena
   INTO v_cena
   FROM
      komputer
   WHERE
      id_komputer = v_id_komputer;
   RETURN v_cena;
END fn_daj_cene;
/
```

```
CREATE OR REPLACE FUNCTION fn_daj_last RETURN NUMBER AS
   v_last NUMBER;
BEGIN
   SELECT
      last_number
   INTO v_last
   FROM
      user_sequences
   WHERE
      sequence_name = 'SEQ_FAKTURA_NAGLOWEK';

   RETURN v_last;
END fn_daj_last;
/
CREATE OR REPLACE FUNCTION fn_daj_stan (
   v_id_komputer IN NUMBER
) RETURN NUMBER AS
   v_stan komputer.stan%TYPE;
BEGIN
   SELECT
      stan
   INTO v_stan
   FROM
      komputer
   WHERE
      id_komputer = v_id_komputer;

   RETURN v_stan;
END fn_daj_stan;
/
```

```
CREATE OR REPLACE FUNCTION fn_los_data RETURN DATE AS
   dni NUMBER;
BEGIN
   dni := round(dbms_random.value(1, 90));
   return(SYSDATE - dni);
END fn_los_data;
/
CREATE OR REPLACE FUNCTION fn_los_ilosc RETURN NUMBER AS
BEGIN
   RETURN round(dbms_random.value(1, 4));
END;
/
CREATE TABLE karta_graficzna (
   id_karty       NUMBER(4) NOT NULL,
   nazwa_karty    VARCHAR2(40),
   id_producent   NUMBER(4) NOT NULL
)
LOGGING;
ALTER TABLE karta_graficzna ADD CONSTRAINT karta_graficzna_pk PRIMARY KEY ( id_karty );
CREATE OR REPLACE FUNCTION fn_los_karta RETURN NUMBER AS
   v_id karta_graficzna.id_karty%TYPE;
BEGIN
   SELECT
      *
   INTO v_id
   FROM
      (
         SELECT
            id_karty
         FROM
            karta_graficzna
```

```
        ORDER BY
            dbms_random.value
    )
  WHERE
    ROWNUM = 1;

  RETURN v_id;
END fn_los_karta;
/
CREATE TABLE klient (
  id_klient   NUMBER(4) NOT NULL,
  imie        VARCHAR2(30) NOT NULL,
  nazwisko    VARCHAR2(50) NOT NULL,
  email       VARCHAR2(50),
  telefon     VARCHAR2(9)
)
LOGGING;
ALTER TABLE klient ADD CONSTRAINT klient_pk PRIMARY KEY ( id_klient );
CREATE OR REPLACE FUNCTION fn_los_klient RETURN NUMBER AS
  v_id klient.id_klient%TYPE;
BEGIN
  SELECT
    *
  INTO v_id
  FROM
    (
      SELECT
        id_klient
      FROM
        klient
      ORDER BY
```

```
            dbms_random.value
        )
    WHERE
        ROWNUM = 1;

    RETURN v_id;
END fn_los_klient;
/
CREATE OR REPLACE FUNCTION fn_los_komputer RETURN NUMBER AS
    v_id komputer.id_komputer%TYPE;
BEGIN
    SELECT
        *
    INTO v_id
    FROM
        (
            SELECT
                id_komputer
            FROM
                komputer
            ORDER BY
                dbms_random.value
        )
    WHERE
        ROWNUM = 1;

    RETURN v_id;
END fn_los_komputer;
/
CREATE TABLE platnosc (
    id_platnosc       NUMBER(4) NOT NULL,
```

```
    forma_platnosci   VARCHAR2(40) NOT NULL
)
LOGGING;
ALTER TABLE platnosc ADD CONSTRAINT platnosc_pk PRIMARY KEY ( id_platnosc );
CREATE OR REPLACE FUNCTION fn_los_platnosc RETURN NUMBER AS
    v_id platnosc.id_platnosc%TYPE;
BEGIN
    SELECT
        *
    INTO v_id
    FROM
        (
            SELECT
                id_platnosc
            FROM
                platnosc
            ORDER BY
                dbms_random.value
        )
    WHERE
        ROWNUM = 1;

    RETURN v_id;
END fn_los_platnosc;
/
CREATE OR REPLACE FUNCTION fn_los_pozycje RETURN NUMBER AS
BEGIN
    RETURN round(dbms_random.value(1, 4));
END fn_los_pozycje;
/
CREATE TABLE procesor (
```

```
   id_procesor       NUMBER(4) NOT NULL,
   nazwa_procesora   VARCHAR2(40) NOT NULL,
   id_producent      NUMBER(4) NOT NULL
)
LOGGING;
ALTER TABLE procesor ADD CONSTRAINT procesor_pk PRIMARY KEY ( id_procesor );
CREATE OR REPLACE FUNCTION fn_los_procesor RETURN NUMBER AS
   v_id procesor.id_procesor%TYPE;
BEGIN
   SELECT
      *
   INTO v_id
   FROM
      (
         SELECT
            id_procesor
         FROM
            procesor
         ORDER BY
            dbms_random.value
      )
   WHERE
      ROWNUM = 1;

   RETURN v_id;
END fn_los_procesor;
/
CREATE TABLE producent (
   id_producent      NUMBER(4) NOT NULL,
   nazwa_producenta   VARCHAR2(40) NOT NULL
)
```

```
LOGGING;
ALTER TABLE producent ADD CONSTRAINT producent_pk PRIMARY KEY ( id_producent );
CREATE OR REPLACE FUNCTION fn_los_producent RETURN NUMBER AS
  v_id producent.id_producent%TYPE;
BEGIN
  SELECT
    *
  INTO v_id
  FROM
    (
      SELECT
        id_producent
      FROM
        producent
      ORDER BY
        dbms_random.value
    )
  WHERE
    ROWNUM = 1;

  RETURN v_id;
END fn_los_producent;
/
CREATE TABLE ram (
  id_ram        NUMBER(4) NOT NULL,
  ilosc_ram     NUMBER(3) NOT NULL,
  id_producent  NUMBER(4) NOT NULL
)
LOGGING;

ALTER TABLE ram ADD CONSTRAINT ram_pk PRIMARY KEY ( id_ram );
```

```
CREATE OR REPLACE FUNCTION fn_los_ram RETURN NUMBER AS
   v_id ram.id_ram%TYPE;
BEGIN
   SELECT
      *
   INTO v_id
   FROM
      (
         SELECT
            id_ram
         FROM
            ram
         ORDER BY
            dbms_random.value
      )
   WHERE
      ROWNUM = 1;

   RETURN v_id;
END fn_los_ram;
/
CREATE TABLE typ (
   id_typ       NUMBER(4) NOT NULL,
   nazwa_typu   VARCHAR2(40) NOT NULL
)
LOGGING;
ALTER TABLE typ ADD CONSTRAINT typ_pk PRIMARY KEY ( id_typ );
CREATE OR REPLACE FUNCTION fn_los_typ RETURN NUMBER AS
   v_id typ.id_typ%TYPE;
BEGIN
```

```
    SELECT
        *
    INTO v_id
    FROM
        (
            SELECT
                id_typ
            FROM
                typ
            ORDER BY
                dbms_random.value
        )
    WHERE
        ROWNUM = 1;

    RETURN v_id;
END fn_los_typ;
/
CREATE TABLE faktura_naglowek (
    id_faktura        NUMBER(4) NOT NULL,
    data_faktury      DATE NOT NULL,
    wartosc_faktury   NUMBER(8, 2) NOT NULL,
    id_platnosc       NUMBER(4) NOT NULL,
    id_klient         NUMBER(4) NOT NULL
)
LOGGING;
ALTER TABLE faktura_naglowek ADD CONSTRAINT faktura_naglowek_pk PRIMARY KEY (
id_faktura );
CREATE TABLE faktura_pozycja (
    id_pozycja    NUMBER(4) NOT NULL,
    ilosc         NUMBER(4) NOT NULL,
```

```
   cena         NUMBER(8, 2) NOT NULL,
   id_faktura    NUMBER(4) NOT NULL,
   id_komputer   NUMBER(4) NOT NULL
)
LOGGING;
ALTER TABLE faktura_pozycja ADD CONSTRAINT faktura_pozycja_pk PRIMARY KEY ( id_pozycja
);
CREATE OR REPLACE PROCEDURE PR_DODAJ_FAKTURE AS
V_PLATNOSC NUMBER;
V_KLIENT NUMBER;
V_DATA DATE;
V_ILOSC NUMBER;
V_POZYCJE NUMBER;
V_KOMPUTER NUMBER;
V_CENA NUMBER(8,2);
V_STAN NUMBER;
V_FAKTURA NUMBER;
V_I NUMBER;
BEGIN
   COMMIT;
   V_PLATNOSC:=FN_LOS_PLATNOSC;
   V_KLIENT:=FN_LOS_KLIENT;
   V_DATA:=FN_LOS_DATA;
   V_POZYCJE:=FN_LOS_POZYCJE;
   V_I := 1;
   INSERT INTO
FAKTURA_NAGLOWEK(DATA_FAKTURY,WARTOSC_FAKTURY,ID_PLATNOSC,ID_KLIENT)
VALUES(V_DATA,0,V_PLATNOSC,V_KLIENT);
   V_FAKTURA:=FN_DAJ_LAST;
   V_FAKTURA:=V_FAKTURA-1;
   FOR V_I IN 1..V_POZYCJE
   LOOP
```

```
      V_KOMPUTER:=FN_LOS_KOMPUTER;
      V_ILOSC:=FN_LOS_ILOSC;
      V_CENA:=FN_DAJ_CENE(V_KOMPUTER);
      V_STAN:=FN_DAJ_STAN(V_KOMPUTER);
      IF V_STAN<V_ILOSC
      THEN
         ROLLBACK;
         dbms_output.put_line('Niewystarczajaca ilosc produktow');
         EXIT;
      ELSE
         INSERT INTO FAKTURA_POZYCJA(ILOSC,CENA,ID_FAKTURA,ID_KOMPUTER)
VALUES(V_ILOSC,V_CENA,V_FAKTURA,V_KOMPUTER);
      END IF;
   END LOOP;
   COMMIT;
END PR_DODAJ_FAKTURE;
/
create or replace PROCEDURE PR_DODAJ_KOMPUTER (V_NAZWA IN VARCHAR,V_CENA IN
FLOAT,V_STAN IN NUMBER)AS
V_TYP NUMBER;
V_KARTA NUMBER;
V_PROCESOR NUMBER;
V_RAM NUMBER;
V_PRODUCENT NUMBER;
BEGIN
V_TYP := FN_LOS_TYP;
V_KARTA := FN_LOS_KARTA;
V_RAM := FN_LOS_RAM;
V_PRODUCENT := FN_LOS_PRODUCENT;
V_PROCESOR := FN_LOS_PROCESOR;
```

```
INSERT INTO
KOMPUTER(NAZWA,CENA,STAN,ID_TYP,ID_KARTY,ID_PROCESOR,ID_RAM,ID_PRODUCENT)
VALUES(V_NAZWA,V_CENA,V_STAN,V_TYP,V_KARTA,V_PROCESOR,V_RAM,V_PRODUCENT);


END PR_DODAJ_KOMPUTER;
/
CREATE OR REPLACE PROCEDURE PR_DODAJ_WIELE_FAKTUR (V_ILOSC IN NUMBER) AS
V_I NUMBER;
BEGIN
V_I:=1;
FOR V_I IN 1..V_ILOSC
LOOP
   PR_DODAJ_FAKTURE;
END LOOP;
END PR_DODAJ_WIELE_FAKTUR;
/
ALTER TABLE faktura_pozycja
   ADD CONSTRAINT faktura_naglowek_fk FOREIGN KEY ( id_faktura )
      REFERENCES faktura_naglowek ( id_faktura )
         ON DELETE CASCADE
   NOT DEFERRABLE;

ALTER TABLE komputer
   ADD CONSTRAINT karta_graficzna_fk FOREIGN KEY ( id_karty )
      REFERENCES karta_graficzna ( id_karty )
         ON DELETE CASCADE
   NOT DEFERRABLE;

ALTER TABLE faktura_naglowek
   ADD CONSTRAINT klient_fk FOREIGN KEY ( id_klient )
      REFERENCES klient ( id_klient )
         ON DELETE CASCADE
```

```
    NOT DEFERRABLE;

ALTER TABLE faktura_pozycja
    ADD CONSTRAINT komputer_fk FOREIGN KEY ( id_komputer )
        REFERENCES komputer ( id_komputer )
    NOT DEFERRABLE;

ALTER TABLE faktura_naglowek
    ADD CONSTRAINT platnosc_fk FOREIGN KEY ( id_platnosc )
        REFERENCES platnosc ( id_platnosc )
    NOT DEFERRABLE;

ALTER TABLE komputer
    ADD CONSTRAINT procesor_fk FOREIGN KEY ( id_procesor )
        REFERENCES procesor ( id_procesor )
            ON DELETE CASCADE
    NOT DEFERRABLE;

ALTER TABLE komputer
    ADD CONSTRAINT producent_fk FOREIGN KEY ( id_producent )
        REFERENCES producent ( id_producent )
            ON DELETE CASCADE
    NOT DEFERRABLE;

ALTER TABLE karta_graficzna
    ADD CONSTRAINT producent_fkv1 FOREIGN KEY ( id_producent )
        REFERENCES producent ( id_producent )
            ON DELETE CASCADE
    NOT DEFERRABLE;

ALTER TABLE procesor
```

```
   ADD CONSTRAINT producent_fkv2 FOREIGN KEY ( id_producent )
      REFERENCES producent ( id_producent )
         ON DELETE CASCADE
   NOT DEFERRABLE;

ALTER TABLE ram
   ADD CONSTRAINT producent_fkv4 FOREIGN KEY ( id_producent )
      REFERENCES producent ( id_producent )
         ON DELETE CASCADE
   NOT DEFERRABLE;

ALTER TABLE komputer
   ADD CONSTRAINT ram_fk FOREIGN KEY ( id_ram )
      REFERENCES ram ( id_ram )
         ON DELETE CASCADE
   NOT DEFERRABLE;

ALTER TABLE komputer
   ADD CONSTRAINT typ_fk FOREIGN KEY ( id_typ )
      REFERENCES typ ( id_typ )
         ON DELETE CASCADE
   NOT DEFERRABLE;

CREATE OR REPLACE TRIGGER TR_INS_FAKTURA_NAGLOWEK
   BEFORE INSERT ON FAKTURA_NAGLOWEK
   FOR EACH ROW
begin
          :NEW.ID_FAKTURA:=SEQ_FAKTURA_NAGLOWEK.nextval;
end;
/
CREATE OR REPLACE TRIGGER TR_INS_FAKTURA_POZYCJA
```

```
   BEFORE INSERT ON FAKTURA_POZYCJA
   FOR EACH ROW
DECLARE
   V_ilosc number;
begin
      :NEW.ID_POZYCJA:=SEQ_FAKTURA_POZYCJA.nextval;

      select CENA*:NEW.ILOSC into :NEW.CENA
      from komputer
      where komputer.id_komputer = :NEW.id_komputer;

      update komputer
      set stan=stan-:NEW.ilosc
      where komputer.id_komputer = :NEW.id_komputer;

      update faktura_naglowek
      set wartosc_faktury = wartosc_faktury + :NEW.cena
      where faktura_naglowek.id_faktura = :NEW.id_faktura;

end;
/
CREATE OR REPLACE TRIGGER TR_INS_KARTA_GRAFICZNA
   BEFORE INSERT ON KARTA_GRAFICZNA
   FOR EACH ROW
begin
:NEW.ID_KARTY:=SEQ_KARTA_GRAFICZNA.nextval;
end;
/
CREATE OR REPLACE TRIGGER TR_INS_KLIENT
   BEFORE INSERT ON KLIENT
   FOR EACH ROW
```

```
begin
:NEW.ID_KLIENT:=SEQ_KLIENT.nextval;
end;
/
CREATE OR REPLACE TRIGGER TR_INS_KOMPUTER
   BEFORE INSERT ON KOMPUTER
   FOR EACH ROW
begin
:NEW.ID_KOMPUTER:=SEQ_KOMPUTER.nextval;
end;
/
CREATE OR REPLACE TRIGGER TR_INS_PLATNOSC
   BEFORE INSERT ON PLATNOSC
   FOR EACH ROW
begin
:NEW.ID_PLATNOSC:=SEQ_PLATNOSC.nextval;
end;
/
CREATE OR REPLACE TRIGGER TR_INS_PROCESOR
   BEFORE INSERT ON PROCESOR
   FOR EACH ROW
begin
:NEW.ID_PROCESOR:=SEQ_PROCESOR.nextval;
end;
/
CREATE OR REPLACE TRIGGER TR_INS_PRODUCENT
   BEFORE INSERT ON PRODUCENT
   FOR EACH ROW
begin
:NEW.ID_PRODUCENT:=SEQ_PRODUCENT.nextval;
end;
```

```
/
CREATE OR REPLACE TRIGGER TR_INS_RAM
   BEFORE INSERT ON RAM
   FOR EACH ROW
begin
:NEW.ID_RAM:=SEQ_RAM.nextval;
end;
/
CREATE OR REPLACE TRIGGER TR_INS_TYP
   BEFORE INSERT ON TYP
   FOR EACH ROW
begin
:NEW.ID_TYP:=SEQ_TYP.nextval;
end;
/
CREATE OR REPLACE VIEW KOMPUTERY_ZYSK AS
SELECT NAZWA_PRODUCENTA||' '||NAZWA AS "Komputer",NAZWA_PROCESORA AS
"Procesor",NAZWA_KARTY AS "Karta Graficzna",RAM.ILOSC_RAM||' GB' AS
"RAM",SUM(FAKTURA_POZYCJA.ILOSC) AS "Ilosc
sprzedanych",SUM(FAKTURA_POZYCJA.CENA) AS "Zysk_calkowity"
FROM KOMPUTER
INNER JOIN PRODUCENT ON PRODUCENT.ID_PRODUCENT = KOMPUTER.ID_PRODUCENT
INNER JOIN FAKTURA_POZYCJA ON
FAKTURA_POZYCJA.ID_KOMPUTER=KOMPUTER.ID_KOMPUTER
INNER JOIN PROCESOR ON PROCESOR.ID_PROCESOR=KOMPUTER.ID_PROCESOR
INNER JOIN KARTA_GRAFICZNA ON KARTA_GRAFICZNA.ID_KARTY=KOMPUTER.ID_KARTY
INNER JOIN RAM ON RAM.ID_RAM=KOMPUTER.ID_RAM
GROUP BY NAZWA_PRODUCENTA||'
'||NAZWA,NAZWA_PROCESORA,NAZWA_KARTY,ILOSC_RAM
ORDER BY "Zysk_calkowity" DESC;
/
CREATE OR REPLACE VIEW OSTATNI_MIESIAC AS
SELECT *
```

```
FROM FAKTURA_NAGLOWEK

WHERE DATA_FAKTURY BETWEEN SYSDATE - 30 AND SYSDATE

ORDER BY DATA_FAKTURY;

/

CREATE OR REPLACE VIEW TOP20_KLIENTOW AS

SELECT IMIE AS "Imie",NAZWISKO AS "Nazwisko",COUNT(*) AS "Ilosc
zamowien",SUM(WARTOSC_FAKTURY) AS "Wydatki"

FROM KLIENT

INNER JOIN FAKTURA_NAGLOWEK ON FAKTURA_NAGLOWEK.ID_KLIENT = KLIENT.ID_KLIENT

WHERE ROWNUM <= 20

GROUP BY IMIE,NAZWISKO

ORDER BY "Wydatki" DESC;

/

--PLATNOSC

insert into PLATNOSC (forma_platnosci) values ('Karta');

insert into PLATNOSC (forma_platnosci) values ('Gotowka');

insert into PLATNOSC (forma_platnosci) values ('BLIK');

--KLIENT

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Fern', 'Gashion',
'fgashion0@pbs.org', '165598110');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Xaviera', 'Franzel',
'xfranzel1@earthlink.net', '124053900');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Kelli', 'Bontein',
'kbontein2@ucoz.ru', '206710993');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Lorna', 'Dewdeny',
'ldewdeny3@un.org', '845526361');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Blancha', 'Forstall',
'bforstall4@fema.gov', '573724709');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Arabela', 'Sondland',
'asondland5@ucoz.com', '994968634');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Lanae', 'Corben',
'lcorben6@google.fr', '184423861');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Herta', 'Easen',
'heasen7@paypal.com', '538441699');
```

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Ferguson', 'Trounson', 'ftrounson8@mediafire.com', '142604022');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Fields', 'Dunlap', 'fdunlap9@live.com', '251964619');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Kaycee', 'Edgeson', 'kedgesona@princeton.edu', '796564282');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Christian', 'Amburgy', 'camburgyb@cnn.com', '897461444');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Vivyan', 'Bernollet', 'vbernolletc@archive.org', '503538798');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Sarge', 'Smallcomb', 'ssmallcombd@123-reg.co.uk', '367053080');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Meade', 'June', 'mjunee@4shared.com', '839872189');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Caspar', 'Sandey', 'csandeyf@indiatimes.com', '695978424');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Monah', 'Copes', 'mcopesg@1und1.de', '422468789');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Jere', 'Helkin', 'jhelkinh@seesaa.net', '871710239');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Roby', 'Bonnor', 'rbonnori@about.me', '170928853');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Amabel', 'Ethelston', 'aethelstonj@hao123.com', '278757043');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Abbye', 'McIlvoray', 'amcilvorayk@ifeng.com', '844171545');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Cobb', 'Crosi', 'ccrosil@cloudflare.com', '439878148');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Margaux', 'MacAvddy', 'mmacavddym@globo.com', '119961194');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Carl', 'Assender', 'cassendern@usda.gov', '463411457');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Rand', 'Deakins', 'rdeakinso@marketwatch.com', '269984690');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Ely', 'Kornacki', 'ekornackip@godaddy.com', '955738566');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Son', 'Garnsey', 'sgarnseyq@pagesperso-orange.fr', '583883254');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Olly', 'Hambright', 'ohambrightr@google.com.hk', '335139075');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Zaccaria', 'Tolle', 'ztolles@hatena.ne.jp', '771846623');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Georgianne', 'Heaton', 'gheatont@cbsnews.com', '861815457');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Basilius', 'Prine', 'bprineu@odnoklassniki.ru', '213192358');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Chrotoem', 'Peacock', 'cpeacockv@opensource.org', '425609867');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Audrye', 'Peiro', 'apeirow@indiatimes.com', '967177301');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Junia', 'Bartak', 'jbartakx@soundcloud.com', '318749992');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Janina', 'Blyth', 'jblythy@disqus.com', '914957034');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Goober', 'D''Almeida', 'gdalmeidaz@jiathis.com', '481306878');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Nevins', 'Jendrusch', 'njendrusch10@google.ca', '856397450');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Klemens', 'Tocknell', 'ktocknell11@sphinn.com', '263153753');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Yancy', 'Ashington', 'yashington12@dagondesign.com', '516640771');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Hildegaard', 'Leroux', 'hleroux13@mail.ru', '897301861');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Ulla', 'Kenion', 'ukenion14@ftc.gov', '847156048');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Zondra', 'Iceton', 'ziceton15@freewebs.com', '849636809');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Chanda', 'Simione', 'csimione16@symantec.com', '175125542');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Fred', 'Aburrow', 'faburrow17@123-reg.co.uk', '116360414');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Fitz', 'Hiers', 'fhiers18@wikia.com', '835577001');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Phil', 'Marklund', 'pmarklund19@istockphoto.com', '379640670');

```
insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Corly', 'Brekonridge', 'cbrekonridge1a@census.gov', '496334124');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Dom', 'Kingdon', 'dkingdon1b@ifeng.com', '903170391');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Joby', 'Gilmartin', 'jgilmartin1c@vk.com', '278562893');

insert into KLIENT (IMIE, NAZWISKO, EMAIL, TELEFON) values ('Daveta', 'Saldler', 'dsaldler1d@fda.gov', '161321085');


--PRODUCENT

insert into PRODUCENT(NAZWA_PRODUCENTA) values ('ASUS');

insert into PRODUCENT(NAZWA_PRODUCENTA) values ('AMD');

insert into PRODUCENT(NAZWA_PRODUCENTA) values ('Lenovo');

insert into PRODUCENT(NAZWA_PRODUCENTA) values ('ACER');

insert into PRODUCENT(NAZWA_PRODUCENTA) values ('Sony');

insert into PRODUCENT(NAZWA_PRODUCENTA) values ('Intel');

insert into PRODUCENT(NAZWA_PRODUCENTA) values ('GIGABYTE');

insert into PRODUCENT(NAZWA_PRODUCENTA) values ('HP');

insert into PRODUCENT(NAZWA_PRODUCENTA) values ('NVIDIA');

--PROCESOR

insert into procesor(NAZWA_PROCESORA,ID_PRODUCENT) values ('Core i7 8240 KL',6);

insert into procesor(NAZWA_PROCESORA,ID_PRODUCENT) values ('Core i5 7300 HQ',6);

--RAM

insert into RAM(ILOSC_RAM,ID_PRODUCENT) values (8,7);

insert into RAM(ILOSC_RAM,ID_PRODUCENT) values (16,7);

insert into RAM(ILOSC_RAM,ID_PRODUCENT) values (32,7);

insert into RAM(ILOSC_RAM,ID_PRODUCENT) values (8,6);

insert into RAM(ILOSC_RAM,ID_PRODUCENT) values (16,6);

--TYP

insert into TYP(NAZWA_TYPU) values ('Komputer stacjonarny');

insert into TYP(NAZWA_TYPU) values ('Notebook');

insert into TYP(NAZWA_TYPU) values ('Netbook');
```

```
insert into TYP(NAZWA_TYPU) values ('Chromebook');

--KARTY GRAFICZNE

insert into KARTA_GRAFICZNA(NAZWA_KARTY,ID_PRODUCENT) values ('GEFORCA GTX
1050',9);

insert into KARTA_GRAFICZNA(NAZWA_KARTY,ID_PRODUCENT) values ('GEFORCA GTX 1050
Ti',9);

insert into KARTA_GRAFICZNA(NAZWA_KARTY,ID_PRODUCENT) values ('GEFORCA GTX
2050',9);

--KOMPUTERY

insert into
KOMPUTER(NAZWA,CENA,STAN,ID_TYP,ID_PROCESOR,ID_KARTY,ID_RAM,ID_PRODUCENT)
values ('PREDATOR',4500.00,300,2,2,3,4,2);

insert into
KOMPUTER(NAZWA,CENA,STAN,ID_TYP,ID_PROCESOR,ID_KARTY,ID_RAM,ID_PRODUCENT)
values ('LEGION Y520',3800.00,400,2,1,2,2,4);

insert into
KOMPUTER(NAZWA,CENA,STAN,ID_TYP,ID_PROCESOR,ID_KARTY,ID_RAM,ID_PRODUCENT)
values ('OMEN',2800.00,200,3,1,1,1,2);

--

/

BEGIN

PR_DODAJ_KOMPUTER('INSPIRON',120,2000);

PR_DODAJ_KOMPUTER('CERBERUS',230,2000);

PR_DODAJ_KOMPUTER('THINKPAD',300,2000);

PR_DODAJ_KOMPUTER('IDEAPAD',70,2000);

PR_DODAJ_KOMPUTER('ROG',400,2000);

PR_DODAJ_KOMPUTER('HYPERBOOST',520,2000);

PR_DODAJ_KOMPUTER('ASPIRE',340,2000);

PR_DODAJ_WIELE_FAKTUR(150);

END;

/

COMMIT;
```

4.2. Skrypt deinstalacyjny wraz z usuwanie funkcji, procedur oraz widoków

```
DROP TABLE FAKTURA_NAGLOWEK CASCADE CONSTRAINTS ;
DROP TABLE FAKTURA_POZYCJA CASCADE CONSTRAINTS ;
DROP TABLE KARTA_GRAFICZNA CASCADE CONSTRAINTS ;
DROP TABLE KLIENT CASCADE CONSTRAINTS ;
DROP TABLE KOMPUTER CASCADE CONSTRAINTS ;
DROP TABLE PLATNOSC CASCADE CONSTRAINTS ;
DROP TABLE PROCESOR CASCADE CONSTRAINTS ;
DROP TABLE PRODUCENT CASCADE CONSTRAINTS ;
DROP TABLE RAM CASCADE CONSTRAINTS ;
DROP TABLE TYP CASCADE CONSTRAINTS ;
DROP SEQUENCE SEQ_FAKTURA_NAGLOWEK ;
DROP SEQUENCE SEQ_FAKTURA_POZYCJA ;
DROP SEQUENCE SEQ_KARTA_GRAFICZNA ;
DROP SEQUENCE SEQ_KLIENT ;
DROP SEQUENCE SEQ_KOMPUTER ;
DROP SEQUENCE SEQ_PLATNOSC ;
DROP SEQUENCE SEQ_PROCESOR ;
DROP SEQUENCE SEQ_PRODUCENT ;
DROP SEQUENCE SEQ_RAM ;
DROP SEQUENCE SEQ_TYP ;
DROP FUNCTION FN_LOS_KLIENT;
DROP FUNCTION FN_LOS_PLATNOSC;
DROP FUNCTION FN_LOS_POZYCJE;
DROP FUNCTION FN_LOS_KOMPUTER;
DROP FUNCTION FN_DAJ_STAN;
DROP FUNCTION FN_DAJ_CENE;
DROP FUNCTION FN_LOS_ILOSC;
DROP FUNCTION FN_LOS_TYP;
DROP FUNCTION FN_LOS_PROCESOR;
DROP FUNCTION FN_LOS_KARTA;
DROP FUNCTION FN_LOS_RAM;
```

```
DROP FUNCTION FN_LOS_PRODUCENT;
DROP FUNCTION FN_LOS_DATA;
DROP FUNCTION FN_DAJ_LAST;
DROP PROCEDURE PR_DODAJ_KOMPUTER;
DROP PROCEDURE PR_DODAJ_FAKTURE;
DROP PROCEDURE PR_DODAJ_WIELE_FAKTUR;
DROP VIEW KOMPUTERY_ZYSK;
DROP VIEW OSTATNI_MIESIAC;
DROP VIEW TOP20_KLIENTOW;
COMMIT;
```

5. Instrukcja instalacji projektu

Aby zainstalować poprawnie projekt należy w programie Oracle SQL Developer uruchomić najpierw skrypt deinstalujący, wklejając go do przestrzeni roboczej a następnie kompilując klawiszem F5. Jeżeli nie wystąpiły żadne błędy podczas uruchamiania, to oznacza że uruchamianie skryptu zakończyło się powodzeniem. Następnie tak samo postępujemy ze skryptem instalującym. By sprawdzić czy projekt został poprawnie stworzony, możemy zapisać kilka prostych zapytań.

W skrypcie wdrażającym użyte są procedury tworzące nowe komputery oraz nowe faktury. By wywołać je ręcznie należy skorzystać z poniższej konstrukcji:

```
BEGIN
PR_DODAJ_WIELE_FAKTUR(5);
END;
```

6. Raporty

6.1. Zestawienie komputerów i zysków z ich sprzedaży. Zrealizowane zostało za pomocą wyżej zadeklarowanego widoku.

Ilość sprzedanych komputerów

Komputer	Procesor	Karta Graficzna	RAM	Ilosc	Zysk_calkowity
AMD PREDATOR	Core i5 7300 HQ	GEFORCA GTX 2050	8 GB	91	409500
AMD OMEN	Core i7 8240 KL	GEFORCA GTX 1050	8 GB	107	299600
ACER LEGION	Core i7 8240 KL	GEFORCA GTX 1050 Ti	16 GB	72	273600
GIGABYTE	Core i5 7300 HQ	GEFORCA GTX 2050	16 GB	90	46800
GIGABYTE ROG	Core i7 8240 KL	GEFORCA GTX 1050 Ti	16 GB	115	46000
ACER ASPIRE	Core i5 7300 HQ	GEFORCA GTX 2050	8 GB	98	33320
AMD THINKPAD	Core i7 8240 KL	GEFORCA GTX 1050	8 GB	91	27300
ASUS CERBERUS	Core i7 8240 KL	GEFORCA GTX 1050 Ti	16 GB	89	20470
HP INSPIRON	Core i5 7300 HQ	GEFORCA GTX 2050	32 GB	98	11760
AMD IDEAPAD	Core i7 8240 KL	GEFORCA GTX 2050	8 GB	94	6580

6.2. Zestawienie 20 pierwszych klientów posiadających największe wydatki. Zrealizowane zostało za pomocą wyżej zadeklarowanego widoku.

Top 20 klientów

Imie	Nazwisko	Ilosc zamowien	Wydatki
Audrye	Peiro	2	42640
Arabela	Sondland	2	33710
Ulla	Kenion	1	23600
Joby	Gilmartin	1	22030
Fern	Gashion	2	20890
Blancha	Forstall	2	20520
Herta	Easen	2	16430
Corly	Brekonridge	1	11760
Lorna	Dewdeny	1	11380
Xaviera	Franzel	1	8400
Rand	Deakins	1	4600
Basilius	Prine	1	1950
Zondra	Iceton	1	1790
Yancy	Ashington	1	1500
Chrotoem	Peacock	1	760