

**Instytut Informatyki i Automatyki**

**Kierunek Informatyka**

**Projekt zaliczeniowy 2**

**z Projektowania baz danych**

***Temat:*** PODPROGRAMY.

**Prowadzący:** Dr inż. Eugenia Busłowska

Wykonujący projekt: 8096, Budzyński Karol

7455, Zalewski Łukasz

*Nr indeksu, Nazwisko imię*

Studia niestacjonarne I stopnia

Kierunek: **Informatyka**

Semestr: IV, grupa 1

...........................

Liczba Punktów

Data: 19.05.2017r.

**1. Utworzyć dwa bloki anonimowe do własnych tabel.**

Blok anonimowy podający najniższą, najwyższą i średnią cenę abonamentu:

SET SERVEROUTPUT ON

DECLARE

A\_Cena\_avg PS1N\_BZ.abonament.Cena%TYPE;

A\_Cena\_min PS1N\_BZ.abonament.Cena%TYPE;

A\_Cena\_max PS1N\_BZ.abonament.Cena%TYPE;

BEGIN

select avg(Cena) into A\_Cena\_avg

from PS1N\_BZ.abonament A;

select min(Cena) into A\_Cena\_min

from PS1N\_BZ.abonament A;

select max(Cena) into A\_Cena\_max

from PS1N\_BZ.abonament A;

DBMS\_OUTPUT.PUT\_LINE('Cena najtanszego abonamentu: '||ROUND(A\_Cena\_min,2)||'zl');

DBMS\_OUTPUT.PUT\_LINE('Cena najdrozszego abonamentu: '||ROUND(A\_Cena\_max,2)||'zl');

DBMS\_OUTPUT.PUT\_LINE('Srednia cena abonamentu: '||ROUND(A\_Cena\_avg,2)||'zl');

END;

Blok anonimowy podający dane klienta który dokonał najwyższej wpłaty oraz kwotę wpłaty:

SET SERVEROUTPUT ON

DECLARE

nazwa PS1N\_BZ.klienci.nazwa\_klient%TYPE;

imie PS1N\_BZ.klienci.imie\_klient%TYPE;

nazwisko PS1N\_BZ.klienci.nazwisko\_klient%TYPE;

kwota PS1N\_BZ.wplaty.kwota\_wplaty%TYPE;

BEGIN

select max(kwota\_wplaty) into kwota from PS1N\_BZ.wplaty;

select k.nazwa\_klient, k.imie\_klient, k.nazwisko\_klient into nazwa, imie, nazwisko

from PS1N\_BZ.klienci k, PS1N\_BZ.wplaty w

where k.id\_klient = w.id\_klient

and w.kwota\_wplaty = (select max(kwota\_wplaty) from PS1N\_BZ.wplaty);

DBMS\_OUTPUT.PUT\_LINE('Najwyzszej wplaty dokonal klient: '||nazwa||' '||imie||' '||nazwisko);

DBMS\_OUTPUT.PUT\_LINE('Kwota wplaty: '||kwota||'zl');

END;

**2. Utworzyć procedurę do wprowadzania danych do wybranej tabeli oraz procedurę bez parametrów. Podać przykłady wywołań.**

Procedura do wprowadzania nowych abonamentów i przykłady wywołań:

SET SERVEROUTPUT ON

CREATE OR REPLACE procedure dodaj\_abonament

(Nazwa IN PS1N\_BZ.abonament.nazwa\_abonament%TYPE,

DL IN PS1N\_BZ.abonament.predkosc\_dl%TYPE,

UL IN PS1N\_BZ.abonament.predkosc\_ul%TYPE,

cena IN PS1N\_BZ.abonament.cena%TYPE)

IS

id\_abo PS1N\_BZ.abonament.id\_abonament%TYPE;

BEGIN

SELECT NVL(Max(PS1N\_BZ.abonament.id\_abonament)+1,1) INTO id\_abo

FROM PS1N\_BZ.abonament;

INSERT INTO PS1N\_BZ.abonament VALUES(id\_abo, Nazwa, DL, UL, cena);

COMMIT;

END;

execute dodaj\_abonament('Abo L', '10Mbit', '1Mbit', 40);

execute dodaj\_abonament('Abo XL', '30Mbit', '3Mbit', 50);

execute dodaj\_abonament('Abo XXL', '60Mbit', '6Mbit', 60);

Procedura bez parametrów podająca ilość klientów mających abonament „Lekki”:

SET SERVEROUTPUT ON

create or replace procedure ilosc\_klientow

as

ile\_kli number(5);

begin

select count(1) into ile\_kli

from PS1N\_BZ.klienci k, PS1N\_BZ.umowy u, PS1N\_BZ.abonament a

where k.id\_klient = u.id\_klient

and u.id\_abonament = a.id\_abonament

and a.nazwa\_abonament = 'Lekki';

DBMS\_OUTPUT.PUT\_LINE('Abonament Lekki posiada '||ile\_kli||' klientow.');

end;

execute ilosc\_klientow;

Abonament Lekki posiada 2 klientow.

**3. Utworzyć dwie funkcje operujące na danych wybranych własnych tabel. Jedną z funkcji wywołać w zapytaniu wybierającym.**

Funkcja zwracająca liczbę wpłat dokonanych przez klienta o id podanym jako parametr:

create or replace function ile\_wplat (id\_klient in number)

return number

as

ile\_wplat number(5);

begin

select count(1) into ile\_wplat

from PS1N\_BZ.wplaty w, PS1N\_BZ.klienci k

where k.id\_klient = id\_wlas and k.id\_klient = w.id\_klient;

return ile\_wplat;

end;

SET SERVEROUTPUT ON

declare

ile\_wplat2 number(5);

id\_klient2 number(5) := &wp;

begin

ile\_wplat2 := ile\_wplat(id\_klient2);

DBMS\_OUTPUT.PUT\_LINE('Klient o ID: '||id\_klient2||' dokonal: '||ile\_wplat2||' wplaty');

end;

Funkcja zwracająca ID pracownika przypisanego do zlecenia p ID podanym jako parametr:

create or replace function kto\_robi (id\_zlec in number)

return number

as

id\_prac number(5);

begin

select z.id\_pracownik into id\_prac

from PS1N\_BZ.zlecenia z

where z.id\_zlecenia = id\_zlec;

return id\_prac;

end;

SET SERVEROUTPUT ON

declare

imie PS1N\_BZ.pracownicy.imie\_pracownik%type;

nazwisko PS1N\_BZ.pracownicy.nazwisko\_pracownik%type;

id\_zlec2 number(5) := &wp;

id\_prac2 number(5);

begin

select (IMIE\_PRACOWNIK) into imie

from PS1N\_BZ.pracownicy p, PS1N\_BZ.zlecenia z

where z.id\_zlecenia = id\_zlec2 and z.ID\_PRACOWNIK = p.ID\_PRACOWNIK;

select (nazwisko\_pracownik) into nazwisko

from PS1N\_BZ.pracownicy p, PS1N\_BZ.zlecenia z

where z.id\_zlecenia = id\_zlec2 and z.ID\_PRACOWNIK = p.ID\_PRACOWNIK;

id\_prac2 := kto\_robi(id\_zlec2);

DBMS\_OUTPUT.PUT\_LINE('Pracownik '||imie||' '||nazwisko||' o numerze ID: '||id\_prac2||' jest przypisany do zlecenia o numerze ID: '||id\_zlec2||'.');

end;

**4. Utworzyć sekwencję cykliczną do generowania wartości malejących o jeden. Sprawdzić poprawność działania sekwencji w procedurze, w poleceniu insert podczas wprowadzania wartości do własnej tabeli.**

create sequence seq\_malejeaco

minvalue 1

maxvalue 1000

start with 1000

increment by -1

cache 20;

SET SERVEROUTPUT ON

CREATE OR REPLACE procedure dodaj\_abonament2

(Nazwa IN PS1N\_BZ.abonament.nazwa\_abonament%TYPE,

DL IN PS1N\_BZ.abonament.predkosc\_dl%TYPE,

UL IN PS1N\_BZ.abonament.predkosc\_ul%TYPE,

cena IN PS1N\_BZ.abonament.cena%TYPE)

IS

id\_abo PS1N\_BZ.abonament.id\_abonament%TYPE;

BEGIN

select seq\_malejeaco.nextval into id\_abo from dual;

INSERT INTO PS1N\_BZ.abonament VALUES(id\_abo, Nazwa, DL, UL, cena);

COMMIT;

END;

execute dodaj\_abonament2('Abo CCCC', '600Mbit', '60Mbit', 610);

**5. Utworzyć następujące typy kursorów bazujących na własnych tabelach:**

* **Kursor klasyczny z dwoma parametrami.**

Kursor zwracający imię i nazwisko pracownika o numerze id i stanowisku podanych jako parametry:

DECLARE CURSOR c\_prac (ident NUMBER,

st VARCHAR2)

IS

select nazwisko\_pracownik, imie\_pracownik from PS1N\_BZ.pracownicy

where PS1N\_BZ.pracownicy.id\_pracownik = ident and PS1N\_BZ.pracownicy.stanowisko\_pracownik = st;

v\_imie PS1N\_BZ.pracownicy.imie\_pracownik%type;

v\_nazwisko PS1N\_BZ.pracownicy.nazwisko\_pracownik%type;

begin

open c\_prac(1, 'Wlasciciel');

loop

fetch c\_prac into v\_nazwisko, v\_imie;

exit when (c\_prac%notfound);

DBMS\_OUTPUT.PUT\_LINE(v\_nazwisko||' '||v\_imie);

end loop;

close c\_prac;

end;

* **Kursor w pętli FOR.**

Kursor podający abonamenty których cena jest większa od 100zł:

SET SERVEROUTPUT ON

declare cursor c\_abo

is

select a.id\_abonament id\_ab, a.nazwa\_abonament na\_ab, a.cena cen\_ab from PS1N\_BZ.abonament a

where a.cena>100;

begin

for ks in c\_abo

loop

DBMS\_OUTPUT.PUT\_LINE(c\_abo%rowcount||') ID abonamentu: '||ks.id\_ab||', Nazwa abonamentu: '||ks.na\_ab||', Cena abonametnu: '||ks.cen\_ab||'zl');

end loop;

end;

* **Anonimową pętlę FOR kursora.**

Kursora podający nazwiska i imiona klientów, którzy składali zlecenia w 2016 roku:

begin

for ks in (select nazwisko\_klient, imie\_klient from PS1N\_BZ.klienci, PS1N\_BZ.zlecenia

where PS1N\_BZ.klienci.id\_klient = PS1N\_BZ.zlecenia.id\_klient and PS1N\_BZ.zlecenia.data\_zlecenia > '15/12/31' and PS1N\_BZ.zlecenia.data\_zlecenia < '17/01/01')

loop

DBMS\_OUTPUT.PUT\_LINE(ks.nazwisko\_klient||' '||ks.imie\_klient);

end loop;

end;

* **Kursor SELECT FOR UPDATE.**

Kursor SELECT FOR UPDATE modyfikujący nazwiska pracowników na pisane wielkimi literami:

SET SERVEROUTPUT ON

declare cursor c\_wielkie

is

select \* from PS1N\_BZ.pracownicy for update of nazwisko\_pracownik;

v\_nazw PS1N\_BZ.pracownicy.nazwisko\_pracownik%type;

begin

for ks in c\_wielkie

loop

v\_nazw := upper(ks.nazwisko\_pracownik);

update PS1N\_BZ.pracownicy set nazwisko\_pracownik = v\_nazw

where current of c\_wielkie;

end loop;

commit;

end;

**6. Pakiet z wcześniej utworzoną: procedurą, funkcją i kursorem.**

create or replace package pack\_proj\_6 as

procedure ilosc\_klientow ;

function ile\_wplat (id\_klient in number) return number;

cursor c\_abo;

end pack\_proj\_6;

create or replace package body pack\_proj\_6 as

procedure ilosc\_klientow

as

ile\_kli number(5);

begin

select count(1) into ile\_kli

from PS1N\_BZ.klienci k, PS1N\_BZ.umowy u, PS1N\_BZ.abonament a

where k.id\_klient = u.id\_klient

and u.id\_abonament = a.id\_abonament

and a.nazwa\_abonament = 'Lekki';

DBMS\_OUTPUT.PUT\_LINE('Abonament Lekki posiada '||ile\_kli||' klientow.');

end;

function ile\_wplat (id\_klient in number) return number

as

ile\_wplat number(5);

begin

select count(1) into ile\_wplat

from PS1N\_BZ.wplaty w, PS1N\_BZ.klienci k

where k.id\_klient = id\_wlas and k.id\_klient = w.id\_klient;

return ile\_wplat;

end;

cursor c\_abo

is

select a.id\_abonament id\_ab, a.nazwa\_abonament na\_ab, a.cena cen\_ab from PS1N\_BZ.abonament a

where a.cena>100;

begin

for ks in c\_abo

loop

DBMS\_OUTPUT.PUT\_LINE(c\_abo%rowcount||') ID abonamentu: '||ks.id\_ab||', Nazwa abonamentu: '||ks.na\_ab||', Cena abonametnu: '||ks.cen\_ab||'zl');

end loop;

end;

end pack\_proj\_6;

**7. Obsłużyć dwa własne wyjątki w jednych z podprogramów.**

Wyjątek niepozwalający wprowadzać ujemnej kwoty wpłaty, oraz daty większej niż bieżąca do tabeli WPŁATY:

SET SERVEROUTPUT ON

declare

wyjatek1 exception;

wyjatek2 exception;

wplata PS1N\_BZ.wplaty.kwota\_wplaty%type;

data\_wp PS1N\_BZ.wplaty.data\_wplaty%type;

data\_cur PS1N\_BZ.wplaty.data\_wplaty%type;

begin

wplata := &wp;

if wplata < 0 then raise wyjatek1;

end if;

select current\_date into data\_cur from dual;

data\_wp := &dw;

if data\_wp > data\_cur then raise wyjatek2;

end if;

insert into PS1N\_BZ.wplaty values (100, 'abonament 05.2017r.', wplata, data\_wp, 1);

DBMS\_OUTPUT.PUT\_LINE('Wplata dodana do bazy.');

exception

when wyjatek1 then

DBMS\_OUTPUT.PUT\_LINE('!!! Kwota wplaty nie moze byc mniejsza od 0 !!!');

when wyjatek2 then

DBMS\_OUTPUT.PUT\_LINE('!!! Data wplaty nie moze byc wieksza od '||data\_cur||' !!!');

end;