1.Zadanie Pierwsze

Procedura - dzien narodzin //nie wiem co to maska więc zadanie nie pełne

```
CREATE OR REPLACE PROCEDURE DZIEN_URODZIN
(
  DATAA IN VARCHAR2
, TYP IN NUMBER
, DZIEN OUT VARCHAR2
) AS
mojWyjatek EXCEPTION;
BEGIN
  iF DATAA IS NULL THEN
    RAISE mojWyjatek;
  ELSE
    select to_char(to_date(DATAA, 'YYYY-MM-DD'),'FmDay') INTO DZIEN from dual;
  END IF;
EXCEPTION
WHEN OTHERS THEN DBMS_OUTPUT.PUT_LINE('Bledne dane wejsciowe');

END DZIEN_URODZIN;
```

Wywołanie procedury

```
DECLARE
   DZIEN_TEN  VARCHAR2(20);
   DATAA_TEN VARCHAR2(20);
   TYP_TEN INT;
BEGIN
TYP_TEN  :=5;
--DATAA_TEN:='1997-11-20';
DATAA_TEN:='2019-01-22';
  DZIEN_URODZIN(
    DATAA=>DATAA_TEN,
    TYP => TYP_TEN,
    DZIEN => DZIEN_TEN
  );
DBMS_OUTPUT.PUT_LINE('DZIEN NARODZIN: ' || DZIEN_TEN);
END;
```

2.Zadanie Drugie

a) Skopiowanie dept i emp

b) utowrzenie sekwencji(cykliczna od 90 do 55)

c) klucze podstawowe dla tabel

d) wyzwalacz który automatycznie numeruję pole EMPNO w tabeli EMP

e) w EMP do HIREDATE przypisz DEFAULT w celu wprowadzenia aktualnej daty

f) w EMP, kolumna COMM ma miec wartośc 100<nasz_wartosc<1000 lub null

g) utwórz odpowiednie więzy integralności referencyjnej

h) pokazac, że to działa wstawiając odpowiednie rekordy

```
  --  A -kopiowanie
  CREATE TABLE emp AS select * from SCOTT.EMP;
  CREATE TABLE dept AS select * from SCOTT.DEPT;
  -- B  -sekwencja
  CREATE SEQUENCE SEQ1          --currval-zwraca aktualna wartośc sekwecnji
         MINVALUE 55            --nextval-bierze następną wartośc
         MAXVALUE 90
         START WITH 90
         INCREMENT BY -1
         CYCLE;--przekręca się, może też by NOCYCLE
  -- C -klucze podstawowe
  alter table emp add CONSTRAINT klucz_podstawowy_emp PRIMARY KEY (EMPNO);
  alter table dept add CONSTRAINT klucz_podstawowy_dept PRIMARY KEY (DEPTNO);
  -- D -wyzwalacz
CREATE OR REPLACE TRIGGER automatyczna_numeracja BEFORE INSERT ON EMP_J
FOR EACH ROW
declare tym NUMBER;
begin
SELECT MAX(EMPNO) into tym FROM EMP_J;
:NEW.EMPNO := tym+1;
end;
--jakąś dowolną liczbę trzeba niestety nadal wstwaiac na początek...
INSERT INTO EMP_J(EMPNO,ENAME,JOB ,MGR ,HIREDATE ,SAL,COMM,DEPTNO )
VALUES (1,'Mateusz','CLERK',7902,to_date('17-12-1980','dd-mm-yyyy'),800,NULL,20);
-- E - DEFAULT VALUE
ALTER TABLE EMP_J
MODIFY (HIREDATE DEFAULT SYSDATE );
-- F  -ograniczenie na coom 100 do 1000 lub null
ALTER TABLE EMP_J
ADD CONSTRAINT wartosc_comm CHECK ((COMM  BETWEEN 100 and 1000) OR (COMM IS
NULL))ENABLE NOVALIDATE;
-- G - wiezy integralnosci
---a C****J z NIMI... z grubsza dziala
```

Zadanie 3

Funkcja

```
CREATE OR REPLACE FUNCTION TEST_ROK
(
  ROK IN NUMBER
, CZY_PRZESTEPNY OUT VARCHAR2
) RETURN VARCHAR2 AS
mojWyjatek EXCEPTION;
BEGIN
  if ROK<0 OR ROK IS NULL then
        RAISE mojWyjatek;
  else
      if (MOD(rok,400) =0) OR (MOD(rok,4)=0 AND MOD(rok,100)<>0)   then
      return 'true';
      else
      return 'false';
      end if;
  end if;
EXCEPTION
WHEN mojWyjatek THEN
    DBMS_OUTPUT.PUT_LINE ('ARGUMNET MUSI BYC wiekszy od 0 i byc liczba czlkiwta');
    RETURN null;
END TEST_ROK;
```

wywołanie funkcji

```
SElect TEST_ROK(2001) from dual;
```

Zadanie 5(4 nie ma jak coś)

dodatkowa kolumna UNIQE plus trigger tworzący maila

```
--dodawanie kolumny
ALTER TABLE EMP_J
ADD (E_MAIL VARCHAR2(20) UNiQUE);
--trigger --nawet działa dobrze
CREATE OR REPLACE TRIGGER tworenia_maila BEFORE INSERT ON EMP_J
FOR EACH ROW
begin
:NEW.e_mail:=CONCAT(REPLACE(LOWER(:NEW.ENAME),' ','_'),'@test.pl');
end;
```

XXXXX - maybe will be useful

```
select * from SCOTT.DEPT;

create table dept as select * from scott.dept;

select * from scott.emp;
create table emp as select * from scott.emp;

select * from emp;
select * from dept;

--1. wyświetlić wszystkich pracowników posortowanych po nazwisku
select * from emp order by ename asc;
--2. wyświetlić uporzadkowane po nazwisku 3 pierwsze krotki
select * from (select * from emp order by ename asc) XXX where ROWNUM <= 3;
--3. wyświetlić pierwsze 3 krotki z emp
select * from emp where ROWNUM <= 3;
--4. wybierz nazwiska pracowników i ustal ich roczne zarobki
--5. ustal roczne zarobki pracowników po podwyższeniu pensi o $250
--6. wybierz z emp wszystkie wzajemnie różne kombinacje nr departamentu i
stanowiska pracy
--7. wybierz anzwiska i pensje wszystkich pracowników, których nazwiska zaczynają
się na Ssa
--8. wybierz nazwiska i wartości zarobków pracowników wącznie z obniżeniem
prowizji
--9. wybierz pracowników, którzy mają wyższą prowizję niż pensję


-----------------------------------------------------

select * from emp;
--1.oblicz średni zarobek w firmie
select avg(sal) from emp;
--2.wybierz stanowiska pracy i maksymalne zarobki na tych stanowiskach
select job, max(sal) from emp group by job;
--3.oblicz minimalny zarobek w każdym departamencie z podzialem na stanowiska
select deptno, job, min(sal) from emp group by deptno, job;
--4.wybrać średnie zarobki dla grup zawodowych gdzie maksymalne zarobki są wyższe
niż 2000
select job, avg(sal) from emp group by job having max(sal) > 2000;
--5.wybrać pracowników, którzy zarabiają mniej od swoich kierowników
select e.*, m.ename, m.sal from emp e join emp m on e.mgr=m.empno where e.sal <
m.sal;
--6.przetestować dziannie union i union all
select job from emp union select ename from emp;
select job from emp union all select ename from emp;
--7.przetestować intersect
--8.operator minus
--9.znaleźć pracowników, których pensja jest na liście najwyższych zarobków w
departamentach
select * from emp where sal in (select distinct max(sal) from emp group by
deptno);
--9b.osoba ma miec zarobek pokrywajacy sie z maksymalnym zarobkiem z departamentu
z ktorego pochodzi
```

```
select * from emp, (select max(sal) ms, deptno from emp group by deptno) d where
sal=ms and emp.deptno=d.deptno;

--10.wykorzystujac all lub any zrealizowac zapytanie: znalezc wszystkich
pracownikow ktorzy zarabiaja wiecej niz ktokolwiek w departamencie 30
select * from emp where sal > all(select distinct sal from emp where deptno=30);
--11.wybrac stanowisko na ktorym sa najnizsze srednie zarobki
select job from (select job, avg(sal) ssal from emp group by job order by ssal)
where ROWNUM=1;
--12.zapytaniem skorelowanym zrealizowac: znalezc osoby, ktore zarabiaja mniej niz
wynosi srednia w ich zawodach
select * from emp e where e.sal < (select avg(sal) from emp s where e.job=s.job
group by job);
--13.za pomoca operatora exist znalezc pracownikow ktorzy maja podwladnuych
select * from emp e where exists (select * from emp p where e.empno=p.mgr);
--14.znalezc departament w ktorym nikt nie pracuje
select * from dept where not exists(select * from emp where
emp.deptno=dept.deptno);


-------------------------------------------------------

select to_char(sysdate, 'YYYY-MM-DD') from dual;

--1.policzyć w dniach ile dni uplynego od dnia urodzin
select sysdate - to_date('17.05.1993', 'DD-MM-YYYY') from dual;
--2.-||- uzyc round
select round(sysdate) - to_date('17.05.1993', 'DD-MM-YYYY') from dual;
--3.-||- uzyc trucate
select trunc(sysdate) - to_date('17.05.1993', 'DD-MM-YYYY') from dual;
--4.dzien tygodnia urodzin
select to_char(to_date('17.05.1993', 'DD-MM-YYYY'), 'DAY') from dual;


---------

select * from emp;

--1.zadeklarowac 3 zmienne czesci deklaracyjnej, a w begin-end przelac do tych
zmiennych dla danego pracownika wartosci

declare
  nazwisko varchar(20);
  pensja number(5);
  data_zatr date;
begin
  select ename, sal, hiredate into nazwisko, pensja, data_zatr from emp where
empno=7369;
  SYS.DBMS_OUTPUT.PUT_LINE(nazwisko || ' ' || pensja || ' ' || to_char(data_zatr,
'YYYY-MM-DD'));
end;


--2.
```

```
declare
  prac emp%rowtype;
begin
  select * into prac from emp where empno=7369;
  SYS.DBMS_OUTPUT.PUT_LINE(prac.ename || ' ' || prac.sal);
end;




---------------------------

select * from emp;

declare
  pensja emp.sal%type;
  x int;
begin
  x := 500;
  select sal into pensja from emp where empno=7369;
  if pensja > x then
    dbms_output.put_line('p');
  elsif pensja < x then
    dbms_output.put_line('x');
  else
    dbms_output.put_line('=');
  end if;
end;

-------------------------------------

declare
  x int := 0;
begin
  loop
    if x > 10 then
      exit;
    end if;
    dbms_output.put_line(x);
    x := x + 1;
  end loop;
end;

-----------------------------------------

begin
  for i in 0..10 loop
    dbms_output.put_line(i);
  end loop;
end;

-----------------------------------------

declare
```

```
  x int := 0;
begin
  <<hehe>>
  dbms_output.put_line(x);
  if x < 10 then
    x := x + 1;
    goto hehe;
  end if;
end;

----------------------------------------

select deptno, dname, count(empno) as liczba from dept left join emp using(deptno)
group by deptno, dname;

declare
  liczba int;
  brak_pracownikow exception;
  za_duzo_pracownikow exception;
begin
  select count(empno) into liczba from dept left join emp using(deptno) group by
deptno having deptno=15;
  if liczba < 1 then
    raise brak_pracownikow;
  end if;
  if liczba > 10 then
    raise za_duzo_pracownikow;
  end if;
exception
  when brak_pracownikow then
    dbms_output.put_line('Brak pracownikow');
  when za_duzo_pracownikow then
    dbms_output.put_line('Za duzo pracownikow');
end;

--------------------------------

--opracować przykladowy blok z dwiema zmiennymi: uzytkownik i haslo. blok ma
sprawdzic czy uzytkownik i haslo jest inne, jesli tak to sprawdzi czy haslo nie
jest trywialne (nie zawiera slow oracle, baza), jesli nie zawiera to sprawdzi czy
minimalna dlugosc hasla jest wieksza od 3 ale mniejsza od 6 jesli tak to sprawdzi
czy w hasle jest cyfra, jesli tak to wyswietli OK

declare
  uzytkownik varchar(20) := 'Abcd';
  haslo varchar(20) := 'xxxxx';
  cyfra boolean := false;
  haslo_jak_uzytkownik exception;
  proste_haslo exception;
  krotkie_haslo exception;
  dlugie_haslo exception;
  brak_cyfry exception;
begin
  uzytkownik := lower(uzytkownik);
```

```
  haslo := lower(haslo);
  if haslo = uzytkownik then
    raise haslo_jak_uzytkownik;
  end if;
  if haslo in ('oracle', 'baza') then
    raise proste_haslo;
  end if;
  if length(haslo) < 3 then
    raise krotkie_haslo;
  end if;
  if length(haslo) > 6 then
    raise dlugie_haslo;
  end if;
  for i in 0..9
  loop
    if instr(haslo, to_char(i)) > 0 then
      cyfra := true;
    end if;
  end loop;
  dbms_output.put_line('OK');
exception
  when haslo_jak_uzytkownik then
    dbms_output.put_line('Haslo musi byc inne niz nazwa uzytkownika.');
  when proste_haslo then
    dbms_output.put_line('Haslo nie moze zawierac "oracle" ani "baza".');
  when krotkie_haslo then
    dbms_output.put_line('Za krotkie haslo.');
  when dlugie_haslo then
    dbms_output.put_line('Za dlugie haslo.');
  when brak_cyfry then
    dbms_output.put_line('Haslo musi zawierac cyfre.');
end;

---

declare
  uzytkownik varchar(20) := 'Abcd';
  haslo varchar(20) := 'abc12';
  cyfra boolean := false;
  haslo_jak_uzytkownik exception;
  proste_haslo exception;
  krotkie_haslo exception;
  dlugie_haslo exception;
  brak_cyfry exception;
begin
  uzytkownik := lower(uzytkownik);
  haslo := lower(haslo);
  if haslo = uzytkownik then
    raise haslo_jak_uzytkownik;
  end if;
  if haslo in ('oracle', 'baza') then
    raise proste_haslo;
  end if;
  if length(haslo) < 3 then
```

```
    raise krotkie_haslo;
  end if;
  if length(haslo) > 6 then
    raise dlugie_haslo;
  end if;
  for i in 0..9
  loop
    if instr(haslo, to_char(i)) > 0 then
      cyfra := true;
    end if;
  end loop;
  if not cyfra then
    raise brak_cyfry;
  end if;
  dbms_output.put_line('OK');
exception
  when haslo_jak_uzytkownik then
    dbms_output.put_line('Haslo musi byc inne niz nazwa uzytkownika.');
  when proste_haslo then
    dbms_output.put_line('Haslo nie moze zawierac "oracle" ani "baza".');
  when krotkie_haslo then
    dbms_output.put_line('Za krotkie haslo.');
  when dlugie_haslo then
    dbms_output.put_line('Za dlugie haslo.');
  when brak_cyfry then
    dbms_output.put_line('Haslo musi zawierac cyfre.');
end;

--------------------------------------------------------------------------------

CREATE SEQUENCE seq2 INCREMENT BY -1 START WITH 10 MAXVALUE 10 MINVALUE 1 CYCLE
NOCACHE;

select seq2.nextval from dual;

----------------------------------------

create table czytelnicy (
  idCzytelnika int primary key,
  nazwisko varchar(20),
  imie varchar(20),
  dataUrodz date,
  miasto varchar(15));

create table publikacje (
  idPublikacji int primary key,
  tytul varchar(30));

create table rejestr (
  idCzytelnika int not null,
  idPublikacji int not null,
  dataWyp date,
  dataZwrotu date,
  constraint fkCzyt foreign key (idCzytelnika) references
```

```
czytelnicy(idCzytelnika),
  constraint fkPub foreign key (idPublikacji) references publikacje(idPublikacji),
  constraint pkRej primary key (idCzytelnika, idPublikacji, dataWyp));

insert into czytelnicy values (seq2.nextval, 'Kucharski', 'Karol', '1993-05-17',
'Warszawa2');
alter table czytelnicy drop constraint miastoCzytelnika;
alter table czytelnicy add constraint miastoCzytelnika check(miasto='Lodz' or
miasto='Piotrkow') enable novalidate;
alter table czytelnicy disable constraint miastoCzytelnika;
alter table czytelnicy modify constraint miastoCzytelnika enable novalidate;

insert into publikacje values(seq2.nextval, 'Aaa');
insert into publikacje values(seq2.nextval, 'Bbbbb');

insert into rejestr values(3, 10, sysdate, null);
insert into rejestr values(5, 9, sysdate, null);
insert into rejestr values(3, 8, sysdate, null);

create or replace view czytelnikKsiazka as
select nazwisko, imie, tytul, to_char(dataWyp) dataWyp from czytelnicy natural
join rejestr natural join publikacje;

select * from publikacje natural full join rejestr natural full join czytelnicy;


--------------------------------------------------------------------------------
-------------------------------------

create or replace view czytLodz1 as select * from czytelnicy where upper(miasto) =
'LODZ';
create or replace view czytLodz2 as select * from czytelnicy where upper(miasto) =
'LODZ' with check option;

insert into czytLodz1 values (seq2.nextval, 'Aaaa', 'Bbb', '1993-05-17',
'Piotrkow');
insert into czytLodz2 values (seq2.nextval, 'Aaaa', 'Bbb', '1993-05-17',
'Piotrkow');


--nie da sie zapisac osob lat<10 lub lat>100
--alter table czytelnicy add constraint wiek check (months_between(sysdate,
dataurodz) >= 120) novalidate; --nie dziala

create table zdarzenie (d date, txt varchar(100));

create or replace trigger usuwanie_prac before delete on emp for each row
begin
  insert into zdarzenie values(sysdate, 'skasowano wiersze z emp');
end;

delete from emp;

rollback;
```

```
create or replace trigger zmiana_pensji after update of sal on emp for each row
begin
  insert into zdarzenie values(sysdate, 'zmiana pensji pracownika ' || :old.empno
|| ' z ' || :old.sal || ' na ' || :new.sal);
end;

update emp set sal=sal+100;

select * from zdarzenie;

--

create or replace trigger zmiana_pensji before insert or update of sal on emp for
each row
begin
  if inserting then
    insert into zdarzenie values(sysdate, 'dodanie pracownika ' || :new.empno || '
z pensja ' || :new.sal);
  end if;
  if updating then
    if (:old.job != 'PRESIDENT') and (:new.sal > (:old.sal * 1.1)) then
      :new.sal := :old.sal * 1.1;
    end if;
    insert into zdarzenie values(sysdate, 'zmiana pensji pracownika ' ||
:old.empno || ' z ' || :old.sal || ' na ' || :new.sal);
  end if;
end;

update emp set sal=sal+10000;

rollback;

-------------------------------------------------------------------------

create or replace trigger logowanie
after logon on schema
begin
  insert into zdarzenie values(sysdate, 'logowanie');
end;

------------------------

create or replace trigger tworzenie
after create on schema
declare
  ipad varchar(20);
begin
  select sys_context('userenv', 'ip_address') into ipad from dual;
  insert into zdarzenie values(sysdate, 'uzytkownik ' || ora_login_user || ' z IP
' || ipad || ' utworzyl obiekt typu ' || ora_dict_obj_type);
end;

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

```
create or replace procedure aktualizacja(oile float, komu int)
is
begin
  update emp set sal = sal + oile where empno = komu;
end;

call aktualizacja(1000, 7369);

------------------------

create or replace procedure ile_publikacji(kto int, ile out int)
is
 i int;
begin
  select count(idpublikacji) into i from (select * from rejestr where idczytelnika
= kto group by idpublikacji);
  ile := i;
end;

set serveroutput on;
declare
  ile int;
begin
  ile_publikacji(3, ile);
  dbms_output.put_line(ile);
end;

-------------------------

-- funkcja z 2 par wej ktora ma obsl bledow i ktora zwroci pole trojkata
prostokatnego

create or replace function pole_trojkata(a float, b float) return float
is
  ujemna_dlugosc exception;
begin
  if a < 0 or b < 0 then
    raise ujemna_dlugosc;
  end if;
  return a * b / 2;
exception
  when ujemna_dlugosc then
    dbms_output.put_line('ujemny bok');
    return null;
end;

select pole_trojkata(3, -5) from dual;

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