Zadania do realizacji

1. Utwórz tabelę pracownicy(id_pracownik, imie, nazwisko, plec, data_urodzenia, data_zatrudnienia, stanowisko, pensja, dodatek) i dodaj do niej co najmniej 10 rekordów, np:
   INSERT INTO pracownicy VALUES (1,'Anna','Kowalska','K','1975-05-25','2001-12-01', 'kierownik',2300,500);
   Usuń z tabeli pracownicy co drugi rekord względem nazwisk i imion ułożonych w porządku alfabetycznym (czyli usuwamy 2, 4, 6, itd. rekord w narzuconym porządku). Użyj kursora jawnego i pętli LOOP. Posłuż się poleceniem FETCH.

2. Wykorzystaj tabelę pracownicy z zadania 1. Zdefiniuj kursor jawny zawierający imiona, nazwiska i daty zatrudnienia wszystkich kierowników. Posłuż się kursorem do wyświetlenia rekordów w formie zdań: "Pracownik <imie> <nazwisko> pracuje na stanowisku kierownika od <data_zatrudnienia>.". Użyj pętli WHILE i posłuż się poleceniem FETCH.

3. Wykorzystaj tabelę pracownicy z zadania 1. Zdefiniuj kursor jawny, dzięki któremu będzie można wyświetlić na ekranie trzech najlepiej zarabiających pracowników patrząc na wysokość zarobków (bez dodatku). Uwzględnij miejsca ex aequo. Użyj pętli FOR.

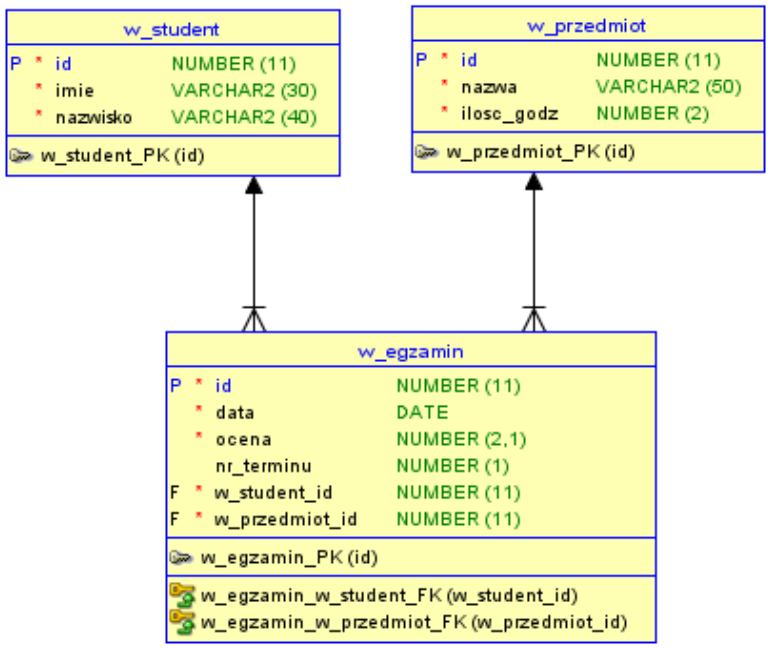
4. Wykorzystaj tabelę pracownicy z zadania 1. Napisz program, który zapyta się użytkownika o nazwę stanowiska, a następnie wypisze na ekranie w porządku alfabetycznym wszystkich pracowników pracujących na zadanym stanowisku. Zastosuj pętlę FOR z kursorem sparametryzowanym.

5. Wykorzystaj tabelę pracownicy z zadania 1. W pewnej firmie powstała inicjatywa przyznania dodatku motywacyjnego dla jej pracowników. Dodatek jest przyznawany procentowo w stosunku do pensji podstawowej każdego pracownika.
   *) Każdemu pracownikowi zarabiającemu poniżej 3000 zł przyznano tyle % dodatku, aby jego pensja plus ten dodatek była równa 3000 zł.
   *) Za każdy pełny rok pracy w firmie przyznano dodatkowo 1% dodatku.
   *) Osoby zajmujące wyższe stanowiska otrzymały dodatkowo: dyrektor – 20%, kierownik – 10%, brygadzista – 5% dodatku.
   *) Kobiety otrzymały dodatkowo dodatek 5%.
   Wszystkie dodatki się sumują i są przyznawane w podanej kolejności, jednak w sumie nie mogą być wyższe niż 60% pensji podstawowej pracownika.

Wyliczone kwoty dodatków dla pracowników należy zapisać w kolumnie dodatek w tabeli pracownicy. (Wartość poprzedniego dodatku w tabeli pracownicy nie ma znaczenia.) Użyj pętli FOR z podzapytaniem.

6. Dana jest tabela
   CREATE TABLE punkt(x INT, y INT);
   i przykładowe rekordy:
   BEGIN
   FOR v_i IN 1..1000 LOOP
   INSERT INTO punkt(x,y) VALUES (MOD(123*v_i,MOD(127*v_i,27)),MOD(147*v_i,37));
   END LOOP;
   END;
   Po wykonaniu zapytania
   SELECT x, y, count(*) AS ile FROM punkt GROUP BY x, y ORDER BY ile DESC;
   zauważono, że niektóre wartości współrzędnych punktów się powtarzają.
   Napisz program w języku PL/SQL, który pozostawi w tabeli punkt unikatowe współrzędne punktów (x,y).
   (Należy usunąć wielokrotne wystąpienia danego punktu (x,y) zachowując pojedyncze wystąpienie tego punktu w tabeli punkt.) Użyj pętli FOR z podzapytaniem.

7. Dane są tabele:



i przykładowe rekordy:

ID	NAZWA	ILOSC_GODZ
1	Bazy danych	60
2	Analiza matematyczna	45
3	Rachunek prawdopodobieństwa	30

ID	IMIE	NAZWISKO
1	Jan	Kowalski
2	Anna	Komar
3	Jerzy	Nowak
4	Sebastian	Rybicki

ID	DATA	OCENA	NR_TERMINU	W_STUDENT_ID	W_PRZEDMIOT_ID
1	14/11/03	2	1	1	3
2	14/11/03	2	1	2	3
3	14/11/03	2	1	3	3
4	14/11/04	4	1	1	1
5	14/11/04	4,5	1	2	1
6	14/11/04	5	1	4	1
7	14/11/11	3	2	1	3
8	14/11/11	2	2	2	3
9	14/11/12	2	1	3	2

a) Napisać instrukcje SQL, które utworzą przedstawioną strukturę tabel i dodadzą przykładowe rekordy.
b) Stworzyć tabelę w_warunek(id,w_student_id, w_przedmiot_id, kwota), gdzie kolumna id powinna być automatycznie autoinkrementowana.
c) Napisać instrukcje PL/SQL, które dla każdego przedmiotu sprawdzą, czy student ma zaliczony już ten przedmiot (zaliczenie przedmiotu=ocena z egzaminu z dowolnego podejścia >2.0). W przypadku braku zaliczenia sprawdzanego przedmiotu należy dodać wpis do tabeli w_warunek z informacją jaki student i za jaki przedmiot ile powinien zapłacić (przyjmijmy, że kwotę wyliczamy wzorem ilość_godzin*30 zł).

Dla danych z zadania należy spodziewać się wyniku:

ID	W_STUDENT_ID	W_PRZEDMIOT_ID	KWOTA
1	1	2	1350
2	4	2	1350
3	3	2	1350
4	3	1	1800
5	2	2	1350
6	3	3	900
7	4	3	900
8	2	3	900

Rozwiązania

1. Utwórz tabelę pracownicy(id_pracownik, imie, nazwisko, plec, data_urodzenia, data_zatrudnienia, stanowisko, pensja, dodatek) i dodaj do niej co najmniej 10 rekordów, np:
   INSERT INTO pracownicy VALUES (1,'Anna','Kowalska','K','1975-05-25','2001-12-01', 'kierownik',2300,500);
   Usuń z tabeli pracownicy co drugi rekord względem nazwisk i imion ułożonych w porządku alfabetycznym (czyli usuwamy 2, 4, 6, itd. rekord w narzuconym porządku). Użyj kursora jawnego i pętli LOOP. Posłuż się poleceniem FETCH.

```
CREATE TABLE pracownicy(
  id_pracownik NUMBER(11),
  imie VARCHAR2(15),
  nazwisko VARCHAR2(20),
  plec CHAR(1),
  data_urodzenia DATE,
  data_zatrudnienia DATE,
  stanowisko VARCHAR2(15),
  pensja NUMBER(8,2),
  dodatek NUMBER(8,2)
);
/
```

```
INSERT INTO pracownicy VALUES
(1,'Ewa','Nowak','K','1989-01-01','2010-11-11','brygadzista',1000,100);
INSERT INTO pracownicy VALUES
(3,'Ala','Nowak','K','1989-01-02','2010-11-12','sprzedawca',1100,110);
INSERT INTO pracownicy VALUES
(5,'Ania','Nowak','K','1989-01-03','2010-11-13','kierownik',3200,120);
INSERT INTO pracownicy VALUES
(7,'Joanna','Nowak','K','1989-01-04','2011-11-14','sprzedawca',3300,130);
INSERT INTO pracownicy VALUES
(9,'Milena','Nowak','K','1989-01-05','2011-11-15','kierownik',3400,140);
INSERT INTO pracownicy VALUES
(11,'Kasia','Nowak','K','1989-01-06','2011-11-16','sprzedawca',3500,150);
INSERT INTO pracownicy VALUES
(13,'Jola','Nowak','K','1989-01-07','2012-11-17','sprzedawca',3600,160);
INSERT INTO pracownicy VALUES
(15,'Basia','Nowak','K','1989-01-08','2012-11-18','kierownik',3700,170);
INSERT INTO pracownicy VALUES
(17,'Monika','Nowak','K','1989-01-09','2013-11-19','dyrektor',3800,180);
INSERT INTO pracownicy VALUES
(19,'Wioleta','Nowak','K','1989-01-10','2013-11-20','manager',3900,190);
/
```

```
34  DECLARE
35    CURSOR cur_pracownicy IS SELECT * FROM pracownicy ORDER BY nazwisko, imie;
36    v_record pracownicy%rowtype;
37    v_id INTEGER:=1;
38  BEGIN
39    OPEN cur_pracownicy;
40    LOOP
41      FETCH cur_pracownicy INTO v_record;
42      EXIT WHEN cur_pracownicy%notfound;
43      IF (MOD(v_id,2)=0) THEN
44        DELETE FROM pracownicy WHERE id_pracownik=v_record.id_pracownik;
45      END IF;
46      v_id:=v_id+1;
47    END LOOP;
48    CLOSE cur_pracownicy;
49  END;
50  /
51  SELECT * FROM pracownicy ORDER BY nazwisko, imie;
```

Stan przed usunięciem rekordów:

Script Output | Query Result

All Rows Fetched: 10 in 0 seconds

	ID_PRACOWNIK	IMIE	NAZWISKO	PLEC	DATA_URODZENIA	DATA_ZATRUDNIENIA	STANOWISKO	PENSJA	DODATEK
1	3	Ala	Nowak	K	89/01/02	10/11/12	sprzedawca	1100	110
2	5	Ania	Nowak	K	89/01/03	10/11/13	kierownik	3200	120
3	15	Basia	Nowak	K	89/01/08	12/11/18	kierownik	3700	170
4	1	Ewa	Nowak	K	89/01/01	10/11/11	brygadzista	1000	100
5	7	Joanna	Nowak	K	89/01/04	11/11/14	sprzedawca	3300	130
6	13	Jola	Nowak	K	89/01/07	12/11/17	sprzedawca	3600	160
7	11	Kasia	Nowak	K	89/01/06	11/11/16	sprzedawca	3500	150
8	9	Milena	Nowak	K	89/01/05	11/11/15	kierownik	3400	140
9	17	Monika	Nowak	K	89/01/09	13/11/19	dyrektor	3800	180
10	19	Wioleta	Nowak	K	89/01/10	13/11/20	manager	3900	190

Stan po usunięciu rekordów:

Script Output | Query Result

All Rows Fetched: 5 in 0 seconds

	ID_PRACOWNIK	IMIE	NAZWISKO	PLEC	DATA_URODZENIA	DATA_ZATRUDNIENIA	STANOWISKO	PENSJA	DODATEK
1	3	Ala	Nowak	K	89/01/02	10/11/12	sprzedawca	1100	110
2	15	Basia	Nowak	K	89/01/08	12/11/18	kierownik	3700	170
3	7	Joanna	Nowak	K	89/01/04	11/11/14	sprzedawca	3300	130
4	11	Kasia	Nowak	K	89/01/06	11/11/16	sprzedawca	3500	150
5	17	Monika	Nowak	K	89/01/09	13/11/19	dyrektor	3800	180

2. Wykorzystaj tabelę pracownicy z zadania 1. Zdefiniuj kursor jawny zawierający imiona, nazwiska i daty zatrudnienia wszystkich kierowników. Posłuż się kursorem do wyświetlenia rekordów w formie zdań: "Pracownik <imie> <nazwisko> pracuje na stanowisku kierownika od <data_zatrudnienia>.". Użyj pętli WHILE i posłuż się poleceniem FETCH.

```
1  --wersja z kursorem
2  SET SERVEROUTPUT ON
3  DECLARE
4    CURSOR cur_pracownicy IS SELECT imie,nazwisko,data_zatrudnienia
5      FROM pracownicy WHERE stanowisko='kierownik';
6    v_imie pracownicy.imie%TYPE;
7    v_nazwisko pracownicy.nazwisko%TYPE;
8    v_data_zatrudnienia pracownicy.data_zatrudnienia%TYPE;
9  BEGIN
10   OPEN cur_pracownicy;
11   FETCH cur_pracownicy INTO v_imie, v_nazwisko, v_data_zatrudnienia;
12   WHILE (cur_pracownicy%found) LOOP
13     dbms_output.put_line('Pracownik '||v_imie||' ' ||v_nazwisko||
14       ' pracuje na stanowisku kierownika od '||v_data_zatrudnienia||'.');
15     FETCH cur_pracownicy INTO v_imie, v_nazwisko, v_data_zatrudnienia;
16   END LOOP;
17   CLOSE cur_pracownicy;
18 END;
```

```
19 --wersja bez kursora
20 SELECT 'Pracownik '||imie||' ' ||nazwisko||
21   ' pracuje na stanowisku kierownika od '||data_zatrudnienia||'.' AS dane
22 FROM pracownicy
23 WHERE stanowisko='kierownik';
```

3. Wykorzystaj tabelę pracownicy z zadania 1. Zdefiniuj kursor jawny, dzięki któremu będzie można wyświetlić na ekranie trzech najlepiej zarabiających pracowników patrząc na wysokość zarobków (bez dodatku). Uwzględnij miejsca ex aequo. Użyj pętli FOR.

```
1  DECLARE
2    CURSOR cur_pracownicy IS SELECT * FROM pracownicy ORDER BY pensja DESC;
3    v_record pracownicy%rowtype;
4    v_ile int:=3; --ilość unikatowych pensji
5    v_pensja int:=-1;
6    v_i INT:=0; --numerujemy pensje
7  BEGIN
8    FOR v_rekord IN cur_pracownicy LOOP
9      IF (v_pensja!=v_rekord.pensja) THEN
10       v_ile:=v_ile-1;
11       v_i:=v_i+1;
12     END IF;
13     EXIT WHEN v_ile<0;
14     v_pensja:=v_rekord.pensja;
15     dbms_output.put_line(v_i||') '||v_rekord.imie||' '||v_rekord.nazwisko||
16       ' -> '||v_rekord.pensja||' zł');
17   END LOOP;
18 END;
```

Script Output | Task completed in 0,031 seconds

```
anonymous block completed
1) Wioleta Nowak -> 3900 zł
2) Monika Nowak -> 3800 zł
3) Basia Nowak -> 3700 zł
```

4. Wykorzystaj tabelę pracownicy z zadania 1. Napisz program, który zapyta się użytkownika o nazwę stanowiska, a następnie wypisze na ekranie w porządku alfabetycznym wszystkich pracowników pracujących na zadanym stanowisku. Zastosuj pętlę FOR z kursorem sparametryzowanym.

```
1  DECLARE
2    v_nazwa pracownicy.stanowisko%TYPE:='&Podaj_nazwe_stanowiska';
3    CURSOR cur_pracownicy(v_stanowisko VARCHAR2) IS
4      SELECT * FROM pracownicy WHERE stanowisko=v_stanowisko;
5  BEGIN
6    FOR v_rekord IN cur_pracownicy(v_nazwa) LOOP
7      dbms_output.put_line(v_rekord.stanowisko||': '||v_rekord.imie||' '
8        ||v_rekord.nazwisko);
9    END LOOP;
10 END;
```

5. Wykorzystaj tabelę pracownicy z zadania 1. W pewnej firmie powstała inicjatywa przyznania dodatku motywacyjnego dla jej pracowników. Dodatek jest przyznawany procentowo w stosunku do pensji podstawowej każdego pracownika.
   *) Każdemu pracownikowi zarabiającemu poniżej 3000 zł przyznano tyle % dodatku, aby jego pensja plus ten dodatek była równa 3000 zł.
   *) Za każdy pełny rok pracy w firmie przyznano dodatkowo 1% dodatku.
   *) Osoby zajmujące wyższe stanowiska otrzymały dodatkowo: dyrektor – 20%, kierownik – 10%, brygadzista – 5% dodatku.
   *) Kobiety otrzymały dodatkowo dodatek 5%.
   Wszystkie dodatki się sumują i są przyznawane w podanej kolejności, jednak w sumie nie mogą być wyższe niż 60% pensji podstawowej pracownika.
   Wyliczone kwoty dodatków dla pracowników należy zapisać w kolumnie dodatek w tabeli pracownicy. (Wartość poprzedniego dodatku w tabeli pracownicy nie ma znaczenia.) Użyj pętli FOR z podzapytaniem.

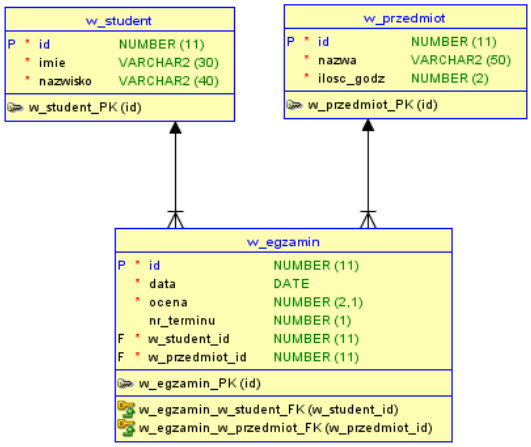
```
1  DECLARE
2    CURSOR cur_pracownicy IS SELECT * FROM pracownicy FOR UPDATE;
3    v_dodatek DECIMAL(8,2);
4    v_ile_dodatkow INTEGER;
5  BEGIN
6    FOR v_rekord IN cur_pracownicy LOOP
7
8      v_dodatek:= 0;
9      v_ile_dodatkow:=0;
10
11     --gwiazdka numer 1
12     IF (v_rekord.pensja<3000) THEN
13       v_dodatek:=(3000-v_rekord.pensja)/v_rekord.pensja*100;
14       v_ile_dodatkow:=v_ile_dodatkow+1;
15     END IF;
16
```

```
17      --gwiazdka numer 2
18      IF (round((sysdate-v_rekord.data_zatrudnienia)/365)>0) THEN
19          v_dodatek:=v_dodatek+round((sysdate-v_rekord.data_zatrudnienia)/365);
20          v_ile_dodatkow:=v_ile_dodatkow+1;
21      END IF;
22
23      --gwiazdka numer 3
24      CASE v_rekord.stanowisko
25       WHEN 'dyrektor' THEN v_dodatek:=v_dodatek+20; v_ile_dodatkow:=v_ile_dodatkow+1;
26       WHEN 'kierownik' THEN v_dodatek:=v_dodatek+10; v_ile_dodatkow:=v_ile_dodatkow+1;
27       WHEN 'brygadzista' THEN  v_dodatek:=v_dodatek+5; v_ile_dodatkow:=v_ile_dodatkow+1;
28       ELSE NULL;
29      END CASE;
30
31      --gwiazdka numer 4
32      IF (v_rekord.plec='K') THEN
33        v_dodatek:=v_dodatek+5;
34        v_ile_dodatkow:=v_ile_dodatkow+1;
35      END IF;
36
37      --korekta dodatku
38      IF (v_dodatek>60) THEN
39        v_dodatek:=60;
40      END IF;
41      UPDATE pracownicy SET dodatek=v_rekord.pensja*v_dodatek/100
42      WHERE CURRENT OF cur_pracownicy;
43
44      dbms_output.put_line('Dodatek dla '||v_rekord.imie||' '||v_rekord.nazwisko||
45        ' dodatek wynosi: '|| round(v_rekord.pensja*v_dodatek/100,2)||
46        '%. Ilosc uwzglednionych dodatkow: '|| v_ile_dodatkow );
47    END LOOP;
48  end;
49  /
```

6. Dana jest tabela
   CREATE TABLE punkt(x INT, y INT);
   i przykładowe rekordy:
   BEGIN
    FOR v_i IN 1..1000 LOOP
     INSERT INTO punkt(x,y) VALUES (MOD(123*v_i,MOD(127*v_i,27)),MOD(147*v_i,37));
    END LOOP;
   END;
   Po wykonaniu zapytania
   SELECT x, y, count(*) AS ile FROM punkt GROUP BY x, y ORDER BY ile DESC;
   zauważono, że niektóre wartości współrzędnych punktów się powtarzają.
   Napisz program w języku PL/SQL, który pozostawi w tabeli punkt unikatowe współrzędne punktów (x,y).
   (Należy usunąć wielokrotne wystąpienia danego punktu (x,y) zachowując pojedyncze wystąpienie tego punktu w tabeli punkt.) Użyj pętli FOR z podzapytaniem.

```
1  CREATE TABLE punkt(x INT, y INT);
2  /
3  BEGIN
4    FOR v_i IN 1..1000 LOOP
5      INSERT INTO punkt(x,y) VALUES (MOD(123*v_i, MOD(127*v_i,27)),MOD(147*v_i,37));
6    END LOOP;
7  END;
8  /
9  SELECT 'wszystkie', count(*) FROM punkt;
10 SELECT 'unikatowe', count(*) FROM (SELECT DISTINCT x, y FROM punkt);
11 /
12 BEGIN
13   FOR v_rekord IN (SELECT x, y FROM punkt GROUP BY x, y HAVING count(x)>1)
14   LOOP
15     DELETE FROM punkt WHERE v_rekord.x=x AND v_rekord.y=y;
16     INSERT INTO punkt(x,y) VALUES(v_rekord.x,v_rekord.y);
17   END LOOP;
18 END;
19 /
20 SELECT 'wszystkie', count(*) FROM punkt;
21 SELECT 'unikatowe', count(*) FROM (SELECT DISTINCT x,y FROM punkt);
```

7. Dane są tabele:



i przykładowe rekordy:

ID	NAZWA	ILOSC_GODZ
1	Bazy danych	60
2	Analiza matematyczna	45
3	Rachunek prawdopodobieństwa	30

ID	IMIE	NAZWISKO
1	Jan	Kowalski
2	Anna	Komar
3	Jerzy	Nowak
4	Sebastian	Rybicki

ID	DATA	OCENA	NR_TERMINU	W_STUDENT_ID	W_PRZEDMIOT_ID
1	14/11/03	2	1	1	3
2	14/11/03	2	1	2	3
3	14/11/03	2	1	3	3
4	14/11/04	4	1	1	1
5	14/11/04	4,5	1	2	1
6	14/11/04	5	1	4	1
7	14/11/11	3	2	1	3
8	14/11/11	2	2	2	3
9	14/11/12	2	1	3	2

a) Napisać instrukcje SQL, które utworzą przedstawioną strukturę tabel i dodadzą przykładowe rekordy.
b) Stworzyć tabelę w_warunek(id,w_student_id, w_przedmiot_id, kwota), gdzie kolumna id powinna być automatycznie autoinkrementowana.
c) Napisać instrukcje PL/SQL, które dla każdego przedmiotu sprawdzą, czy student ma zaliczony już ten przedmiot (zaliczenie przedmiotu=ocena z egzaminu z dowolnego podejścia >2.0). W przypadku braku zaliczenia sprawdzanego przedmiotu należy dodać wpis do tabeli w_warunek z informacją jaki student i za jaki przedmiot ile powinien zapłacić (przyjmijmy, że kwotę wyliczamy wzorem ilość_godzin*30 zł).

Dla danych z zadania należy spodziewać się wyniku:

ID	W_STUDENT_ID	W_PRZEDMIOT_ID	KWOTA
1	1	2	1350
2	4	2	1350
3	3	2	1350
4	3	1	1800
5	2	2	1350
6	3	3	900
7	4	3	900
8	2	3	900

Rozwiązanie zadania 7:

```
1  CREATE TABLE w_egzamin
2    (
3      id              NUMBER(11) NOT NULL ,
4      data            DATE NOT NULL ,
5      ocena           NUMBER(2,1) NOT NULL ,
6      nr_terminu      NUMBER(1) ,
7      w_student_id    NUMBER(11) NOT NULL ,
8      w_przedmiot_id NUMBER(11) NOT NULL
9    ) ;
10 ALTER TABLE w_egzamin ADD CONSTRAINT w_egzamin_PK PRIMARY KEY ( id ) ;
11
12 CREATE TABLE w_przedmiot
13   (
14     id          NUMBER(11) NOT NULL ,
15     nazwa       VARCHAR2(50) NOT NULL ,
16     ilosc_godz NUMBER(2) NOT NULL
17   ) ;
18 ALTER TABLE w_przedmiot ADD CONSTRAINT w_przedmiot_PK PRIMARY KEY ( id ) ;
19
```

```
20 CREATE TABLE w_student
21   (
22     id       NUMBER(11) NOT NULL ,
23     imie     VARCHAR2(30) NOT NULL ,
24     nazwisko VARCHAR2(40) NOT NULL
25   ) ;
26 ALTER TABLE w_student ADD CONSTRAINT w_student_PK PRIMARY KEY ( id ) ;
27
28 ALTER TABLE w_egzamin ADD CONSTRAINT w_egzamin_w_przedmiot_FK
29 FOREIGN KEY ( w_przedmiot_id ) REFERENCES w_przedmiot ( id ) ;
30
31 ALTER TABLE w_egzamin ADD CONSTRAINT w_egzamin_w_student_FK
32 FOREIGN KEY ( w_student_id ) REFERENCES w_student ( id ) ;
33 /
34 --b)
35 CREATE TABLE w_warunek
36   (
37     id              NUMBER(11) NOT NULL ,
38     w_student_id    NUMBER(11) NOT NULL ,
39     w_przedmiot_id NUMBER(11) NOT NULL ,
40     kwota           NUMBER(8,2) NOT NULL
41   ) ;
42 ALTER TABLE w_warunek ADD CONSTRAINT w_warunek_PK PRIMARY KEY ( id ) ;
43
44 ALTER TABLE w_warunek ADD CONSTRAINT w_warunek_w_przedmiot_FK
45 FOREIGN KEY ( w_przedmiot_id ) REFERENCES w_przedmiot ( id ) ;
46
47 ALTER TABLE w_warunek ADD CONSTRAINT w_warunek_w_student_FK
48 FOREIGN KEY ( w_student_id ) REFERENCES w_student ( id ) ;
49
```

```
CREATE SEQUENCE w_warunek_id_SEQ START WITH 1 NOCACHE ORDER ;
CREATE OR REPLACE TRIGGER w_warunek_id_TRG BEFORE
  INSERT ON w_warunek FOR EACH ROW WHEN (NEW.id IS NULL)
BEGIN
  :NEW.id := w_warunek_id_SEQ.NEXTVAL;
END;
/
Insert into W_PRZEDMIOT (ID,NAZWA,ILOSC_GODZ)
values ('1','Bazy danych','60');
Insert into W_PRZEDMIOT (ID,NAZWA,ILOSC_GODZ)
values ('2','Analiza matematyczna','45');
Insert into W_PRZEDMIOT (ID,NAZWA,ILOSC_GODZ)
values ('3','Rachunek prawdopodobieństwa','30');

Insert into W_STUDENT (ID,IMIE,NAZWISKO) values ('1','Jan','Kowalski');
Insert into W_STUDENT (ID,IMIE,NAZWISKO) values ('2','Anna','Komar');
Insert into W_STUDENT (ID,IMIE,NAZWISKO) values ('3','Jerzy','Nowak');
Insert into W_STUDENT (ID,IMIE,NAZWISKO) values ('4','Sebastian','Rybicki');

Insert into W_EGZAMIN (ID,DATA,OCENA,NR_TERMINU,W_STUDENT_ID,W_PRZEDMIOT_ID)
values ('1',to_date('14/11/03','RR/MM/DD'),'2','1','1','3');
Insert into W_EGZAMIN (ID,DATA,OCENA,NR_TERMINU,W_STUDENT_ID,W_PRZEDMIOT_ID)
values ('2',to_date('14/11/03','RR/MM/DD'),'2','1','2','3');
Insert into W_EGZAMIN (ID,DATA,OCENA,NR_TERMINU,W_STUDENT_ID,W_PRZEDMIOT_ID)
values ('3',to_date('14/11/03','RR/MM/DD'),'2','1','3','3');
Insert into W_EGZAMIN (ID,DATA,OCENA,NR_TERMINU,W_STUDENT_ID,W_PRZEDMIOT_ID)
values ('4',to_date('14/11/04','RR/MM/DD'),'4','1','1','1');
Insert into W_EGZAMIN (ID,DATA,OCENA,NR_TERMINU,W_STUDENT_ID,W_PRZEDMIOT_ID)
values ('5',to_date('14/11/04','RR/MM/DD'),'4,5','1','2','1');
Insert into W_EGZAMIN (ID,DATA,OCENA,NR_TERMINU,W_STUDENT_ID,W_PRZEDMIOT_ID)
values ('6',to_date('14/11/04','RR/MM/DD'),'5','1','4','1');
Insert into W_EGZAMIN (ID,DATA,OCENA,NR_TERMINU,W_STUDENT_ID,W_PRZEDMIOT_ID)
values ('7',to_date('14/11/11','RR/MM/DD'),'3','2','1','3');
Insert into W_EGZAMIN (ID,DATA,OCENA,NR_TERMINU,W_STUDENT_ID,W_PRZEDMIOT_ID)
values ('8',to_date('14/11/11','RR/MM/DD'),'2','2','2','3');
Insert into W_EGZAMIN (ID,DATA,OCENA,NR_TERMINU,W_STUDENT_ID,W_PRZEDMIOT_ID)
values ('9',to_date('14/11/12','RR/MM/DD'),'2','1','3','2');
/
```

```
88  --c)
89  DECLARE
90    CURSOR cur_warunek IS
91      SELECT p.id AS przedmiot, s.id AS student,
92             max(coalesce(e.ocena,2.0)) AS ocena_koncowa
93      FROM w_przedmiot p CROSS JOIN w_student s
94           FULL JOIN w_egzamin e
95           ON e.w_student_id=s.id AND e.w_przedmiot_id=p.id
96      GROUP BY p.id, s.id
97      HAVING max(coalesce(e.ocena,2.0))=2;
98    v_stawka number(8,2) := 30;
99    v_ilosc_godz number(2);
100 BEGIN
101   DELETE FROM w_warunek;
102   FOR v_rek IN cur_warunek
103   LOOP
104     SELECT ilosc_godz INTO v_ilosc_godz
105     FROM w_przedmiot
106     WHERE id = v_rek.przedmiot;
107
108     INSERT INTO w_warunek (w_student_id, w_przedmiot_id, kwota)
109     VALUES(v_rek.student, v_rek.przedmiot, (v_ilosc_godz*v_stawka));
110   END LOOP;
111 END;
```