# **Raport z ćwiczenia 1 – ORACLE PL/SQL**

Magdalena Nowak, czw. 14.40 B

Zad.1

*Tabele stworzyłam korzystając z kodu przeklejonego z treści zadania.*

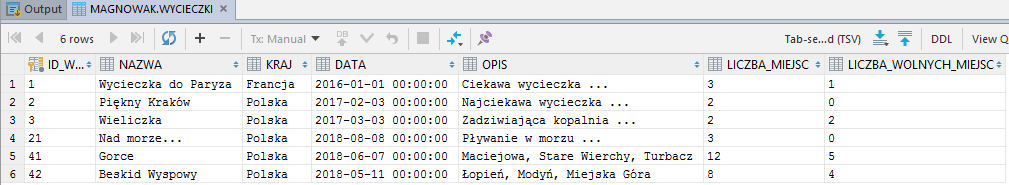
Zad.2

*Następnie wypełniłam tabele danymi:*

Dane wstawione do tabeli **WYCIECZKI**:

**INSERT INTO** wycieczki (**nazwa**, **kraj**, **data**, **opis**, **liczba\_miejsc**) **VALUES** (**'Wycieczka do Paryza'**,**'Francja'**,**'2016-01-01'**,**'Ciekawa wycieczka ...'**,3);  
  
**INSERT INTO** wycieczki (**nazwa**, **kraj**, **data**, **opis**, **liczba\_miejsc**) **VALUES** (**'Piękny Kraków'**,**'Polska'**,**'2017-02-03'**,**'Najciekawa wycieczka ...'**,2);  
  
**INSERT INTO** wycieczki (**nazwa**, **kraj**, **data**, **opis**, **liczba\_miejsc**) **VALUES** (**'Wieliczka'**,**'Polska'**,**'2017-03-03'**,**'Zadziwiająca kopalnia ...'**,2);  
  
**INSERT INTO** wycieczki (**nazwa**, **kraj**, **data**, **opis**, **liczba\_miejsc**) **VALUES** (**'Gorce'**,**'Polska'**,**'2018-06-07'**,**'Maciejowa, Stare Wierchy, Turbacz'**,10);  
  
**INSERT INTO** wycieczki (**nazwa**, **kraj**, **data**, **opis**, **liczba\_miejsc**) **VALUES** (**'Nad morze...'**,**'Polska'**,**'2018-08-08'**,**'Pływanie w morzu ...'**,3);

SELECT \* FROM WYCIECZKI;

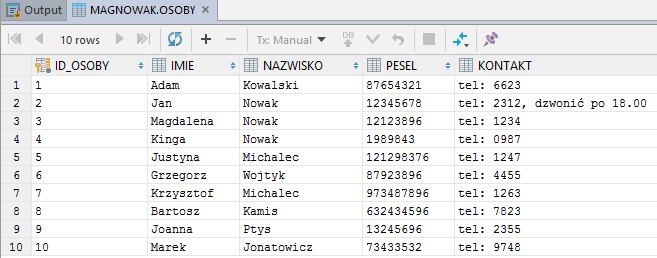


*Dane wstawione do tabeli* ***OSOBY****:*

**INSERT INTO** osoby (**imie**, **nazwisko**, **pesel**, **kontakt**) **VALUES**(**'Adam'**, **'Kowalski'**, **'87654321'**, **'tel: 6623'**);  
  
**INSERT INTO** osoby (**imie**, **nazwisko**, **pesel**, **kontakt**) **VALUES**(**'Jan'**, **'Nowak'**, **'12345678'**, **'tel: 2312, dzwonić po 18.00'**);  
  
**INSERT INTO** osoby (**imie**, **nazwisko**, **pesel**, **kontakt**) **VALUES**(**'Magdalena'**, **'Nowak'**, **'12123896'**, **'tel: 1234'**);

**INSERT INTO** osoby (**imie**, **nazwisko**, **pesel**, **kontakt**) **VALUES**(**'Kinga'**, **'Nowak'**, **'1989843'**, **'tel: 0987'**);

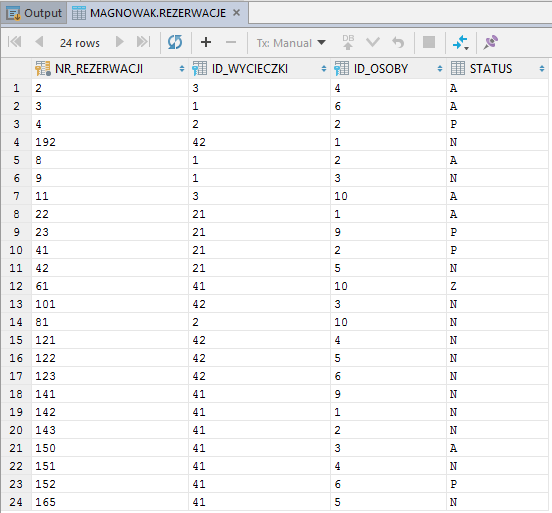
I**NSERT INTO** osoby (**imie**, **nazwisko**, **pesel**, **kontakt**) **VALUES**(**'Justyna'**, **'Michalec'**, **'121298376'**, **'tel: 1247'**);  
  
**INSERT INTO** osoby (**imie**, **nazwisko**, **pesel**, **kontakt**) **VALUES**(**'Grzegorz'**, **'Wojtyk'**, **'87923896'**, **'tel: 4455'**);  
  
**INSERT INTO** osoby (**imie**, **nazwisko**, **pesel**, **kontakt**) **VALUES**(**'Krzysztof'**, **'Michalec'**, **'973487896'**, **'tel: 1263'**);  
  
**INSERT INTO** osoby (**imie**, **nazwisko**, **pesel**, **kontakt**) **VALUES**(**'Bartosz'**, **'Kamis'**, **'632434596'**, **'tel: 7823'**);  
  
**INSERT INTO** osoby (**imie**, **nazwisko**, **pesel**, **kontakt**) **VALUES**(**'Joanna'**, **'Ptys'**, **'13245696'**, **'tel: 2355'**);  
  
**INSERT INTO** osoby (**imie**, **nazwisko**, **pesel**, **kontakt**) **VALUES**(**'Marek'**, **'Jonatowicz'**, **'73433532'**, **'tel: 9748'**);



*Dane wstawione do tabeli* ***REZERWACJE:***

**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (3,4,**'A'**);  
  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (1,6,**'P'**);  
  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (2,2,**'P'**);  
  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (21,1,**'A'**);  
  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (1,2,**'P'**);  
  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (1,3,**'N'**);

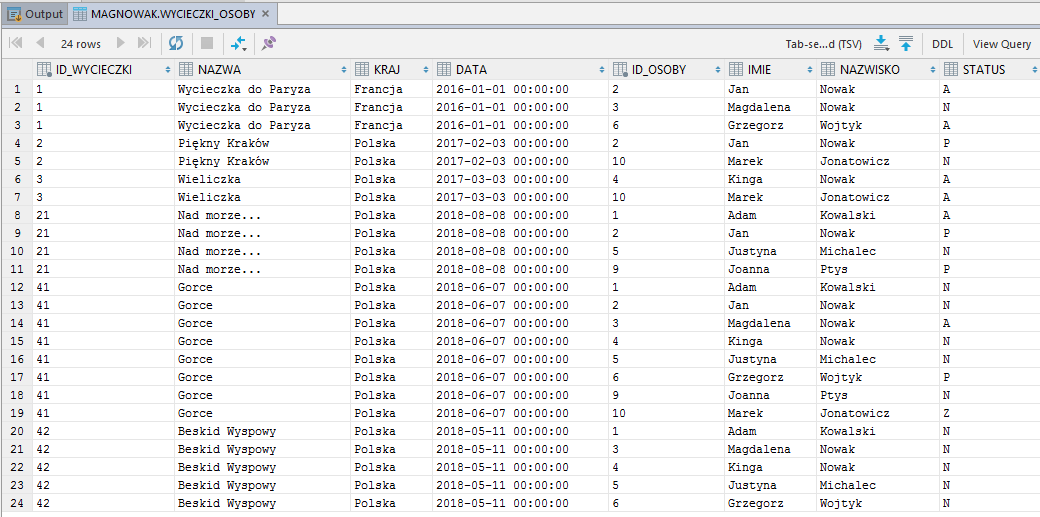
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (1,1,**'N'**);  
  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (3,10,**'A'**);  
  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (21,9,**'P'**);  
  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (21,2,**'N'**);  
  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (21,5,**'N'**);  
  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (41,10,**'N'**);  
**INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (41,5,**'N'**);



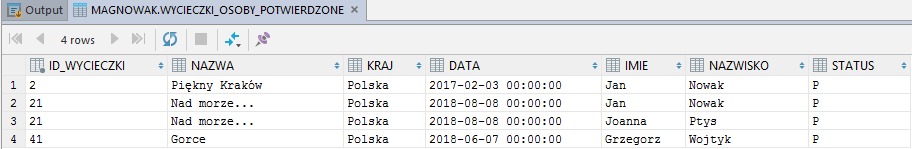
Zad. 3

*Przygotowałam następujący zestaw widoków:*

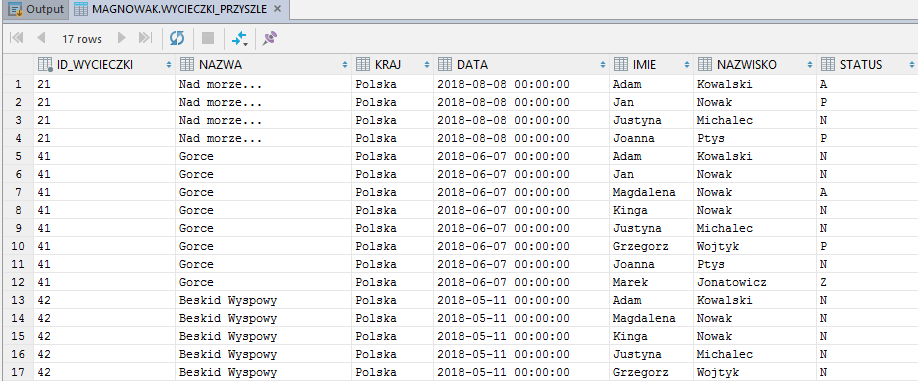
**CREATE OR REPLACE VIEW** wycieczki\_osoby  
 **AS  
 SELECT** w.**ID\_WYCIECZKI**, w.**NAZWA**, w.**KRAJ**, w.**DATA**, o.**ID\_OSOBY**, o.**IMIE**, o.**NAZWISKO**,  
 r.**STATUS  
 FROM** WYCIECZKI w  
 **JOIN** REZERWACJE r **ON** w.**ID\_WYCIECZKI** = r.**ID\_WYCIECZKI  
 JOIN** OSOBY o **ON** r.**ID\_OSOBY** = o.**ID\_OSOBY**;



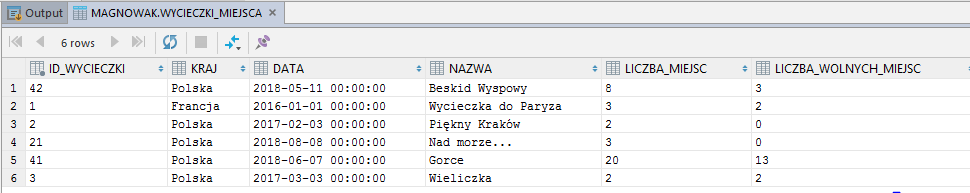
**CREATE OR REPLACE VIEW** wycieczki\_osoby\_potwierdzone  
 **AS  
 SELECT** w.**ID\_WYCIECZKI**, **NAZWA**, **KRAJ**, **DATA**, w.**IMIE**, w.**NAZWISKO**, w.**STATUS  
 FROM** wycieczki\_osoby w  
 **WHERE** w.**STATUS** = **'P'**;



**CREATE OR REPLACE VIEW** wycieczki\_przyszle  
 **AS  
 SELECT** w.**ID\_WYCIECZKI**, **NAZWA**, **KRAJ**, **DATA**, w.**IMIE**, w.**NAZWISKO**, w.**STATUS  
 FROM** wycieczki\_osoby w  
 **WHERE DATA** > *CURRENT\_DATE*;



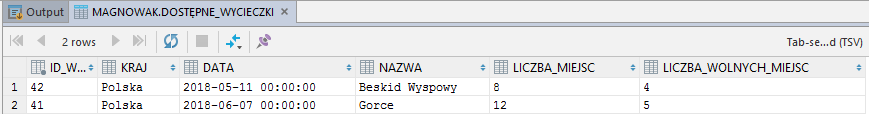
**CREATE OR REPLACE VIEW** wycieczki\_miejsca  
 **AS  
 SELECT** w.**ID\_WYCIECZKI**, w.**KRAJ**, w.**DATA**, w.**NAZWA**, w.**LICZBA\_MIEJSC**,  
 w.**LICZBA\_MIEJSC** - *NVL*(r.LICZBA\_REZERWACJI,0) **AS** "LICZBA\_WOLNYCH\_MIEJSC"  
 **FROM** wycieczki w  
 **LEFT OUTER JOIN** (**SELECT** w.**ID\_WYCIECZKI**,  
 *count* (r.**ID\_WYCIECZKI**) **AS** "LICZBA\_REZERWACJI"  
 **FROM** wycieczki w  
 **LEFT OUTER JOIN** REZERWACJE r  
 **ON** r.**ID\_WYCIECZKI** = w.**ID\_WYCIECZKI  
 WHERE** r.**STATUS** != **'A'  
 GROUP BY** w.**ID\_WYCIECZKI**, w.**LICZBA\_MIEJSC**) r  
 **ON** r.**ID\_WYCIECZKI** = w.**ID\_WYCIECZKI**;



*--widok zwraca tylko te wycieczki, na które są jeszcze miejsca i jeszcze się nie odbyły***CREATE OR REPLACE VIEW** dostępne\_wycieczki

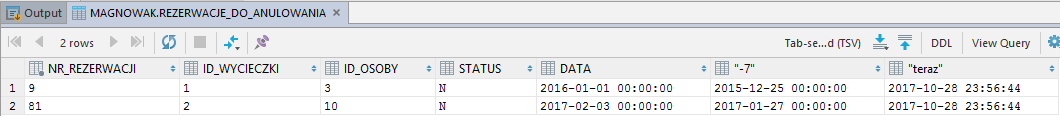
**AS  
 SELECT** *\** **FROM** wycieczki\_miejsca w  
 **WHERE** w.LICZBA\_WOLNYCH\_MIEJSC > 0 **and** w.**DATA** > *CURRENT\_DATE*;

**SELECT** *\** **FROM** dostępne\_wycieczki;



*--widok zwraca te rezerwacje, które są rezerwacajmi na wycieczki, które się odbędą za mniej niż tydzień,   
--a nie są jeszcze potwierdzone, czyli mają status 'N'*  
**CREATE OR REPLACE VIEW** rezerwacje\_do\_anulowania **AS  
 SELECT** r.**NR\_REZERWACJI**, r.**ID\_WYCIECZKI**, r.**ID\_OSOBY**, r.**STATUS  
 FROM** REZERWACJE r  
 **JOIN** WYCIECZKI w **ON** w.**ID\_WYCIECZKI** = r.**ID\_WYCIECZKI  
 WHERE** r.**STATUS** = **'N' AND** *CURRENT\_DATE* > w.**DATA** - 7;

**SELECT** *\** **FROM** rezerwacje\_do\_anulowania;



Zad. 4

*Przygotowałam też zestaw procedur i funkcji ułatwiających dostęp do danych.*

*We wszystkih miejscach, gdzie potrzebowałam rzucenia wyjątku korzystam z mechanizmu rzucania wyjątku aplikacji i definiowałam numer wyjątku wraz z krótkim opisem.*

*Do sprawdzania poprawności wprowadzanych argumentów dopisałam dwie procedury czy\_istnieje\_taka\_osoba i czy\_istnieje\_taka\_wycieczka. Ich jedynym zadaniem jest sprawdzenie poprawności argumentów i ewentualne rzucenie wyjątku. Korzystam z nich później również w innych punktach.*

**CREATE OR REPLACE PROCEDURE** *czy\_istnieje\_taka\_osoba* (osoba\_id **IN** wycieczki\_osoby.**ID\_WYCIECZKI**%**TYPE**)  
 **AS** v\_check osoby.**ID\_OSOBY**%**TYPE**;  
 NIE\_MA\_OSOBY\_O\_TAKIM\_ID **EXCEPTION**;  
 **BEGIN  
 SELECT** *COUNT*(*\**)  
 **INTO** v\_check **FROM** OSOBY  
 **WHERE** osoby.**ID\_OSOBY**= osoba\_id;  
 **if**(v\_check = 0) **then RAISE** NIE\_MA\_OSOBY\_O\_TAKIM\_ID;  
 **END if**;  
 **EXCEPTION  
 WHEN** NIE\_MA\_OSOBY\_O\_TAKIM\_ID **THEN** *RAISE\_APPLICATION\_ERROR*(-20001,**'Nie ma osoby o podanym ID'**);  
 **END**;  
  
**CREATE OR REPLACE PROCEDURE** *czy\_istnieje\_taka\_wycieczka* (wycieczka\_id **IN** wycieczki.**ID\_WYCIECZKI**%**TYPE**)  
 **AS** v\_check WYCIECZKI.**ID\_WYCIECZKI**%**TYPE**;  
 NIE\_MA\_WYCIECZKI\_O\_TAKIM\_ID **EXCEPTION**;  
 **BEGIN  
 SELECT** *COUNT*(*\**)  
 **INTO** v\_check **FROM** WYCIECZKI  
 **WHERE** WYCIECZKI.**ID\_WYCIECZKI** = wycieczka\_id;  
 **IF**(v\_check = 0) **THEN  
 RAISE** NIE\_MA\_WYCIECZKI\_O\_TAKIM\_ID;  
 **END IF**;  
 **EXCEPTION  
 WHEN** NIE\_MA\_WYCIECZKI\_O\_TAKIM\_ID **THEN** *RAISE\_APPLICATION\_ERROR*(-20002,**'Nie ma wycieczki o podanym ID'**);  
 **END**;

*Stworzone przeze mnie funkcje są funkcjami strumieniowymi, zwracają tablice typu rekordowego. W tym celu stworzyłam nowy typ, rekord rezerwacja oraz typ tablicowy tabela\_rezerwacji, który jest tabelą składajacą się z rekordów typu rezerwacja.*

**OR REPLACE TYPE** rezerwacja **AS OBJECT** (  
 ID\_WYCIECZKI **INT**,  
 NAZWA **VARCHAR2**(100),  
 KRAJ **VARCHAR2**(50),  
 **DATA DATE**,  
 ID\_OSOBY **INT**,  
 IMIE **VARCHAR2**(50),  
 NAZWISKO **VARCHAR2**(50),  
 STATUS **CHAR**);

**CREATE OR REPLACE TYPE** tabela\_rezerwacji **AS TABLE OF** rezerwacja;

*Funkcje i procedury:*  
  
**CREATE OR REPLACE FUNCTION** *uczestnicy\_wycieczki* (wycieczka\_id **IN** wycieczki\_osoby.**ID\_WYCIECZKI**%**TYPE**)  
 **return** tabela\_rezerwacji **PIPELINED  
 AS  
 CURSOR** cur\_wycieczki\_osoby **is  
 select** *\** **from** wycieczki\_osoby w  
 **where** w.**ID\_WYCIECZKI** = wycieczka\_id;  
 rekord rezerwacja := rezerwacja(**NULL**, **NULL**, **NULL**, **NULL**, **NULL**, **NULL**, **NULL**, **NULL**);  
 **BEGIN** *czy\_istnieje\_taka\_wycieczka*(wycieczka\_id);  
 **OPEN** cur\_wycieczki\_osoby;  
 **LOOP  
 FETCH** cur\_wycieczki\_osoby **INTO** rekord.ID\_WYCIECZKI, rekord.NAZWA, rekord.KRAJ,rekord.DATA, rekord.ID\_OSOBY,  
 rekord.IMIE, rekord.NAZWISKO, rekord.STATUS;  
 **EXIT WHEN** cur\_wycieczki\_osoby%**NOTFOUND**;  
 **PIPE ROW** (rekord);  
 **END LOOP**;  
 **CLOSE** cur\_wycieczki\_osoby;  
 **END**;

**CREATE OR REPLACE FUNCTION** *rezerwacje\_osoby* (osoba\_id **IN** wycieczki\_osoby.**ID\_WYCIECZKI**%**TYPE**)  
 **RETURN** tabela\_rezerwacji **PIPELINED  
 AS  
 CURSOR** cur\_wycieczki\_osoby **is  
 SELECT** *\** **FROM** wycieczki\_osoby w  
 **WHERE** w.**ID\_OSOBY** = osoba\_id;  
 rekord rezerwacja := rezerwacja(**NULL**, **NULL**, **NULL**, **NULL**, **NULL**, **NULL**, **NULL**, **NULL**);  
 **BEGIN** *czy\_istnieje\_taka\_osoba*(osoba\_id);  
 **OPEN** cur\_wycieczki\_osoby;  
 **LOOP  
 FETCH** cur\_wycieczki\_osoby **INTO** rekord.ID\_WYCIECZKI, rekord.NAZWA, rekord.KRAJ,rekord.DATA, rekord.ID\_OSOBY,  
 rekord.IMIE, rekord.NAZWISKO, rekord.STATUS;  
 **EXIT WHEN** cur\_wycieczki\_osoby%**NOTFOUND**;  
 **PIPE ROW** (rekord);  
 **END LOOP**;  
 **CLOSE** cur\_wycieczki\_osoby;  
 **END**;

**CREATE OR REPLACE FUNCTION** *przyszle\_rezerwacje\_osoby* (osoba\_id **IN** wycieczki\_osoby.**ID\_WYCIECZKI**%**TYPE**)  
 **RETURN** tabela\_rezerwacji **PIPELINED  
 AS  
 CURSOR** cur\_wycieczki\_osoby **is  
 SELECT** *\** **FROM** wycieczki\_osoby w  
 **WHERE** w.**ID\_OSOBY** = osoba\_id **and** w.**DATA** > *CURRENT\_DATE*;  
 rekord rezerwacja := rezerwacja(**NULL**, **NULL**, **NULL**, **NULL**, **NULL**, **NULL**, **NULL**, **NULL**);  
 **BEGIN** *czy\_istnieje\_taka\_osoba*(osoba\_id);  
 **OPEN** cur\_wycieczki\_osoby;  
 **LOOP  
 FETCH** cur\_wycieczki\_osoby **INTO** rekord.ID\_WYCIECZKI, rekord.NAZWA, rekord.KRAJ,rekord.DATA, rekord.ID\_OSOBY,  
 rekord.IMIE, rekord.NAZWISKO, rekord.STATUS;  
 **EXIT WHEN** cur\_wycieczki\_osoby%**NOTFOUND**;  
 **PIPE ROW** (rekord);  
 **END LOOP**;  
 **CLOSE** cur\_wycieczki\_osoby;  
 **END**;

**CREATE OR REPLACE PROCEDURE** *czy\_istnieje\_taka\_rezerwacja* (rezerwacja\_id **IN** REZERWACJE.**NR\_REZERWACJI**%**TYPE**)  
 **AS** v\_check REZERWACJE.**NR\_REZERWACJI**%**TYPE**;  
 NIE\_MA\_REZERWACJI\_O\_TAKIM\_ID **EXCEPTION**;  
 **BEGIN  
 SELECT** *COUNT*(*\**)  
 **INTO** v\_check **FROM** REZERWACJE  
 **WHERE** REZERWACJE.**NR\_REZERWACJI** = rezerwacja\_id;  
 **IF**(v\_check = 0) **THEN  
 RAISE** NIE\_MA\_REZERWACJI\_O\_TAKIM\_ID;  
 **END IF**;  
 **EXCEPTION  
 WHEN** NIE\_MA\_REZERWACJI\_O\_TAKIM\_ID **THEN** *RAISE\_APPLICATION\_ERROR*(-20005,**'Nie ma rezerwacji o podanym ID'**);  
 **END**;

*Nowy typ rekordowy i poniżej tabelowy::*

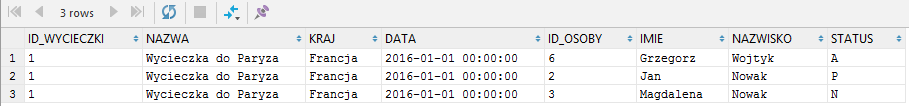
**CREATE OR REPLACE TYPE** wycieczka **AS OBJECT** (  
 ID\_WYCIECZKI **INT**,  
 KRAJ **VARCHAR2**(50),  
 **DATA DATE**,  
 NAZWA **VARCHAR2**(100),  
 LICZBA\_MIEJSC **INT**,  
 LICZBA\_WOLNYCH\_MIEJSC **INT**);

**CREATE OR REPLACE TYPE** tabela\_wycieczek **AS TABLE OF** wycieczka;

**CREATE OR REPLACE FUNCTION** *dostępne\_wycieczki\_f* (v\_kraj **IN** dostępne\_wycieczki.**KRAJ**%**TYPE**, data\_od **IN** dostępne\_wycieczki.**DATA**%**TYPE**, data\_do **IN** dostępne\_wycieczki.**DATA**%**TYPE**)  
 **RETURN** tabela\_wycieczek **PIPELINED  
 AS  
 CURSOR** cur\_dostepne\_wycieczki **is  
 SELECT** *\** **FROM** dostępne\_wycieczki w  
 **WHERE** w.**kraj** = v\_kraj **and** w.**DATA** > data\_od **and** w.**DATA** < data\_do;  
 rekord wycieczka := wycieczka(**NULL**, **NULL**, **NULL**, **NULL**, **NULL**, **NULL**);  
 **BEGIN  
 OPEN** cur\_dostepne\_wycieczki;  
 **LOOP  
 FETCH** cur\_dostepne\_wycieczki **INTO** rekord.ID\_WYCIECZKI, rekord.KRAJ, rekord.DATA, rekord.NAZWA, rekord.LICZBA\_MIEJSC,  
 rekord.LICZBA\_WOLNYCH\_MIEJSC;  
 **EXIT WHEN** cur\_dostepne\_wycieczki%**NOTFOUND**;  
 **PIPE ROW** (rekord);  
 **END LOOP**;  
 **CLOSE** cur\_dostepne\_wycieczki;  
 **END**;

*Przykładowe wywołania funkcji:*

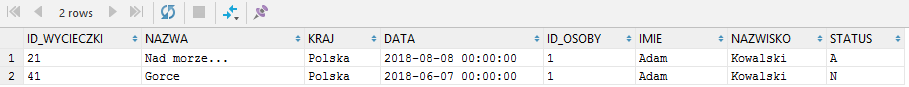
**select** *\** **from table**(*UCZESTNICY\_WYCIECZKI*(1));



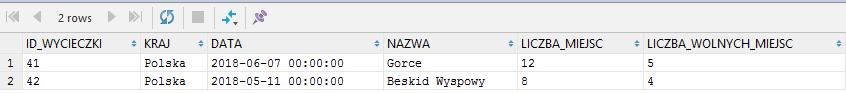
**select** *\** **from table**(*REZERWACJE\_OSOBY*(100)); (błędne dane)



**select** *\** **from table**(*PRZYSZLE\_REZERWACJE\_OSOBY*(1));



**select** *\** **from table**(*DOSTĘPNE\_WYCIECZKI\_F*(**'Polska'**, **'2018-01-01'**, **'2018-09-09'**));



Zad. 5

*Następnie stworzyłam procedury modyfikujące dane.*

*Procedury od razu zawierają wprowadzanie do tabeli REZERWACJE\_LOG, której stworzenie jest wymagane w pkt. 6. Tam też znajduje się kod tworzący tę tabelę.*

**CREATE OR REPLACE PROCEDURE** *dodaj\_rezerwacje* (wycieczka\_id **IN** WYCIECZKI.**ID\_WYCIECZKI**%**TYPE**, osoba\_id **IN** OSOBY.**ID\_OSOBY**%**TYPE**)  
 **AS** v\_miejsca wycieczki\_miejsca.liczba\_wolnych\_miejsc%**TYPE**;  
 v\_data WYCIECZKI.**data**%**TYPE**;  
 v\_rez\_id REZERWACJE.**NR\_REZERWACJI**%**TYPE**;  
 WYCIECZKA\_SIE\_ODBYLA **EXCEPTION**;  
 NIE\_MA\_WOLNYCH\_MIEJSC **EXCEPTION**;  
  
 **BEGIN** *czy\_istnieje\_taka\_osoba*(osoba\_id);  
 *czy\_istnieje\_taka\_wycieczka*(wycieczka\_id);  
  
 **SELECT** wycieczki\_miejsca.LICZBA\_WOLNYCH\_MIEJSC  
 **INTO** v\_miejsca **FROM** wycieczki\_miejsca  
 **WHERE** wycieczki\_miejsca.**ID\_WYCIECZKI** = wycieczka\_id;  
 **IF**(v\_miejsca = 0) **THEN  
 RAISE** NIE\_MA\_WOLNYCH\_MIEJSC;  
 **END IF**;  
 **SELECT** wycieczki.**DATA  
 INTO** v\_data **FROM** wycieczki  
 **WHERE** WYCIECZKI.**ID\_WYCIECZKI** = wycieczka\_id;  
 **IF**(v\_data < *CURRENT\_DATE*) **THEN  
 RAISE** WYCIECZKA\_SIE\_ODBYLA;  
 **END IF**;  
  
 **INSERT INTO** rezerwacje (**id\_wycieczki**, **id\_osoby**, **status**)**VALUES** (wycieczka\_id,osoba\_id,**'N'**);  
  
 **SELECT** REZERWACJE.**NR\_REZERWACJI into** v\_rez\_id **from** REZERWACJE  
 **WHERE** (REZERWACJE.**ID\_WYCIECZKI** =wycieczka\_id **and** REZERWACJE.**ID\_OSOBY** = osoba\_id

**AND** REZERWACJE.**STATUS** = **'N'**);  
  
 **INSERT INTO** REZERWACJE\_LOG (**NR\_REZERWACJI**, **DATA**, **STATUS**)**VALUES** (v\_rez\_id,*CURRENT\_DATE*,**'N'**);  
 **EXCEPTION  
 WHEN** NIE\_MA\_WOLNYCH\_MIEJSC **THEN** *RAISE\_APPLICATION\_ERROR*(-20003,**'Brak wolnych miejsc'**);  
 **WHEN** WYCIECZKA\_SIE\_ODBYLA **THEN** *RAISE\_APPLICATION\_ERROR*(-20004,**'Wycieczka się już odbyła'**);  
 **END**;

*Procedura zmień status rezerwacji sprawdza czy możliwa jest dana zmiana statusu rezerwacji i dopuszcza w zależności od rodzaju zmiany stosując zasady opisane w tabelce poniżej:*

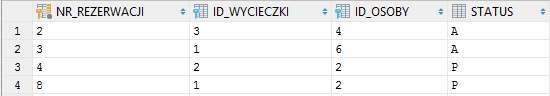
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **na taką** | | | |
|  |  | **A** | **N** | **P** | **Z** |
| **z takiej** | **A** | X | Spr1 | Spr1 | Spr1 |
| **N** | Można | X | Spr2 | Spr2 |
| **P** | Można | Można | X | Spr2 |
| **Z** | Można | Nie | Nie | X |

|  |  |
| --- | --- |
| X | *brak zmiany* |
| Spr1 | *sprawdzenie czy nie jest za późno (data) i czy są jeszcze wolne miejsca* |
| Spr2 | *sprawdzenie czy nie jest za późno (data)* |
| Można | *można dokonać takiej zmiany statusu* |
| Nie | *nie można dokonać takiej zmiany statusu* |

**CREATE OR REPLACE PROCEDURE** *zmien\_status\_rezerwacji* (rezerwacja\_id **IN** REZERWACJE.**NR\_REZERWACJI**%**TYPE**, s **IN** REZERWACJE.**STATUS**%**TYPE**)  
 **AS** v\_status REZERWACJE.**STATUS**%**TYPE**;  
 BRAK\_ZMIANY **EXCEPTION**;  
 BYLA\_ZAPLACONA **EXCEPTION**;  
 **BEGIN** *czy\_istnieje\_taka\_rezerwacja*(rezerwacja\_id);  
 **select** REZERWACJE.**STATUS into** v\_status **FROM** REZERWACJE  
 **where** REZERWACJE.**NR\_REZERWACJI** = rezerwacja\_id;  
 **if**(v\_status = s) **then  
 RAISE** BRAK\_ZMIANY;  
 **end if**;  
 **CASE** v\_status  
 **WHEN 'A' then** *--sprawdz czy mozna nadal, trzeba sprawdzic miejscawolne i date  
 czy\_wycieczka\_sie\_juz\_odbyla\_i\_czy\_sa\_miejsca*(rezerwacja\_id);  
 **WHEN 'N' THEN  
 if**(s = **'P'**) **or** (s = **'Z'**) **THEN** *--czy można, nie trzeba sprawdzać miejsc, wystarczy sprawdzić datę  
 czy\_wycieczka\_sie\_juz\_odbyla*(rezerwacja\_id);  
 **end if**;  
 **WHEN 'P' THEN  
 if**(s = **'Z'**) **then** *--czy dalej można, sprawdzamy tylko datę  
 czy\_wycieczka\_sie\_juz\_odbyla*(rezerwacja\_id);  
 **end if**;  
 **WHEN 'Z' THEN  
 if**(s = **'N'**) **or** (s = **'P'**) **THEN  
 RAISE** BYLA\_ZAPLACONA;  
 **end if**;  
 **end Case**;  
 **UPDATE** REZERWACJE **set STATUS** = s **WHERE** REZERWACJE.**NR\_REZERWACJI** = rezerwacja\_id;  
 **INSERT INTO** REZERWACJE\_LOG (**NR\_REZERWACJI**, **DATA**, **STATUS**)**VALUES** (rezerwacja\_id,*CURRENT\_DATE*,s);

**EXCEPTION  
 WHEN** BRAK\_ZMIANY **THEN** *RAISE\_APPLICATION\_ERROR*(-20006,**'Rezerwacja ma obecnie taki status'**);  
 **WHEN** BYLA\_ZAPLACONA **THEN** *RAISE\_APPLICATION\_ERROR*(-20007,**'Rezerwacja opłacona'**);  
 **END**;  
  
*Przykładowe użycie:*

*Dla rezerwacji o nr 8.*

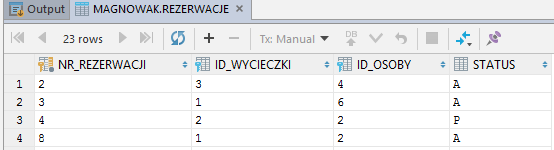
*Przed*  


**BEGIN** *zmien\_status\_rezerwacji*(8,**'P'**);   
**END**;



**BEGIN** *zmien\_status\_rezerwacji*(8,**'A'**);  
**END**;

*Po:*



**CREATE OR REPLACE PROCEDURE** *czy\_wycieczka\_sie\_juz\_odbyla* (rezerwacja\_id **IN** REZERWACJE.**NR\_REZERWACJI**%**TYPE**)  
 **AS** v\_date WYCIECZKI.**DATA**%**TYPE**;  
 v\_wycieczka REZERWACJE.**ID\_WYCIECZKI**%**TYPE**;  
 **BEGIN  
 select** REZERWACJE.**ID\_WYCIECZKI into** v\_wycieczka **from** REZERWACJE  
 **where** REZERWACJE.**NR\_REZERWACJI** = rezerwacja\_id;  
 **select** WYCIECZKI.**DATA into** v\_date **from** WYCIECZKI  
 **where** WYCIECZKI.**ID\_WYCIECZKI**= v\_wycieczka;  
 **if**(v\_date< *CURRENT\_DATE*) **then** *RAISE\_APPLICATION\_ERROR*(-20008,**'Wycieczka się już odbyła'**);  
 **end if**;  
 **END**;

**CREATE OR REPLACE PROCEDURE** *czy\_wycieczka\_sie\_juz\_odbyla\_i\_czy\_sa\_miejsca* (rezerwacja\_id **IN** REZERWACJE.**NR\_REZERWACJI**%**TYPE**)  
 **AS** v\_ilosc **NUMBER**;  
 v\_wycieczka REZERWACJE.**ID\_WYCIECZKI**%**TYPE**;  
 **BEGIN** *czy\_wycieczka\_sie\_juz\_odbyla*(rezerwacja\_id);  
  
 **select** REZERWACJE.**ID\_WYCIECZKI into** v\_wycieczka **from** REZERWACJE  
 **where** REZERWACJE.**NR\_REZERWACJI** = rezerwacja\_id;  
  
 **select** *COUNT*(*\**) **into** v\_ilosc **from** dostępne\_wycieczki  
 **where** DOSTĘPNE\_WYCIECZKI.**ID\_WYCIECZKI**= v\_wycieczka;  
  
 **if**(v\_ilosc = 0) **then** *RAISE\_APPLICATION\_ERROR*(-20009,**'Brak wolnych miejsc'**);  
 **end if**;  
 **END**;

**CREATE OR REPLACE PROCEDURE** *zmien\_liczbe\_miejsc* (wycieczka\_id **IN** WYCIECZKI.**ID\_WYCIECZKI**%**TYPE**, nowa\_l\_miejsc **IN** WYCIECZKI.**LICZBA\_MIEJSC**%**TYPE**)  
 **AS** v\_ilosc\_rezerwacji **NUMBER**;  
  
 **BEGIN** *czy\_istnieje\_taka\_wycieczka*(wycieczka\_id);  
 *czy\_wycieczka\_sie\_juz\_odbyla*(wycieczka\_id);

**SELECT** *COUNT* (*\**) **INTO** V\_ILOSC\_REZERWACJI **FROM** REZERWACJE  
 **WHERE** REZERWACJE.**ID\_WYCIECZKI** = wycieczka\_id;  
  
 **IF**(v\_ilosc\_rezerwacji > nowa\_l\_miejsc) **THEN** *RAISE\_APPLICATION\_ERROR*(-20010, **'Nie można zmniejszyć ilości miejsc, ponieważ obecnie jest więcej rezerwacji'**);  
 **END IF**;  
 **UPDATE** WYCIECZKI **set LICZBA\_MIEJSC** = nowa\_l\_miejsc

**WHERE** WYCIECZKI.**ID\_WYCIECZKI** = wycieczka\_id;  
 **END**;

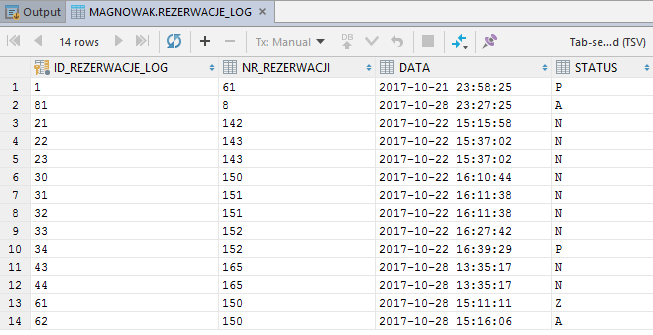
Zad. 6

*Dodanie tabeli dziennikującej zmiany:*

**CREATE TABLE** REZERWACJE\_LOG  
(  
 **ID\_REZERWACJE\_LOG INT GENERATED ALWAYS AS IDENTITY NOT NULL** ,  
 **NR\_REZERWACJI NUMBER**,  
 **DATA DATE** ,  
 **STATUS CHAR**(1) ,  
 **CONSTRAINT** REZERWACJE\_LOG\_PK **PRIMARY KEY** (  
 **ID\_REZERWACJE\_LOG** )  
 **ENABLE**);

*Procedury modyfikujące w pkt.5 uwzględniają już istnienie tej nowej tabeli. Zmiany jakie należało wprowadzić sprowadzały się do dodania 2 linijek do funkcji modyfikujących dane, które wstawiały odpowiedni rekord do tabeli REZERWACJE\_LOG.*

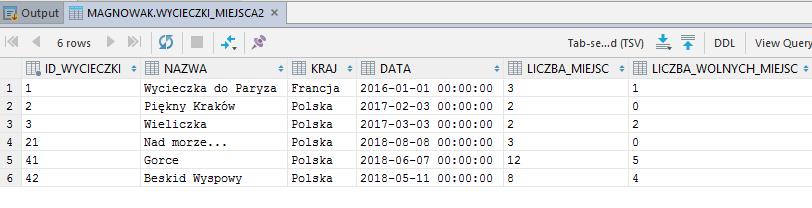
**SELECT** *\** **from** REZERWACJE\_LOG;



Zad. 7

*Modyfikacja widoków. W mojej implementacji wystarczyło zmienić tylko poniższe widoki:*

**CREATE OR REPLACE VIEW** wycieczki\_miejsca2  
 **AS  
 SELECT** w.**ID\_WYCIECZKI**, w.**NAZWA**, w.**KRAJ**, w.**DATA**, w.**LICZBA\_MIEJSC**, w.**LICZBA\_WOLNYCH\_MIEJSC  
 FROM** wycieczki w;

*--widok zwraca tylko te wycieczki, na które są jeszcze miejsca i jeszcze się nie odbyły***CREATE OR REPLACE VIEW** dostępne\_wycieczki2  
 **AS  
 SELECT** *\** **FROM** WYCIECZKI w  
 **WHERE** w.**LICZBA\_WOLNYCH\_MIEJSC** > 0 **and** w.**DATA** > *CURRENT\_DATE*;  
  


*Procedura przeliczająca wartość dla pola LICZBA\_WOLNYCH\_MIEJSC, korzysta z widoku wycieczki\_miejsca, który jest dynamicznie obliczany, więc wartość jaka znajduje się w polu LICZBA\_WOLNYCH\_MIEJSC w tym widoku jest zawsze poprawna, dlatego wystarczy ją tylko przepisać do nowego pola w tabeli WYCIECZKI.*

**CREATE OR REPLACE PROCEDURE** *przelicz* **AS  
 CURSOR** cur\_wycieczki\_miejsca **is  
 select** *\** **from** wycieczki\_miejsca;  
 **BEGIN  
 FOR** i **in** cur\_wycieczki\_miejsca **LOOP  
 UPDATE** WYCIECZKI wy **set** wy.**LICZBA\_WOLNYCH\_MIEJSC** = i.LICZBA\_WOLNYCH\_MIEJSC  
 **where** wy.**ID\_WYCIECZKI** = i.**ID\_WYCIECZKI**;  
 **END LOOP**;  
 **END**;

*Ta procedura jest analogiczna do procedury powyżej, tylko oblicza tę wartość dla jednego rekordu wycieczki o podanym ID, a nie dla wszystkich.*

**CREATE PROCEDURE** *przelicz2* (wycieczka\_id **IN** WYCIECZKI.**ID\_WYCIECZKI**%**TYPE**)  
 **AS** v\_wycieczka wycieczki\_miejsca%**ROWTYPE**;  
 **BEGIN  
 select** *\** **into** v\_wycieczka **from** wycieczki\_miejsca  
 **where** wycieczki\_miejsca.**ID\_WYCIECZKI** = wycieczka\_id;  
 **UPDATE** WYCIECZKI wy **set** wy.**LICZBA\_WOLNYCH\_MIEJSC** = v\_wycieczka.LICZBA\_WOLNYCH\_MIEJSC  
 **where** wy.**ID\_WYCIECZKI** = v\_wycieczka.**ID\_WYCIECZKI**;  
 **END**;

*Modyfikacja procedur, tak aby aktualizowały wartość pola LICZBA\_WOLNYCH\_MIEJSC z tabeli WYCIECZKI.  
  
W procedurach pobierających dane korzystałam z widoków, które dynamicznie wyliczają liczbę wolnych miejsc, dlatego nie trzeba było w nich nic zmieniać.  
  
Poprawki wprowadzone do procedur wprowadzających dane:*

*Aby nie przeklejać kodu, wywołałam już działającą funkcję dodaj rezerwacje, a następnie procedurę wcześniej napisaną przelicz2, która przeliczała od nowa wartość pola LICZBA\_WOLNYCH\_MIEJSC.*

*Możliwe też było użycie kodu:*

*UPDATE WYCIECZKI set LICZBA\_WOLNYCH\_MIEJSC = (LICZBA\_WOLNYCH\_MIEJSC -1)  
 where WYCIECZKI.ID\_WYCIECZKI = wycieczka\_id;  
Ale wtedy gdyby gdzieś wcześniej był błąd przy jakiejś transakcji (chociaż nie powinien po żadnej funkcji, ale być może wystąpiłby po jakiś nowych zmianach) i wartość w tym polu byłaby niepoprawna, błąd byłby propagowany. Wyliczenie tej wartości od nowa jest bezpieczniejsze. Z kolei przeliczanie dla całej tabeli trwało by zbyt długo, stąd nowa procedura przelicz2, która jest kompromisem pomiędzy tymi dwoma podejściami.*

**CREATE OR REPLACE PROCEDURE** *dodaj\_rezerwacje2* (wycieczka\_id **IN** WYCIECZKI.**ID\_WYCIECZKI**%**TYPE**, osoba\_id **IN** OSOBY.**ID\_OSOBY**%**TYPE**)  
 **AS  
  
 BEGIN** *dodaj\_rezerwacje*(wycieczka\_id,osoba\_id);  
  *przelicz2*(wycieczka\_id);  
 **END**;

**CREATE OR REPLACE PROCEDURE** *zmien\_status\_rezerwacji2* (rezerwacja\_id **IN** REZERWACJE.**NR\_REZERWACJI**%**TYPE**, s **IN** REZERWACJE.**STATUS**%**TYPE**)  
 **AS** v\_status REZERWACJE.**STATUS**%**TYPE**;  
 BRAK\_ZMIANY **EXCEPTION**;  
 BYLA\_ZAPLACONA **EXCEPTION**;  
 **BEGIN** *zmien\_status\_rezerwacji*(rezerwacja\_id,s);  
 *przelicz*();  
  
 **EXCEPTION  
 WHEN** BRAK\_ZMIANY **THEN** *RAISE\_APPLICATION\_ERROR*(-20006,**'Rezerwacja ma obecnie taki status'**);  
 **WHEN** BYLA\_ZAPLACONA **THEN** *RAISE\_APPLICATION\_ERROR*(-20007,**'Rezerwacja opłacona'**);  
 **END**;  
**CREATE OR REPLACE PROCEDURE** *zmien\_liczbe\_miejsc2* (wycieczka\_id **IN** WYCIECZKI.**ID\_WYCIECZKI**%**TYPE**, nowa\_l\_miejsc **IN** WYCIECZKI.**LICZBA\_MIEJSC**%**TYPE**)  
 **AS  
 BEGIN** *zmien\_liczbe\_miejsc*(wycieczka\_id, nowa\_l\_miejsc);  
 *przelicz2*(wycieczka\_id);  
 **END**;

Zad. 8

*Zmiana procedur modyfikujących dane, od razu jest to kod tych procedur w przypadku, gdy aktywne są już triggery z pkt. 8 oraz z pkt. 9.*

*Zatem:*  
*--usunięto fragment kodu wpisujący rekord do tabeli rezewacje\_log*

*--usunięto wywołania procedur przelicz/przelicz2*

**CREATE OR REPLACE PROCEDURE** *dodaj\_rezerwacje3* (wycieczka\_id **IN** WYCIECZKI.**ID\_WYCIECZKI**%**TYPE**, osoba\_id **IN** OSOBY.**ID\_OSOBY**%**TYPE**)  
 **AS** v\_miejsca wycieczki\_miejsca.liczba\_wolnych\_miejsc%**TYPE**;  
 v\_data WYCIECZKI.**data**%**TYPE**;  
 v\_rez\_id REZERWACJE.**NR\_REZERWACJI**%**TYPE**;  
 WYCIECZKA\_SIE\_ODBYLA **EXCEPTION**;  
 NIE\_MA\_WOLNYCH\_MIEJSC **EXCEPTION**;  
  
 **BEGIN** *czy\_istnieje\_taka\_osoba*(osoba\_id);  
 *czy\_istnieje\_taka\_wycieczka*(wycieczka\_id);  
 **SELECT** wycieczki\_miejsca.LICZBA\_WOLNYCH\_MIEJSC  
 **INTO** v\_miejsca **FROM** wycieczki\_miejsca  
 **WHERE** wycieczki\_miejsca.**ID\_WYCIECZKI** = wycieczka\_id;  
 **IF**(v\_miejsca = 0) **THEN  
 RAISE** NIE\_MA\_WOLNYCH\_MIEJSC;  
 **END IF**;  
 **SELECT** wycieczki.**DATA  
 INTO** v\_data **FROM** wycieczki  
 **WHERE** WYCIECZKI.**ID\_WYCIECZKI** = wycieczka\_id;

**IF**(v\_data < *CURRENT\_DATE*) **THEN  
 RAISE** WYCIECZKA\_SIE\_ODBYLA;  
 **END IF**;  
 **INSERT INTO** REZERWACJE (**ID\_WYCIECZKI**, **ID\_OSOBY**, **STATUS**)**VALUES** (WYCIECZKA\_ID,OSOBA\_ID,**'N'**);  
 **EXCEPTION  
 WHEN** NIE\_MA\_WOLNYCH\_MIEJSC **THEN** *RAISE\_APPLICATION\_ERROR*(-20003,**'Brak wolnych miejsc'**);  
 **WHEN** WYCIECZKA\_SIE\_ODBYLA **THEN** *RAISE\_APPLICATION\_ERROR*(-20004,**'Wycieczka się już odbyła'**);  
 **END**;

**CREATE OR REPLACE PROCEDURE** *zmien\_status\_rezerwacji3* (rezerwacja\_id **IN** REZERWACJE.**NR\_REZERWACJI**%**TYPE**, s **IN** REZERWACJE.**STATUS**%**TYPE**)  
 **AS** v\_status REZERWACJE.**STATUS**%**TYPE**;  
 BRAK\_ZMIANY **EXCEPTION**;  
 BYLA\_ZAPLACONA **EXCEPTION**;  
 **BEGIN** *czy\_istnieje\_taka\_rezerwacja*(rezerwacja\_id);  
 **select** REZERWACJE.**STATUS into** v\_status **FROM** REZERWACJE  
 **where** REZERWACJE.**NR\_REZERWACJI** = rezerwacja\_id;  
 **if**(v\_status = s) **then  
 RAISE** BRAK\_ZMIANY;  
 **end if**;  
 **CASE** v\_status  
 **WHEN 'A' then** *czy\_wycieczka\_sie\_juz\_odbyla\_i\_czy\_sa\_miejsca*(rezerwacja\_id);  
 **WHEN 'N' THEN  
 if**(s = **'P'**) **or** (s = **'Z'**) **THEN** *czy\_wycieczka\_sie\_juz\_odbyla*(rezerwacja\_id);  
 **end if**;  
 **WHEN 'P' THEN  
 if**(s = **'Z'**) **then** *czy\_wycieczka\_sie\_juz\_odbyla*(rezerwacja\_id);  
 **end if**;  
 **WHEN 'Z' THEN  
 if**(s = **'N'**) **or** (s = **'P'**) **THEN  
 RAISE** BYLA\_ZAPLACONA;  
 **end if**;  
 **end Case**;  
 **UPDATE** REZERWACJE **set STATUS** = s **WHERE** REZERWACJE.**NR\_REZERWACJI** = rezerwacja\_id;  
 **EXCEPTION  
 WHEN** BRAK\_ZMIANY **THEN** *RAISE\_APPLICATION\_ERROR*(-20006,**'Rezerwacja ma obecnie taki status'**);  
 **WHEN** BYLA\_ZAPLACONA **THEN** *RAISE\_APPLICATION\_ERROR*(-20007,**'Rezerwacja opłacona'**);  
 **END**;

**CREATE OR REPLACE PROCEDURE** *zmien\_liczbe\_miejsc3*

(wycieczka\_id **IN** WYCIECZKI.**ID\_WYCIECZKI**%**TYPE**, nowa\_l\_miejsc **IN** WYCIECZKI.**LICZBA\_MIEJSC**%**TYPE**)  
**AS** v\_ilosc\_rezerwacji **NUMBER**;  
  
**BEGIN** *czy\_istnieje\_taka\_wycieczka*(wycieczka\_id);  
 *czy\_wycieczka\_sie\_juz\_odbyla*(wycieczka\_id);  
 **select** *count* (*\**) **into** v\_ilosc\_rezerwacji **from** REZERWACJE  
 **where** REZERWACJE.**ID\_WYCIECZKI** = wycieczka\_id;  
 **if**(v\_ilosc\_rezerwacji > nowa\_l\_miejsc) **THEN** *RAISE\_APPLICATION\_ERROR*(-20010, **'Nie można zmniejszyć ilości miejsc, ponieważ obecnie jest więcej rezerwacji'**);  
 **end if**;  
 **UPDATE** WYCIECZKI **set LICZBA\_MIEJSC** = nowa\_l\_miejsc **WHERE** WYCIECZKI.**ID\_WYCIECZKI** = wycieczka\_id;  
**END**;

*Oto tiggery dzięki, którym nie trzeba dodawać do procedur/funkcji linijek wstawiających rekordy do REZERWACJE\_LOG, dodatkowo jeśli będzie się wykonywać zwykły insert bez procedury (co jest jednak nie wskazane ze względu na brak sprawdzania poprawności argumentów), to trigger także zadziała.*

*Trigger 1*

**CREATE OR REPLACE TRIGGER** trigger\_przy\_dodawaniu\_rezerwacji  
 **AFTER INSERT ON** REZERWACJE  
 **FOR EACH ROW  
 BEGIN  
 INSERT INTO** REZERWACJE\_LOG(**NR\_REZERWACJI**, **DATA**, **STATUS**) **VALUES**(:**NEW**.**NR\_REZERWACJI**,*current\_date*,:**NEW**.**STATUS**);  
 **END**;

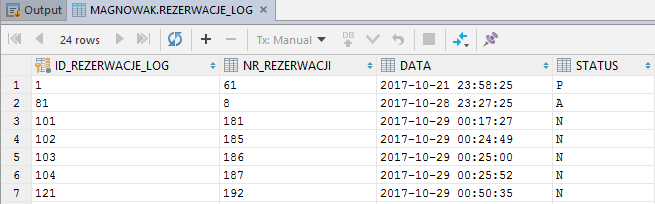
*Trigger 2***CREATE OR REPLACE TRIGGER** triger\_przy\_zmianie\_statusu\_rezerwacji  
 **after update on** REZERWACJE  
 **FOR EACH ROW  
 BEGIN  
 INSERT into** REZERWACJE\_LOG(**NR\_REZERWACJI**, **DATA**, **STATUS**)

**values**(:**OLD**.**NR\_REZERWACJI**,*current\_date*,:**NEW**.**STATUS**);  
 **END**;  
  
*Trigger 3*

**CREATE OR REPLACE TRIGGER** triger\_zabraniający\_usunięcia\_rezerwacji  
 **before delete on** REZERWACJE  
 **FOR EACH ROW  
 BEGIN** *RAISE\_APPLICATION\_ERROR*(-20011,**'Nie można usuwać rezerwacji, możesz zmienić status na Anulowana'**);  
 **END**;

*Działanie triggera 2 (dla rezerwacji o numerze 61):*

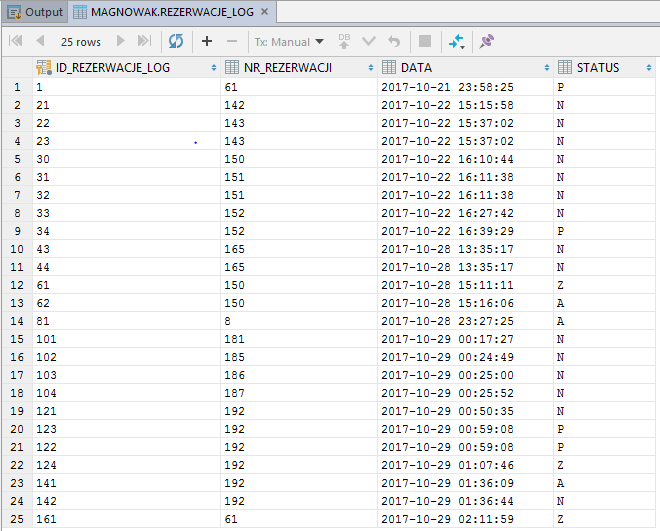
*Przed:*



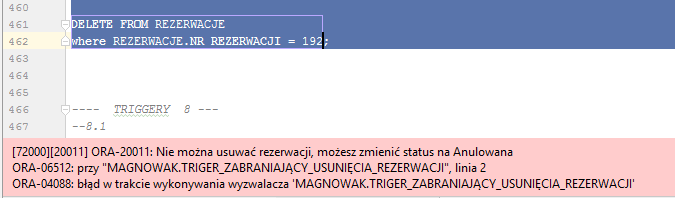
**BEGIN** *zmien\_status\_rezerwacji3*(61,**'Z'**);  
**END**;

*Po:*

**SELECT** *\** **from** REZERWACJE\_LOG  
**order by DATA**;



*Działanie Triggera 3:*



Zad. 9

*Stworzenie triggerów w mojej implementacji bazy obsługujących redundantne pole LICZBA\_WOLNYCH\_MIEJSC sprowadzało się do opakowania w triggery procedur, które miałam już wcześniej napisane.*

*Trigger 1*

**CREATE OR REPLACE TRIGGER** triger\_przy\_dodawaniu\_rezerwacji2  
 **before insert on** REZERWACJE  
 **FOR EACH ROW  
 BEGIN  
 UPDATE** WYCIECZKI wy **set** wy.**LICZBA\_WOLNYCH\_MIEJSC** = wy.**LICZBA\_WOLNYCH\_MIEJSC** - 1  
 **where** wy.**ID\_WYCIECZKI** = :**NEW**.**ID\_WYCIECZKI**;

**END**;

*Trigger 2*

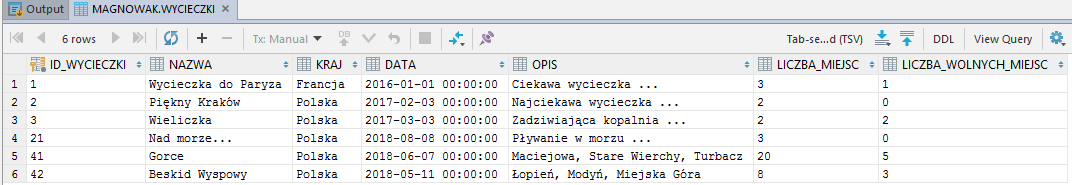
**CREATE OR REPLACE TRIGGER** triger\_przy\_zmianie\_statusu\_rezerwacji2  
 **before update on** REZERWACJE  
 **FOR EACH ROW  
BEGIN  
 IF**(:**NEW**.**STATUS** = **'A'**) **THEN  
 UPDATE** WYCIECZKI wy **set** wy.**LICZBA\_WOLNYCH\_MIEJSC** = wy.**LICZBA\_WOLNYCH\_MIEJSC** + 1  
 **where** wy.**ID\_WYCIECZKI** = :**NEW**.**ID\_WYCIECZKI**;  
 **ELSE  
 IF**(:**OLD**.**STATUS** = **'A'**) **THEN  
 UPDATE** WYCIECZKI wy **set** wy.**LICZBA\_WOLNYCH\_MIEJSC** = wy.**LICZBA\_WOLNYCH\_MIEJSC** - 1  
 **where** wy.**ID\_WYCIECZKI** = :**NEW**.**ID\_WYCIECZKI**;  
 **END IF**;  
**END IF**;

**END**;  
 *Trigger 3*

**CREATE OR REPLACE TRIGGER** triger\_po\_zmianie\_liczby\_miejsc\_dla\_wycieczki  
 **before update on** WYCIECZKI  
 **referencing OLD as old NEW as new  
 FOR EACH ROW  
 BEGIN** :**new**.**LICZBA\_WOLNYCH\_MIEJSC** :=  
 :**old**.**LICZBA\_WOLNYCH\_MIEJSC** + :**new**.**LICZBA\_MIEJSC** - :**old**.**LICZBA\_MIEJSC**;  
 **END**;

*Działanie Triggera 3 (dla wycieczki o id = 41):*

*Przed:*



**BEGIN  
 update** wycieczki **set LICZBA\_MIEJSC** = 25 **where** WYCIECZKI.**ID\_WYCIECZKI** =41;  
**END**;

*Po:*

