

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 Paryża','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;

ID_W...	NAZWA	KRAJ	DATA	OPIS	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC
1	Wycieczka do Paryża	Francja	2016-01-01 00:00:00	Ciekawa wycieczka ...	3	1
2	Piękny Kraków	Polska	2017-02-03 00:00:00	Najciekawa wycieczka ...	2	0
3	Wieliczka	Polska	2017-03-03 00:00:00	Zadziwiająca kopalnia ...	2	2
4	Nad morze...	Polska	2018-08-08 00:00:00	Pływanie w morzu ...	3	0
5	Gorce	Polska	2018-06-07 00:00:00	Maciejowa, Stare Wierchy, Turbacz	12	5
6	Beskid Wyspowy	Polska	2018-05-11 00:00:00	Łopień, Modyń, Miejska Góra	8	4

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');

INSERT 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');

Output		MAGNOWAK.OSOBY x				
10 rows						
	ID_OSOBY	IMIE	NAZWISKO	PESEL	KONTAKT	
1	1	Adam	Kowalski	87654321	tel: 6623	
2	2	Jan	Nowak	12345678	tel: 2312, dzwonić po 18.00	
3	3	Magdalena	Nowak	12123896	tel: 1234	
4	4	Kinga	Nowak	1989843	tel: 0987	
5	5	Justyna	Michalec	121298376	tel: 1247	
6	6	Grzegorz	Wojtyk	87923896	tel: 4455	
7	7	Krzysztof	Michalec	973487896	tel: 1263	
8	8	Bartosz	Kamis	632434596	tel: 7823	
9	9	Joanna	Ptys	13245696	tel: 2355	
10	10	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');

Output		MAGNOWAK.REZERWACJE		
24 rows		Tx: Manual		
	NR_REZERWACJI	ID_WYCIECZKI	ID_OSOBY	STATUS
1	2	3	4	A
2	3	1	6	A
3	4	2	2	P
4	192	42	1	N
5	8	1	2	A
6	9	1	3	N
7	11	3	10	A
8	22	21	1	A
9	23	21	9	P
10	41	21	2	P
11	42	21	5	N
12	61	41	10	Z
13	101	42	3	N
14	81	2	10	N
15	121	42	4	N
16	122	42	5	N
17	123	42	6	N
18	141	41	9	N
19	142	41	1	N
20	143	41	2	N
21	150	41	3	A
22	151	41	4	N
23	152	41	6	P
24	165	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;
```

ID_WYCIECZKI	NAZWA	KRAJ	DATA	ID_OSOBY	IMIE	NAZWISKO	STATUS
1	Wycieczka do Paryza	Francja	2016-01-01 00:00:00	2	Jan	Nowak	A
2	Wycieczka do Paryza	Francja	2016-01-01 00:00:00	3	Magdalena	Nowak	N
3	Wycieczka do Paryza	Francja	2016-01-01 00:00:00	6	Grzegorz	Wojtyk	A
4	Piękny Kraków	Polska	2017-02-03 00:00:00	2	Jan	Nowak	P
5	Piękny Kraków	Polska	2017-02-03 00:00:00	10	Marek	Jonatowicz	N
6	Wieliczka	Polska	2017-03-03 00:00:00	4	Kinga	Nowak	A
7	Wieliczka	Polska	2017-03-03 00:00:00	10	Marek	Jonatowicz	A
8	Nad morze...	Polska	2018-08-08 00:00:00	1	Adam	Kowalski	A
9	Nad morze...	Polska	2018-08-08 00:00:00	2	Jan	Nowak	P
10	Nad morze...	Polska	2018-08-08 00:00:00	5	Justyna	Michalec	N
11	Nad morze...	Polska	2018-08-08 00:00:00	9	Joanna	Ptys	P
12	Gorce	Polska	2018-06-07 00:00:00	1	Adam	Kowalski	N
13	Gorce	Polska	2018-06-07 00:00:00	2	Jan	Nowak	N
14	Gorce	Polska	2018-06-07 00:00:00	3	Magdalena	Nowak	A
15	Gorce	Polska	2018-06-07 00:00:00	4	Kinga	Nowak	N
16	Gorce	Polska	2018-06-07 00:00:00	5	Justyna	Michalec	N
17	Gorce	Polska	2018-06-07 00:00:00	6	Grzegorz	Wojtyk	P
18	Gorce	Polska	2018-06-07 00:00:00	9	Joanna	Ptys	N
19	Gorce	Polska	2018-06-07 00:00:00	10	Marek	Jonatowicz	Z
20	Beskid Wyspowy	Polska	2018-05-11 00:00:00	1	Adam	Kowalski	N
21	Beskid Wyspowy	Polska	2018-05-11 00:00:00	3	Magdalena	Nowak	N
22	Beskid Wyspowy	Polska	2018-05-11 00:00:00	4	Kinga	Nowak	N
23	Beskid Wyspowy	Polska	2018-05-11 00:00:00	5	Justyna	Michalec	N
24	Beskid Wyspowy	Polska	2018-05-11 00:00:00	6	Grzegorz	Wojtyk	N

```
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';
```

ID_WYCIECZKI	NAZWA	KRAJ	DATA	IMIE	NAZWISKO	STATUS
2	Piękny Kraków	Polska	2017-02-03 00:00:00	Jan	Nowak	P
21	Nad morze...	Polska	2018-08-08 00:00:00	Jan	Nowak	P
21	Nad morze...	Polska	2018-08-08 00:00:00	Joanna	Ptys	P
41	Gorce	Polska	2018-06-07 00:00:00	Grzegorz	Wojtyk	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;
```

Output MAGNOWAK.WYCIECZKI_PRZYSZLE							
ID_WYCIECZKI	NAZWA	KRAJ	DATA	IMIE	NAZWISKO	STATUS	
1 21	Nad morze...	Polska	2018-08-08 00:00:00	Adam	Kowalski	A	
2 21	Nad morze...	Polska	2018-08-08 00:00:00	Jan	Nowak	P	
3 21	Nad morze...	Polska	2018-08-08 00:00:00	Justyna	Michalec	N	
4 21	Nad morze...	Polska	2018-08-08 00:00:00	Joanna	Ptys	P	
5 41	Gorce	Polska	2018-06-07 00:00:00	Adam	Kowalski	N	
6 41	Gorce	Polska	2018-06-07 00:00:00	Jan	Nowak	N	
7 41	Gorce	Polska	2018-06-07 00:00:00	Magdalena	Nowak	A	
8 41	Gorce	Polska	2018-06-07 00:00:00	Kinga	Nowak	N	
9 41	Gorce	Polska	2018-06-07 00:00:00	Justyna	Michalec	N	
10 41	Gorce	Polska	2018-06-07 00:00:00	Grzegorz	Wojtyk	P	
11 41	Gorce	Polska	2018-06-07 00:00:00	Joanna	Ptys	N	
12 41	Gorce	Polska	2018-06-07 00:00:00	Marek	Jonatowicz	Z	
13 42	Beskid Wyspowy	Polska	2018-05-11 00:00:00	Adam	Kowalski	N	
14 42	Beskid Wyspowy	Polska	2018-05-11 00:00:00	Magdalena	Nowak	N	
15 42	Beskid Wyspowy	Polska	2018-05-11 00:00:00	Kinga	Nowak	N	
16 42	Beskid Wyspowy	Polska	2018-05-11 00:00:00	Justyna	Michalec	N	
17 42	Beskid Wyspowy	Polska	2018-05-11 00:00:00	Grzegorz	Wojtyk	N	

```

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;

```

Output MAGNOWAK.WYCIECZKI_MIEJSCA						
ID_WYCIECZKI	KRAJ	DATA	NAZWA	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC	
1 42	Polska	2018-05-11 00:00:00	Beskid Wyspowy	8	3	
2 1	Francja	2016-01-01 00:00:00	Wycieczka do Paryza	3	2	
3 2	Polska	2017-02-03 00:00:00	Piękny Kraków	2	0	
4 21	Polska	2018-08-08 00:00:00	Nad morze...	3	0	
5 41	Polska	2018-06-07 00:00:00	Gorce	20	13	
6 3	Polska	2017-03-03 00:00:00	Wieliczka	2	2	

--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;

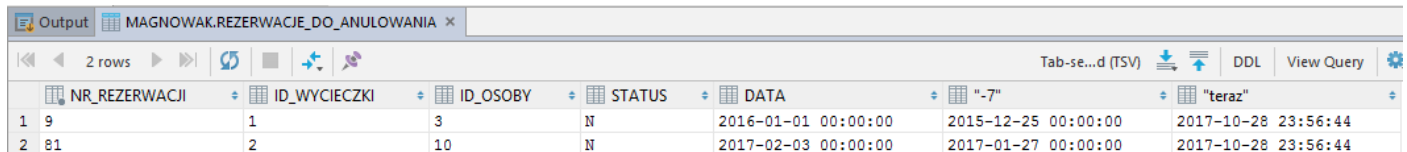
```

Output MAGNOWAK.DOSTĘPNE_WYCIECZKI						
ID_W...	KRAJ	DATA	NAZWA	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC	
1 42	Polska	2018-05-11 00:00:00	Beskid Wyspowy	8	4	
2 41	Polska	2018-06-07 00:00:00	Gorce	12	5	

--widok zwraca te rezerwacje, które są rezerwacjami 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;
```



	NR_REZERWACJI	ID_WYCIECZKI	ID_OSOBY	STATUS	DATA	"-7"	"teraz"
1	9	1	3	N	2016-01-01 00:00:00	2015-12-25 00:00:00	2017-10-28 23:56:44
2	81	2	10	N	2017-02-03 00:00:00	2017-01-27 00:00:00	2017-10-28 23:56:44

Zad. 4

Przygotowałam też zestaw procedur i funkcji ułatwiających dostęp do danych.

We wszystkich 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 składającą 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 dostepne_wycieczki_f  
(v_kraj IN dostepne_wycieczki.KRAJ%TYPE, data_od IN dostepne_wycieczki.DATA%TYPE, data_do IN  
dostepne_wycieczki.DATA%TYPE)  
RETURN tabela_wycieczek PIPELINED  
AS  
CURSOR cur_dostepne_wycieczki is  
  SELECT * FROM dostepne_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(UCZESTNICZY_WYCIECZKI(1));
```

ID_WYCIECZKI	NAZWA	KRAJ	DATA	ID_OSOBY	IMIE	NAZWISKO	STATUS
1	Wycieczka do Paryza	Francja	2016-01-01 00:00:00	6	Grzegorz	Wojtyk	A
2	Wycieczka do Paryza	Francja	2016-01-01 00:00:00	2	Jan	Nowak	P
3	Wycieczka do Paryza	Francja	2016-01-01 00:00:00	3	Magdalena	Nowak	N

```
select * from table(REZERWACJE_OSOBY(100)); (błędne dane)
```

```
[72000][20001] ORA-20001: Nie ma osoby o podanym ID  
ORA-06512: przy "MAGNOWAK.CZY_ISTNIEJE_TAKA_OSOBA", linia 15  
ORA-06512: przy "MAGNOWAK.REZERWACJE_OSOBY", linia 11
```

```
select * from table(PRZYSZLE_REZERWACJE_OSOBY(1));
```

ID_WYCIECZKI	NAZWA	KRAJ	DATA	ID_OSOBY	IMIE	NAZWISKO	STATUS
1	Nad morze...	Polska	2018-08-08 00:00:00	1	Adam	Kowalski	A
2	Gorce	Polska	2018-06-07 00:00:00	1	Adam	Kowalski	N

```
select * from table(DOSTĘPNE_WYCIECZKI_F('Polska', '2018-01-01', '2018-09-09'));
```

	ID_WYCIECZKI	KRAJ	DATA	NAZWA	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC
1	41	Polska	2018-06-07 00:00:00	Gorce	12	5
2	42	Polska	2018-05-11 00:00:00	Beskid Wyspowy	8	4

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_ODBYŁA **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_ODBYŁA;

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_ODBYŁA **THEN**

RAISE_APPLICATION_ERROR(-20004,'Wycieczka się już odbyła');

END;

Procedura zmian statusu 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 mozna, nie trzeba sprawdzac miejsc, wystarczy sprawdzic date
        czy_wycieczka_sie_juz_odbyla(rezerwacja_id);
      end if;
    WHEN 'P' THEN
      if(s = 'Z') then
        --czy dalej mozna, sprawdzamy tylko date
        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

	NR_REZERWACJI	ID_WYCIECZKI	ID_OSOBY	STATUS
1	2	3	4	A
2	3	1	6	A
3	4	2	2	P
4	8	1	2	P

BEGIN

```
zmien_status_rezerwacji(8,'P');
```

END;

```
[72000][20006] ORA-20006: Rezerwacja ma obecnie taki status
ORA-06512: przy "MAGNOWAK.ZMIEN_STATUS_REZERWACJI", linia 39
ORA-06512: przy linia 2
```

BEGIN

```
zmien_status_rezerwacji(8,'A');
```

END;

Po:

	NR_REZERWACJI	ID_WYCIECZKI	ID_OSOBY	STATUS
1	2	3	4	A
2	3	1	6	A
3	4	2	2	P
4	8	1	2	A

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;

Output MAGNOWAK.REZERWACJE_LOG x				
14 rows				
	ID_REZERWACJE_LOG	NR_REZERWACJI	DATA	STATUS
1	1	61	2017-10-21 23:58:25	P
2	81	8	2017-10-28 23:27:25	A
3	21	142	2017-10-22 15:15:58	N
4	22	143	2017-10-22 15:37:02	N
5	23	143	2017-10-22 15:37:02	N
6	30	150	2017-10-22 16:10:44	N
7	31	151	2017-10-22 16:11:38	N
8	32	151	2017-10-22 16:11:38	N
9	33	152	2017-10-22 16:27:42	N
10	34	152	2017-10-22 16:39:29	P
11	43	165	2017-10-28 13:35:17	N
12	44	165	2017-10-28 13:35:17	N
13	61	150	2017-10-28 15:11:11	Z
14	62	150	2017-10-28 15:16:06	A

Zad. 7

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

```
CREATE OR REPLACE VIEW wycieczki_miejsca2
AS
SELECT
    w.ID_WYCIEZKI, 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 WYCIEZKI w
WHERE w.LICZBA_WOLNYCH_MIEJSC > 0 and w.DATA > CURRENT_DATE;
```

Output MAGNOWAK.WYCIEZKI_MIEJSCA2 x						
6 rows						
	ID_WYCIEZKI	NAZWA	KRAJ	DATA	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC
1	1	Wycieczka do Paryża	Francja	2016-01-01 00:00:00	3	1
2	2	Piękny Kraków	Polska	2017-02-03 00:00:00	2	0
3	3	Wieliczka	Polska	2017-03-03 00:00:00	2	2
4	21	Nad morze...	Polska	2018-08-08 00:00:00	3	0
5	41	Gorce	Polska	2018-06-07 00:00:00	12	5
6	42	Beskid Wyspówy	Polska	2018-05-11 00:00:00	8	4

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 (choć 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 triggerzy z pkt. 8 oraz z pkt. 9.

Zatem:

*--usunięto fragment kodu wpisujący rekord do tabeli rezerwacje_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 liniiek 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:

Output MAGNOWAK.REZERWACJE_LOG x				
24 rows				
ID_REZERWACJE_LOG	NR_REZERWACJI	DATA	STATUS	
1	1	2017-10-21 23:58:25	P	
2	81	2017-10-28 23:27:25	A	
3	101	2017-10-29 00:17:27	N	
4	102	2017-10-29 00:24:49	N	
5	103	2017-10-29 00:25:00	N	
6	104	2017-10-29 00:25:52	N	
7	121	2017-10-29 00:50:35	N	

BEGIN

zmien_status_rezerwacji3(61,'Z');

END;

Po:

SELECT * from REZERWACJE_LOG

order by DATA;

Output MAGNOWAK.REZERWACJE_LOG x				
25 rows				
ID_REZERWACJE_LOG	NR_REZERWACJI	DATA	STATUS	
1	1	2017-10-21 23:58:25	P	
2	21	2017-10-22 15:15:58	N	
3	22	2017-10-22 15:37:02	N	
4	23	2017-10-22 15:37:02	N	
5	30	2017-10-22 16:10:44	N	
6	31	2017-10-22 16:11:38	N	
7	32	2017-10-22 16:11:38	N	
8	33	2017-10-22 16:27:42	N	
9	34	2017-10-22 16:39:29	P	
10	43	2017-10-28 13:35:17	N	
11	44	2017-10-28 13:35:17	N	
12	61	2017-10-28 15:11:11	Z	
13	62	2017-10-28 15:16:06	A	
14	81	2017-10-28 23:27:25	A	
15	101	2017-10-29 00:17:27	N	
16	102	2017-10-29 00:24:49	N	
17	103	2017-10-29 00:25:00	N	
18	104	2017-10-29 00:25:52	N	
19	121	2017-10-29 00:50:35	N	
20	123	2017-10-29 00:59:08	P	
21	122	2017-10-29 00:59:08	P	
22	124	2017-10-29 01:07:46	Z	
23	141	2017-10-29 01:36:09	A	
24	142	2017-10-29 01:36:44	N	
25	161	2017-10-29 02:11:59	Z	

Działanie Triggera 3:

```
460
461 DELETE FROM REZERWACJE
462 where REZERWACJE.NR REZERWACJI = 192;
463
464
465
466 ----- TRIGGER 8 -----
467 --8.1
```

```
[72000][20011] ORA-20011: Nie można usuwać rezerwacji, możesz zmienić status na Anulowana
ORA-06512: przy "MAGNOWAK.TRIGER_ZABRANIAJĄCY_USUNIĘCIA_REZERWACJI", linia 2
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'MAGNOWAK.TRIGER_ZABRANIAJĄCY_USUNIĘCIA_REZERWACJI'
```

Zad. 9

Stworzenie triggerów w mojej implementacji bazy obsługujących redundantne pole LICZBA_WOLNYCH_MIEJSC sprowadzało się do opakowania w trigger 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:

Output MAGNOWAK.WYCIECZKI							
Tx: Manual DB							
Tab-se...d (TSV) DDL View Query							
ID_WYCIECZKI	NAZWA	KRAJ	DATA	OPIS	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC	
1	Wycieczka do Paryża	Francja	2016-01-01 00:00:00	Ciekawa wycieczka ...	3	1	
2	Piękny Kraków	Polska	2017-02-03 00:00:00	Najciekawa wycieczka ...	2	0	
3	Wieliczka	Polska	2017-03-03 00:00:00	Zadziwiająca kopalnia ...	2	2	
4	Nad morze...	Polska	2018-08-08 00:00:00	Pływanie w morzu ...	3	0	
5	Gorce	Polska	2018-06-07 00:00:00	Maciejowa, Stare Wierchy, Turbacz	20	5	
6	Beskid Wyspowy	Polska	2018-05-11 00:00:00	Łopień, Modyń, Miejska Góra	8	3	

BEGIN

```
update wycieczki set LICZBA_MIEJSC = 25 where WYCIECZKI.ID_WYCIECZKI =41;  
END;
```

Po:

Output MAGNOWAK.WYCIECZKI							
Tx: Manual DB							
Tab-se...d (TSV) DDL View Query							
ID_WYCIECZKI	NAZWA	KRAJ	DATA	OPIS	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC	
1	Wycieczka do Paryża	Francja	2016-01-01 00:00:00	Ciekawa wycieczka ...	3	1	
2	Piękny Kraków	Polska	2017-02-03 00:00:00	Najciekawa wycieczka ...	2	0	
3	Wieliczka	Polska	2017-03-03 00:00:00	Zadziwiająca kopalnia ...	2	2	
4	Nad morze...	Polska	2018-08-08 00:00:00	Pływanie w morzu ...	3	0	
5	Gorce	Polska	2018-06-07 00:00:00	Maciejowa, Stare Wierchy, Turbacz	25	10	
6	Beskid Wyspowy	Polska	2018-05-11 00:00:00	Łopień, Modyń, Miejska Góra	8	3	