

Laboratorium 1: Oracle PL/SQL

Bartosz Szar

1. Tabele

1.1 Osoby

```
CREATE TABLE OSOBY
(
    ID_OSOBY INT GENERATED ALWAYS AS IDENTITY NOT NULL,
    IMIE VARCHAR2(50),
    NAZWISKO VARCHAR2(50),
    PESEL VARCHAR2(11),
    KONTAKT VARCHAR2(100),
    CONSTRAINT OSOBY_PK PRIMARY KEY
    (
        ID_OSOBY
    )
    ENABLE
);
```

1.2 Wycieczki

```
CREATE TABLE WYCIECZKI
(
    ID_WYCIECZKI INT GENERATED ALWAYS AS IDENTITY NOT NULL,
    NAZWA VARCHAR2(100),
    KRAJ VARCHAR2(50),
    DATA DATE,
    OPIS VARCHAR2(200),
    LICZBA_MIEJSC INT,
    CONSTRAINT WYCIECZKI_PK PRIMARY KEY
    (
        ID_WYCIECZKI
    )
    ENABLE
);
```

1.3 Rezerwacje

```
CREATE TABLE REZERWACJE
(
    NR_REZERWACJI INT GENERATED ALWAYS AS IDENTITY NOT NULL,
    ID_WYCIECZKI INT,
    ID_OSOBY INT,
    STATUS CHAR(1),
    CONSTRAINT REZERWACJE_PK PRIMARY KEY
    (
        NR_REZERWACJI
    )
    ENABLE
);
```

1.4 Constrainty

```
ALTER TABLE REZERWACJE
ADD CONSTRAINT REZERWACJE_FK1 FOREIGN KEY
(
    ID_OSOBY
)
REFERENCES OSOBY
(
    ID_OSOBY
)
ENABLE;

ALTER TABLE REZERWACJE
ADD CONSTRAINT REZERWACJE_FK2 FOREIGN KEY
(
    ID_WYCIECZKI
)
REFERENCES WYCIECZKI
(
    ID_WYCIECZKI
)
ENABLE;
```

2. Uzupełnienie tabel danymi

2.1 Tabela Osoby

```
INSERT INTO OSOBY(IMIE, NAZWISKO, PESEL, KONTAKT)
VALUES('imie', 'nazwisko', '99112211777', '123456789');
```

	ID_OSOBY	IMIE	NAZWISKO	PESEL	KONTAKT
1	1	Krzysztof	Nalepa	99072056971	502748827
2	2	Jerzy	Kiler	77757775777	983940268
3	3	Jakub	Solecki	99041286383	743743271
4	4	Janina	Sanocka	88122798464	728364812
5	5	Marek	Krakus	74111778467	628365834
6	6	Bolesław	Krzywousty	54101475567	328717382
7	7	Krzysztof	Jarzyna	69082445559	927561983
8	8	Stanisław	Silnoręki	43072844453	129849873
9	9	Bronisława	Maślanka	89030824554	217834528
10	10	Robert	Kubica	79050314537	781826734

2.2 Tabela Wycieczki

```
INSERT INTO WYCIECZKI(NAZWA, KRAJ, DATA, OPIS, LICZBA_MIEJSC)
VALUES('nazwa', 'kraj', 'yyyy-mm-dd', 20);
```

	ID_WYCIECZKI	NAZWA	KRAJ	DATA	OPIS	LICZBA_MIEJSC
1	1	Piękne Węgrzce	Polska	2015-05-15 00:00:00	3-dniowa wycieczka po pięknej wsi	20
2	2	Wycieczka do Kopenhagi	Dania	2021-02-18 00:00:00	Poznaj jedno ze skandynawskich państw	34
3	3	Moskiewski szal	Rosja	2020-06-24 00:00:00	Odwiedź stolicę Rosji!	18
4	4	Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Jedno z najpiękniejszych miast Europy	12

2.3 Tabela Rezerwacje

```
INSERT INTO REZERWACJE (ID_WYCIECZKI, ID_OSOBY, STATUS)
VALUES(1,1, 'A');
```

	NR_REZERWACJI	ID_WYCIECZKI	ID_OSOBY	STATUS
1	1	1	3	Z
2	2	1	1	Z
3	3	2	2	N
4	4	2	4	N
5	5	2	6	N
6	6	3	5	P
7	7	3	10	P
8	8	3	9	N
9	9	3	8	N
10	10	3	7	A
11	11	3	1	N
12	12	4	2	Z
13	13	4	5	Z
14	14	4	6	Z
15	15	4	7	A
16	16	4	9	Z

3. Widoki

3.1 Widok rezerwacje_wszystkie

```

CREATE VIEW REZERWACJE_WSZYSTKIE
AS
    SELECT
        w.NAZWA,
        w.KRAJ,
        w.DATA,
        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;

```

	NAZWA	KRAJ	DATA	IMIE	NAZWISKO	STATUS
1	Piękne Węgrzce	Polska	2015-05-15 00:00:00	Krzysztof	Nalepa	Z
2	Moskiewski szal	Rosja	2020-06-24 00:00:00	Krzysztof	Nalepa	N
3	Wycieczka do Kopenhagi	Dania	2021-02-18 00:00:00	Jerzy	Kiler	N
4	Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Jerzy	Kiler	Z
5	Piękne Węgrzce	Polska	2015-05-15 00:00:00	Jakub	Solecki	Z
6	Wycieczka do Kopenhagi	Dania	2021-02-18 00:00:00	Janina	Sanocka	N
7	Moskiewski szal	Rosja	2020-06-24 00:00:00	Marek	Krakus	P
8	Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Marek	Krakus	Z
9	Wycieczka do Kopenhagi	Dania	2021-02-18 00:00:00	Bolesław	Krzywousty	N
10	Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Bolesław	Krzywousty	Z
11	Moskiewski szal	Rosja	2020-06-24 00:00:00	Krzysztof	Jarzyna	A
12	Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Krzysztof	Jarzyna	A
13	Moskiewski szal	Rosja	2020-06-24 00:00:00	Stanisław	Silnoreki	N
14	Moskiewski szal	Rosja	2020-06-24 00:00:00	Bronisława	Maślanka	N
15	Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Bronisława	Maślanka	Z
16	Moskiewski szal	Rosja	2020-06-24 00:00:00	Robert	Kubica	P

3.2 Widok rezerwacje_potwierdzone

```

CREATE VIEW REZERWACJE_POTWIERDZONE
AS
SELECT
    w.NAZWA,
    w.KRAJ,
    w.DATA,
    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
WHERE r.STATUS LIKE 'P' OR r.STATUS LIKE 'Z';

```

	NAZWA	KRAJ	DATA	IMIE	NAZWISKO	STATUS
1	Piękne Węgrzce	Polska	2015-05-15 00:00:00	Krzysztof	Nalepa	Z
2	Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Jerzy	Kiler	Z
3	Piękne Węgrzce	Polska	2015-05-15 00:00:00	Jakub	Solecki	Z
4	Moskiewski szal	Rosja	2020-06-24 00:00:00	Marek	Krakus	P
5	Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Marek	Krakus	Z
6	Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Bolesław	Krzywousty	Z
7	Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Bronisława	Maślanka	Z
8	Moskiewski szal	Rosja	2020-06-24 00:00:00	Robert	Kubica	P

3.3 Widok rezerwacje_w_przyszłości

```
CREATE VIEW REZERWACJE_W_PRZYSZLOSCI
AS
    SELECT
        w.NAZWA,
        w.KRAJ,
        w.DATA,
        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
    WHERE w.DATA > CURRENT_DATE;
```

	NAZWA	KRAJ	DATA	IMIE	NAZWISKO	STATUS
1	Moskiewski szał	Rosja	2020-06-24 00:00:00	Krzysztof	Nalepa	N
2	Wycieczka do Kopenhagi	Dania	2021-02-18 00:00:00	Jerzy	Kiler	N
3	Wycieczka do Kopenhagi	Dania	2021-02-18 00:00:00	Janina	Sanocka	N
4	Moskiewski szał	Rosja	2020-06-24 00:00:00	Marek	Krakus	P
5	Wycieczka do Kopenhagi	Dania	2021-02-18 00:00:00	Bolesław	Krzywousty	N
6	Moskiewski szał	Rosja	2020-06-24 00:00:00	Krzysztof	Jarzyna	A
7	Moskiewski szał	Rosja	2020-06-24 00:00:00	Stanisław	Silnoreki	N
8	Moskiewski szał	Rosja	2020-06-24 00:00:00	Bronisława	Maślanka	N
9	Moskiewski szał	Rosja	2020-06-24 00:00:00	Robert	Kubica	P

3.4 Widok wycieczki_miejsca

```
CREATE VIEW WYCIECZKI_MIEJSCA
AS
    SELECT
        w.KRAJ,
        w.DATA,
        w.NAZWA,
        w.LICZBA_MIEJSC,
        w.LICZBA_MIEJSC - (SELECT COUNT(*)
                           FROM REZERWACJE r
                           WHERE w.ID_WYCIECZKI = r.ID_WYCIECZKI AND
                                r.STATUS <> 'A') WOLNE_MIEJSCA
    FROM WYCIECZKI w;
```

	KRAJ	DATA	NAZWA	LICZBA_MIEJSC	WOLNE_MIEJSCA
1	Polska	2015-05-15 00:00:00	Piękne Węgrzce	20	18
2	Dania	2021-02-18 00:00:00	Wycieczka do Kopenhagi	34	31
3	Rosja	2020-06-24 00:00:00	Moskiewski szal	18	12
4	Węgry	2019-09-28 00:00:00	Budapeszt jesienią	12	7

3.5 Widok wycieczki_dostepne

```
CREATE VIEW WYCIECZKI_DOSTEPNE
AS
SELECT *
FROM WYCIECZKI_MIEJSCA wm
WHERE wm.DATA > CURRENT_DATE AND wm.WOLNE_MIEJSCA > 0;
```

	KRAJ	DATA	NAZWA	LICZBA_MIEJSC	WOLNE_MIEJSCA
1	Dania	2021-02-18 00:00:00	Wycieczka do Kopenhagi	34	31
2	Rosja	2020-06-24 00:00:00	Moskiewski szal	18	12

4. Obiekty

4.1 Obiekt uczestnik_wycieczki

```
CREATE OR REPLACE TYPE UCZESTNIK_WYCIECZKI AS OBJECT
(
    NAZWA VARCHAR2(100),
    KRAJ VARCHAR2(50),
    "DATA" DATE,
    IMIE VARCHAR2(50),
    NAZWISKO VARCHAR2(50),
    STATUS CHAR(1)
);

CREATE OR REPLACE TYPE UCZESTNICZY_WYCIECZKI_TABELA IS TABLE OF
UCZESTNIK_WYCIECZKI;
```

4.2 Obiekt wycieczka

```
CREATE OR REPLACE TYPE WYCIECZKA AS OBJECT
(
    NAZWA VARCHAR2(100),
    KRAJ VARCHAR2(50),
    "DATA" DATE,
    OPIS VARCHAR2(200),
    LICZBA_MIEJSC INT
```

```
);
```

```
CREATE OR REPLACE TYPE WYCIECZKI_TABELA IS TABLE OF WYCIECZKA;
```

5. Funkcje pobierające dane

5.1 Funkcja uczestnicy_wycieczki

```
CREATE OR REPLACE FUNCTION UCZESTNICZY_WYCIECZKI(id INT)
RETURN UCZESTNICZY_WYCIECZKI_TABELA AS uczestnicy
    UCZESTNICZY_WYCIECZKI_TABELA;
istnieje INT;
BEGIN
    SELECT COUNT(*) INTO istnieje
    FROM WYCIECZKI
    WHERE WYCIECZKI.ID_WYCIECZKI = id;

    IF istnieje = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Wycieczka o podanym ID nie
        istnieje');
    END IF;

    SELECT UCZESTNIK_WYCIECZKI(w.NAZWA, w.KRAJ, w.DATA, o.IMIE,
        o.NAZWISKO, r.STATUS)
        BULK COLLECT INTO uczestnicy
    FROM WYCIECZKI w
        JOIN REZERWACJE r
        ON w.ID_WYCIECZKI = r.ID_WYCIECZKI
        JOIN OSOBY o
        ON r.ID_OSOBY = o.ID_OSOBY
    WHERE w.ID_WYCIECZKI = id AND r.STATUS <> 'A';

    RETURN uczestnicy;
END;
```

```
SELECT * FROM UCZESTNICZY_WYCIECZKI(1);
```

	NAZWA	KRAJ	DATA	IMIE	NAZWISKO	STATUS
1	Piękne Węgrzce	Polska	2015-05-15 00:00:00	Jakub	Solecki	Z
2	Piękne Węgrzce	Polska	2015-05-15 00:00:00	Krzysztof	Nalepa	Z

5.2 Funkcja rezerwacje_osoby

```
CREATE OR REPLACE FUNCTION REZERWACJE_OSOBY(id INT)
```



```

RETURN UCZESTNICZY_WYCIECZKI_TABELA AS rezerwacje
    UCZESTNICZY_WYCIECZKI_TABELA;
istnieje INT;
BEGIN
    SELECT COUNT(*) INTO istnieje
    FROM OSOBY
    WHERE OSOBY.ID_OSOBY = id;

    IF istnieje = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Osoba o podanym ID nie
        istnieje');
    END IF;

    SELECT UCZESTNIK_WYCIECZKI(w.NAZWA, w.KRAJ, w.DATA, o.IMIE,
        o.NAZWISKO, r.STATUS)
        BULK COLLECT INTO rezerwacje
    FROM WYCIECZKI w
        JOIN REZERWACJE r
        ON w.ID_WYCIECZKI = r.ID_WYCIECZKI
        JOIN OSOBY o
        ON r.ID_OSOBY = o.ID_OSOBY
    WHERE o.ID_OSOBY = id;
    RETURN REZERWACJE;
END;

```

```
SELECT * FROM REZERWACJE_OSOBY(1);
```

	NAZWA	KRAJ	DATA	IMIE	NAZWISKO	STATUS
1	Piękne Węgrzce	Polska	2015-05-15 00:00:00	Krzysztof	Nalepa	Z
2	Moskiewski szal	Rosja	2020-06-24 00:00:00	Krzysztof	Nalepa	N

5.3 Funkcja dostępne_wycieczki

```

CREATE OR REPLACE FUNCTION DOSTEPNE_WYCIECZKI(kraj WYCIECZKI.KRAJ%TYPE)
RETURN WYCIECZKI_TABELA AS wycieczki WYCIECZKI_TABELA;
BEGIN
    SELECT WYCIECZKA(w.NAZWA, w.KRAJ, w.DATA, w.OPIS,
        w.LICZBA_MIEJSC)
        BULK COLLECT INTO wycieczki
    FROM WYCIECZKI w
    WHERE DOSTEPNE_WYCIECZKI.kraj LIKE w.KRAJ AND w.DATA >
        CURRENT_DATE AND
        w.LICZBA_MIEJSC > (SELECT COUNT(*)
            FROM REZERWACJE r
            WHERE r.STATUS <> 'A' AND

```

```

        r.ID_WYCIECZKI = w.ID_WYCIECZKI);
    RETURN wycieczki;
END;
```

```
SELECT * FROM DOSTEPNE_WYCIECZKI('Rosja');
```

	NAZWA	KRAJ	DATA	OPIS	LICZBA_MIEJSC
1	Moskiewski szlak	Rosja	2020-06-24 00:00:00	Odwiedź stolicę Rosji!	18

6. Procedury modyfikujące dane

6.1 Procedura dodaj_rezerwacje

```

CREATE OR REPLACE PROCEDURE DODAJ_REZERWACJE(IDwycieczki INT, IDosoby
INT)
AS
    person_exists INT;
    trip_exists INT;
    reservation_exists INT;
    free_places INT;
BEGIN
    SELECT COUNT(*) INTO trip_exists
    FROM WYCIECZKI W
    WHERE W.ID_WYCIECZKI = IDwycieczki;

    IF trip_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono wycieczki o
        podanym ID');
    END IF;

    SELECT COUNT(*) INTO person_exists
    FROM OSOBY o
    WHERE o.ID_OSOBY = IDosoby;

    IF person_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono osoby o podanym
        ID');
    END IF;

    SELECT COUNT(*) INTO reservation_exists
    FROM REZERWACJE r
    WHERE r.ID_WYCIECZKI = IDwycieczki
        AND r.ID_OSOBY = IDosoby;

    IF reservation_exists = 0 THEN
```

```

        RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja podanej osoby na tę
        wycieczkę już istnieje');
    END IF;

    SELECT w.LICZBA_MIEJSC - (SELECT COUNT(*)
                                FROM REZERWACJE r
                                INNER JOIN WYCIECZKI w2 ON r.ID_WYCIECZKI
                                    = w2.ID_WYCIECZKI
                                WHERE w.ID_WYCIECZKI = r.ID_WYCIECZKI AND
                                    r.STATUS <> 'A') INTO free_places

    FROM WYCIECZKI w
    WHERE DODAJ_REZERWACJE.IDwycieczki = w.ID_WYCIECZKI;

    IF free_places = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Na wybraną wycieczkę nie ma już
        wolnych miejsc');
    END IF;

    INSERT INTO REZERWACJE(ID_WYCIECZKI, ID_OSOBY, STATUS)
    VALUES (IDwycieczki, IDosoby, 'N');
    COMMIT;
END;

```

```

BEGIN
    DODAJ_REZERWACJE(2,10);
END;

```

	NR_REZERWACJI	ID_WYCIECZKI	ID_OSOBY	STATUS
1	22	2	10	N
2	1	1	3	Z
3	2	1	1	Z
4	3	2	2	N
5	4	2	4	N
6	5	2	6	N
7	6	3	5	P
8	7	3	10	P
9	8	3	9	N
10	9	3	8	N
11	10	3	7	A
12	11	3	1	N
13	12	4	2	Z
14	13	4	5	Z
15	14	4	6	Z
16	15	4	7	A
17	16	4	9	Z

6.2 Procedura zmien_status_rezerwacji

```
CREATE OR REPLACE PROCEDURE ZMIEN_STATUS_REZERWACJI(NRrezerwacji INT,
status REZERWACJE.STATUS%TYPE)
AS
    reservation_exists INT;
    old_status REZERWACJE.STATUS%TYPE;
BEGIN
    SELECT COUNT(*) INTO reservation_exists
    FROM REZERWACJE r
    WHERE r.NR_REZERWACJI = NRrezerwacji;

    IF reservation_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja o podanym numerze
        nie istnieje');
    END IF;

    SELECT STATUS INTO old_status
    FROM REZERWACJE r
    WHERE r.NR_REZERWACJI = NRrezerwacji;

    CASE
        WHEN old_status = 'A' THEN
            RAISE_APPLICATION_ERROR(-20000, 'Nie można zmienić statusu
            anulowanej rezerwacji');
        WHEN old_status = ZMIEN_STATUS_REZERWACJI.status THEN
            RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja już posiada
            podany status');
        WHEN old_status = 'P' THEN
            IF ZMIEN_STATUS_REZERWACJI.status <> 'Z' and
            ZMIEN_STATUS_REZERWACJI.status <> 'A' THEN
                RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji
                "potwierdzona" może być zmieniony jedynie na "opłacona"
                lub "anulowana"');
            END IF;
        WHEN old_status = 'Z' THEN
            IF ZMIEN_STATUS_REZERWACJI.status <> 'A' THEN
                RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji
                "opłacona" może być zmieniony jedynie na "anulowana"');
            END IF;
        WHEN old_status = 'N' THEN
            IF ZMIEN_STATUS_REZERWACJI.status <> 'Z' and
            ZMIEN_STATUS_REZERWACJI.status <> 'P' and
            ZMIEN_STATUS_REZERWACJI.status <> 'A' THEN
                RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji
```

```

        "nowa" może być zmieniony jedynie na "potwierdzona",
        "opłacona" lub "anulowana");
    END IF;
ELSE
    RAISE_APPLICATION_ERROR(-20000, 'CASE not found');
END CASE;

UPDATE REZERWACJE r
SET r.STATUS = ZMIEN_STATUS_REZERWACJI.status
WHERE r.NR_REZERWACJI = ZMIEN_STATUS_REZERWACJI.NRrezerwacji;
COMMIT;
END;

```

```

BEGIN
    ZMIEN_STATUS_REZERWACJI(22, 'P');
END;

```

	NR_REZERWACJI	ID_WYCIECZKI	ID_OSOBY	STATUS
1	22	2	10	P
2	1	1	3	Z
3	2	1	1	Z
4	3	2	2	N
5	4	2	4	N
6	5	2	6	N
7	6	3	5	P
8	7	3	10	P
9	8	3	9	N
10	9	3	8	N
11	10	3	7	A
12	11	3	1	N
13	12	4	2	Z
14	13	4	5	Z
15	14	4	6	Z
16	15	4	7	A
17	16	4	9	Z

6.3 Procedura zmien_liczbe_miejsc

```

CREATE OR REPLACE PROCEDURE ZMIEN_LICZBE_MIEJSC(IDwycieczki INT, miejsca
INT)
AS
    trip_exists INT;
    places_booked INT;
BEGIN

```

```

SELECT COUNT(*) INTO trip_exists
FROM WYCIECZKI w
WHERE w.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC.IDwycieczki;

IF trip_exists = 0 THEN
    RAISE_APPLICATION_ERROR(-20000, 'Wycieczka o podanym ID nie istnieje');
END IF;

SELECT COUNT(*) INTO places_booked
FROM REZERWACJE r
WHERE r.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC.IDwycieczki AND r.STATUS
<> 'A';

IF ZMIEN_LICZBE_MIEJSC.miejsca < places_booked THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nie można zmniejszyć liczby miejsc poniżej liczby miejsc już zarezerwowanych');
END IF;

IF ZMIEN_LICZBE_MIEJSC.miejsca <= 0 THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nie można zmniejszyć liczby miejsc poniżej 1');
END IF;

UPDATE WYCIECZKI w
SET w.LICZBA_MIEJSC = ZMIEN_LICZBE_MIEJSC.miejsca
WHERE w.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC.IDwycieczki;
COMMIT;
END;

```

```

BEGIN
    ZMIEN_LICZBE_MIEJSC(1,50);
END;

```

	ID_WYCIECZKI	NAZWA	KRAJ	DATA	OPIS	LICZBA_MIEJSC
1		1 Piękne Węgrzce	Polska	2015-05-15 00:00:00	3-dniowa wycieczka po pięknej wsi	50
2		2 Wycieczka do Kopenhagi	Dania	2021-02-18 00:00:00	Poznaj jedno ze skandynawskich państw	34
3		3 Moskiewski szal	Rosja	2020-06-24 00:00:00	Odwiedź stolicę Rosji!	18
4		4 Budapeszt jesienią	Węgry	2019-09-28 00:00:00	Jedno z najpiękniejszych miast Europy	12

7. Tabela dziennikująca zmiany statusu rezerwacji

7.1 Tabela

```
CREATE TABLE REZERWACJE_LOG
```

```
(
    ID_LOG INT GENERATED ALWAYS AS IDENTITY NOT NULL,
    NR_REZERWACJI INT,
    DATA DATE,
    STATUS CHAR(1),
    CONSTRAINT REZERWACJE_LOG_PK PRIMARY KEY
    (
        ID_LOG
    )
    ENABLE
);
```

7.2 Constraint

```
ALTER TABLE REZERWACJE_LOG
ADD CONSTRAINT REZERWACJE_FK FOREIGN KEY
(
    NR_REZERWACJI
)
REFERENCES REZERWACJE
(
    NR_REZERWACJI
)
ENABLE;
```

7.3 Zmodyfikowana procedura dodaj_rezerwacje

```
CREATE OR REPLACE PROCEDURE DODAJ_REZERWACJE(IDwycieczki INT, IDosoby
INT)
AS
    person_exists INT;
    trip_exists INT;
    reservation_exists INT;
    free_places INT;
    new_reservation_id INT;
BEGIN
    SELECT COUNT(*) INTO trip_exists
    FROM WYCIECZKI W
    WHERE W.ID_WYCIECZKI = IDwycieczki;

    IF trip_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono wycieczki o
        podanym ID');
    END IF;
```

```

SELECT COUNT(*) INTO person_exists
FROM OSOBY o
WHERE o.ID_OSOBY = IDosoby;

IF person_exists = 0 THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono osoby o podanym
    ID');
END IF;

SELECT COUNT(*) INTO reservation_exists
FROM REZERWACJE r
WHERE r.ID_WYCIECZKI = IDwycieczki
    AND r.ID_OSOBY = IDosoby;

IF reservation_exists > 0 THEN
    RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja podanej osoby na tę
    wycieczkę już istnieje');
END IF;

SELECT w.LICZBA_MIEJSC - (SELECT COUNT(*)
    FROM REZERWACJE r
    INNER JOIN WYCIECZKI w2 ON r.ID_WYCIECZKI
    = w2.ID_WYCIECZKI
    WHERE w.ID_WYCIECZKI = r.ID_WYCIECZKI AND
    r.STATUS <> 'A') INTO free_places

FROM WYCIECZKI w
WHERE DODAJ_REZERWACJE.IDwycieczki = w.ID_WYCIECZKI;

IF free_places = 0 THEN
    RAISE_APPLICATION_ERROR(-20000, 'Na wybraną wycieczkę nie ma już
    wolnych miejsc');
END IF;

INSERT INTO REZERWACJE(ID_WYCIECZKI, ID_OSOBY, STATUS)
VALUES (IDwycieczki, IDosoby, 'N')
RETURNING NR_REZERWACJI INTO new_reservation_id;

INSERT INTO REZERWACJE_LOG(NR_REZERWACJI, DATA, STATUS)
VALUES (new_reservation_id, CURRENT_DATE, 'N');
COMMIT;
END;

```

```

BEGIN
    DODAJ_REZERWACJE(2,8);
END;

```


Tabela rezerwacje:

	NR_REZERWACJI	ID_WYCIECZKI	ID_OSOBY	STATUS
1	41	2	8	N
2	22	2	10	P
3	16	4	9	Z
4	15	4	7	A
5	14	4	6	Z
6	13	4	5	Z
7	12	4	2	Z
8	11	3	1	N
9	10	3	7	A
10	9	3	8	N
11	8	3	9	N
12	7	3	10	P
13	6	3	5	P
14	5	2	6	N
15	4	2	4	N
16	3	2	2	N
17	2	1	1	Z
18	1	1	3	Z

Tabela rezerwacje_log:

	ID_LOG	NR_REZERWACJI	DATA	STATUS
1	1	41	2020-03-08 21:48:06	N

7.4 Zmieniona procedura zmien_status_rezerwacji

```

CREATE OR REPLACE PROCEDURE ZMIEN_STATUS_REZERWACJI(NRrezerwacji INT,
    status REZERWACJE.STATUS%TYPE)
AS
    reservation_exists INT;
    old_status REZERWACJE.STATUS%TYPE;
BEGIN
    SELECT COUNT(*) INTO reservation_exists
    FROM REZERWACJE r
    WHERE r.NR_REZERWACJI = NRrezerwacji;

    IF reservation_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja o podanym numerze
        nie istnieje');
    END IF;

    SELECT STATUS INTO old_status
    FROM REZERWACJE r
    WHERE r.NR_REZERWACJI = NRrezerwacji;

```

```

CASE
  WHEN old_status = 'A' THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nie można zmienić statusu
    anulowanej rezerwacji');
  WHEN old_status = ZMIEN_STATUS_REZERWACJI.status THEN
    RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja już posiada
    podany status');
  WHEN old_status = 'P' THEN
    IF ZMIEN_STATUS_REZERWACJI.status <> 'Z' and
    ZMIEN_STATUS_REZERWACJI.status <> 'A' THEN
      RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji
      "potwierdzona" może być zmieniony jedynie na "opłacona"
      lub "anulowana"');
    END IF;
  WHEN old_status = 'Z' THEN
    IF ZMIEN_STATUS_REZERWACJI.status <> 'A' THEN
      RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji
      "opłacona" może być zmieniony jedynie na "anulowana"');
    END IF;
  WHEN old_status = 'N' THEN
    IF ZMIEN_STATUS_REZERWACJI.status <> 'Z' and
    ZMIEN_STATUS_REZERWACJI.status <> 'P' and
    ZMIEN_STATUS_REZERWACJI.status <> 'A' THEN
      RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji
      "nowa" może być zmieniony jedynie na "potwierdzona",
      "opłacona" lub "anulowana"');
    END IF;
  ELSE
    RAISE_APPLICATION_ERROR(-20000, 'CASE not found');
END CASE;

UPDATE REZERWACJE r
SET r.STATUS = ZMIEN_STATUS_REZERWACJI.status
WHERE r.NR_REZERWACJI = ZMIEN_STATUS_REZERWACJI.NRrezerwacji;

INSERT INTO REZERWACJE_LOG(NR_REZERWACJI, DATA, STATUS)
VALUES(ZMIEN_STATUS_REZERWACJI.NRrezerwacji, CURRENT_DATE,
ZMIEN_STATUS_REZERWACJI.status);
COMMIT;
END;

```

```

BEGIN
  ZMIEN_STATUS_REZERWACJI(41, 'P');
END;

```

Tabela rezerwacje:

	NR_REZERWACJI	ID_WYCIECZKI	ID_OSOBY	STATUS
1	22	2	10	P
2	1	1	3	Z
3	2	1	1	Z
4	3	2	2	N
5	4	2	4	N
6	5	2	6	N
7	6	3	5	P
8	7	3	10	P
9	8	3	9	N
10	9	3	8	N
11	10	3	7	A
12	11	3	1	N
13	12	4	2	Z
14	13	4	5	Z
15	14	4	6	Z
16	15	4	7	A
17	16	4	9	Z
18	41	2	8	P

Tabela rezerwacje_log:

	ID_LOG	NR_REZERWACJI	DATA	STATUS
1	1	41	2020-03-08 21:48:06	N
2	2	41	2020-03-08 21:54:26	P

8. Dodanie redundantnego pola wolne_miejsca do tabeli wycieczki

8.1 Tabela wycieczki

```
ALTER TABLE WYCIECZKI
ADD WOLNE_MIEJSCA INT;
```

8.2 Widok wycieczki_miejsca2

```
CREATE VIEW WYCIECZKI_MIEJSCA2
AS
SELECT
    w.KRAJ,
    w.DATA,
    w.NAZWA,
    w.LICZBA_MIEJSC,
    w.WOLNE_MIEJSCA
```

```
FROM WYCIECZKI w;
```

8.3 Widok wycieczki_dostepne2

```
CREATE VIEW WYCIECZKI_DOSTEPNE2
AS
    SELECT *
    FROM WYCIECZKI_MIEJSCA wm
    WHERE wm.WOLNE_MIEJSCA > 0;
```

8.4 Procedura przelicz

```
CREATE OR REPLACE PROCEDURE PRZELICZ
AS
    BEGIN
        UPDATE WYCIECZKI w
            SET w.WOLNE_MIEJSCA = w.LICZBA_MIEJSC - (SELECT COUNT(*)
                                                    FROM REZERWACJE r
                                                    WHERE r.ID_WYCIECZKI
                                                          = w.ID_WYCIECZKI AND
                                                          r.STATUS <> 'A');

        COMMIT;
    END;
```

8.5 Funkcja pobierająca dane dostepne_wycieczki2

```
CREATE OR REPLACE FUNCTION DOSTEPNE_WYCIECZKI2(kraj WYCIECZKI.KRAJ%TYPE)
RETURN WYCIECZKI_TABELA AS wycieczki WYCIECZKI_TABELA;
BEGIN
    SELECT WYCIECZKA(w.NAZWA, w.KRAJ, w.DATA, w.OPIS,
                    w.LICZBA_MIEJSC)
    BULK COLLECT INTO wycieczki
    FROM WYCIECZKI w
    WHERE DOSTEPNE_WYCIECZKI2.kraj LIKE w.KRAJ AND w.DATA >
        CURRENT_DATE AND w.WOLNE_MIEJSCA > 0;
    RETURN wycieczki;
END;
```

8.6 Procedura wprowadzająca dane dodaj_rezerwacje2

```
CREATE OR REPLACE PROCEDURE DODAJ_REZERWACJE2(IDwycieczki INT, IDosoby
INT)
AS
    person_exists INT;
```

```

trip_exists INT;
reservation_exists INT;
free_places INT;
new_reservation_id INT;
BEGIN
    SELECT COUNT(*) INTO trip_exists
    FROM WYCIECZKI W
    WHERE W.ID_WYCIECZKI = IDwycieczki;

    IF trip_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono wycieczki o
        podanym ID');
    END IF;

    SELECT COUNT(*) INTO person_exists
    FROM OSOBY o
    WHERE o.ID_OSOBY = IDosoby;

    IF person_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono osoby o podanym
        ID');
    END IF;

    SELECT COUNT(*) INTO reservation_exists
    FROM REZERWACJE r
    WHERE r.ID_WYCIECZKI = IDwycieczki
        AND r.ID_OSOBY = IDosoby;

    IF reservation_exists > 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja podanej osoby na tę
        wycieczkę już istnieje');
    END IF;

    SELECT w.WOLNE_MIEJSCA INTO free_places
    FROM WYCIECZKI w
    WHERE DODAJ_REZERWACJE2.IDwycieczki = w.ID_WYCIECZKI;

    IF free_places = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Na wybraną wycieczkę nie ma już
        wolnych miejsc');
    END IF;

    INSERT INTO REZERWACJE(ID_WYCIECZKI, ID_OSOBY, STATUS)
    VALUES (IDwycieczki, IDosoby, 'N')
    RETURNING NR_REZERWACJI INTO new_reservation_id;

```

```

UPDATE WYCIECZKI
SET WOLNE_MIEJSCA = WOLNE_MIEJSCA - 1
WHERE ID_WYCIECZKI = DODAJ_REZERWACJE2.IDwycieczki;

INSERT INTO REZERWACJE_LOG(NR_REZERWACJI, DATA, STATUS)
VALUES (new_reservation_id, CURRENT_DATE, 'N');
COMMIT;
END;

```

8.7 Procedura wprowadzająca dane zmien_status_rezerwacji2

```

CREATE OR REPLACE PROCEDURE ZMIEN_STATUS_REZERWACJI2(NRrezerwacji INT,
status REZERWACJE.STATUS%TYPE)
AS
    reservation_exists INT;
    old_status REZERWACJE.STATUS%TYPE;
BEGIN
    SELECT COUNT(*) INTO reservation_exists
    FROM REZERWACJE r
    WHERE r.NR_REZERWACJI = NRrezerwacji;

    IF reservation_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja o podanym numerze
        nie istnieje');
    END IF;

    SELECT STATUS INTO old_status
    FROM REZERWACJE r
    WHERE r.NR_REZERWACJI = NRrezerwacji;

    CASE
        WHEN old_status = 'A' THEN
            RAISE_APPLICATION_ERROR(-20000, 'Nie można zmienić statusu
            anulowanej rezerwacji');
        WHEN old_status = ZMIEN_STATUS_REZERWACJI2.status THEN
            RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja już posiada
            podany status');
        WHEN old_status = 'P' THEN
            IF ZMIEN_STATUS_REZERWACJI2.status <> 'Z' and
            ZMIEN_STATUS_REZERWACJI2.status <> 'A' THEN
                RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji
                "potwierdzona" może być zmieniony jedynie na "opłacona"
                lub "anulowana"');
            END IF;
    END CASE;
END;

```

```

        WHEN old_status = 'Z' THEN
            IF ZMIEN_STATUS_REZERWACJI2.status <> 'A' THEN
                RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji
                "opłacona" może być zmieniony jedynie na "anulowana"');
            END IF;
        WHEN old_status = 'N' THEN
            IF ZMIEN_STATUS_REZERWACJI2.status <> 'Z' and
                ZMIEN_STATUS_REZERWACJI2.status <> 'P' and
                ZMIEN_STATUS_REZERWACJI2.status <> 'A' THEN
                RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji
                "nowa" może być zmieniony jedynie na "potwierdzona",
                "opłacona" lub "anulowana"');
            END IF;
        ELSE
            RAISE_APPLICATION_ERROR(-20000, 'CASE not found');
        END CASE;

    UPDATE REZERWACJE r
    SET r.STATUS = ZMIEN_STATUS_REZERWACJI2.status
    WHERE r.NR_REZERWACJI = ZMIEN_STATUS_REZERWACJI2.NRrezerwacji;

    IF ZMIEN_STATUS_REZERWACJI2.status = 'A' THEN
        UPDATE WYCIECZKI w
        SET w.WOLNE_MIEJSCA = w.WOLNE_MIEJSCA + 1
        WHERE ID_WYCIECZKI = (SELECT r.ID_WYCIECZKI
                                FROM REZERWACJE r
                                WHERE r.NR_REZERWACJI =
                                ZMIEN_STATUS_REZERWACJI2.NRrezerwacji);

    END IF;

    INSERT INTO REZERWACJE_LOG(NR_REZERWACJI, DATA, STATUS)
    VALUES(ZMIEN_STATUS_REZERWACJI2.NRrezerwacji, CURRENT_DATE,
    ZMIEN_STATUS_REZERWACJI2.status);
    COMMIT;
END;

```

8.8 Procedura wprowadzająca dane zmien_liczbe_miejsc2

```

CREATE OR REPLACE PROCEDURE ZMIEN_LICZBE_MIEJSC2(IDwycieczki INT,
                                                    miejsca INT)

AS
    trip_exists INT;
    places_booked INT;
BEGIN

```

```

SELECT COUNT(*) INTO trip_exists
FROM WYCIECZKI w
WHERE w.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC2.IDwycieczki;

IF trip_exists = 0 THEN
    RAISE_APPLICATION_ERROR(-20000, 'Wycieczka o podanym ID nie istnieje');
END IF;

SELECT COUNT(*) INTO places_booked
FROM REZERWACJE r
WHERE r.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC2.IDwycieczki AND r.STATUS
<> 'A';

IF ZMIEN_LICZBE_MIEJSC2.miejsca < places_booked THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nie można zmniejszyć liczby miejsc poniżej liczby miejsc już zarezerwowanych');
END IF;

IF ZMIEN_LICZBE_MIEJSC2.miejsca <= 0 THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nie można zmniejszyć liczby miejsc poniżej 1');
END IF;

UPDATE WYCIECZKI w
SET w.LICZBA_MIEJSC = ZMIEN_LICZBE_MIEJSC2.miejsca,
    w.WOLNE_MIEJSCA = ZMIEN_LICZBE_MIEJSC2.miejsca -
        ZMIEN_LICZBE_MIEJSC2.places_booked
WHERE w.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC2.IDwycieczki;
COMMIT;
END;

```

9. Triggery

9.1 Trigger dodawanie rezerwacji

```

CREATE OR REPLACE TRIGGER DODANIE_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);

```



```
UPDATE WYCIECZKI w
SET w.WOLNE_MIEJSCA = w.WOLNE_MIEJSCA -1
WHERE w.ID_WYCIECZKI = :NEW.ID_WYCIECZKI;
END;
```

9.2 Trigger zmiana_statusu

```
CREATE OR REPLACE TRIGGER ZMIANA_STAUTSU
AFTER UPDATE
ON REZERWACJE
FOR EACH ROW
DECLARE new_place INT;
BEGIN
    INSERT INTO REZERWACJE_LOG(NR_REZERWACJI, DATA, STATUS)
    VALUES(:NEW.NR_REZERWACJI, CURRENT_DATE, :NEW.STATUS);
    IF :NEW.STATUS = 'A' THEN
        new_place := 1;
    ELSE
        new_place := 0;
    END IF;
    UPDATE WYCIECZKI w
    SET w.WOLNE_MIEJSCA = w.WOLNE_MIEJSCA + new_place
    WHERE w.ID_WYCIECZKI = :NEW.ID_WYCIECZKI;
END;
```

9.3 Trigger blokada_usuniecia_rezerwacji

```
CREATE OR REPLACE TRIGGER BLOKADA_USUNIECIA_REZERWACJI
BEFORE DELETE
ON REZERWACJE
FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20000, 'Nie mozna usunac rezerwacji,
    mozna jedynie zmienic jej status na anulowana');
END;
```

9.4 Trigger zmiana_liczby_miejsc

```
CREATE OR REPLACE TRIGGER ZMIANA_LICZBY_MIEJSC
AFTER UPDATE OF LICZBA_MIEJSC
ON WYCIECZKI
FOR EACH ROW
BEGIN
    UPDATE WYCIECZKI
```

```
        SET WOLNE_MIEJSCA = WOLNE_MIEJSCA + (:NEW.LICZBA_MIEJSC -  
        LICZBA_MIEJSC);  
    END;
```

9.5 Zmodyfikowana procedura wstawiająca dane dodaj_rezerwacje

```
CREATE OR REPLACE PROCEDURE DODAJ_REZERWACJE3(IDwycieczki INT, IDosoby  
INT)  
AS  
    person_exists INT;  
    trip_exists INT;  
    reservation_exists INT;  
BEGIN  
    SELECT COUNT(*) INTO trip_exists  
    FROM WYCIECZKI W  
    WHERE W.ID_WYCIECZKI = IDwycieczki;  
  
    IF trip_exists = 0 THEN  
        RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono wycieczki o  
        podanym ID');  
    END IF;  
  
    SELECT COUNT(*) INTO person_exists  
    FROM OSOBY o  
    WHERE o.ID_OSOBY = IDosoby;  
  
    IF person_exists = 0 THEN  
        RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono osoby o podanym  
        ID');  
    END IF;  
  
    SELECT COUNT(*) INTO reservation_exists  
    FROM REZERWACJE r  
    WHERE r.ID_WYCIECZKI = IDwycieczki  
        AND r.ID_OSOBY = IDosoby;  
  
    IF reservation_exists > 0 THEN  
        RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja podanej osoby na tę  
        wycieczkę już istnieje');  
    END IF;  
  
    INSERT INTO REZERWACJE(ID_WYCIECZKI, ID_OSOBY, STATUS)  
    VALUES (DODAJ_REZERWACJE3.IDwycieczki, DODAJ_REZERWACJE3.IDosoby,  
    'N');
```

```
COMMIT;  
END;
```

9.6 Zmodyfikowana procedura wstawiająca dane zmien_status_rezerwacji

```
CREATE OR REPLACE PROCEDURE ZMIEN_STATUS_REZERWACJI3(NRrezerwacji INT,  
                                                    status REZERWACJE.STATUS%TYPE)  
AS  
    reservation_exists INT;  
    old_status REZERWACJE.STATUS%TYPE;  
BEGIN  
    SELECT COUNT(*) INTO reservation_exists  
    FROM REZERWACJE r  
    WHERE r.NR_REZERWACJI = NRrezerwacji;  
  
    IF reservation_exists = 0 THEN  
        RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja o podanym numerze  
        nie istnieje');  
    END IF;  
  
    SELECT STATUS INTO old_status  
    FROM REZERWACJE r  
    WHERE r.NR_REZERWACJI = NRrezerwacji;  
  
    CASE  
        WHEN old_status = 'A' THEN  
            RAISE_APPLICATION_ERROR(-20000, 'Nie można zmienić statusu  
            anulowanej rezerwacji');  
        WHEN old_status = ZMIEN_STATUS_REZERWACJI3.status THEN  
            RAISE_APPLICATION_ERROR(-20000, 'Rezerwacja już posiada  
            podany status');  
        WHEN old_status = 'P' THEN  
            IF ZMIEN_STATUS_REZERWACJI3.status <> 'Z' and  
               ZMIEN_STATUS_REZERWACJI3.status <> 'A' THEN  
                RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji  
                "potwierdzona" może być zmieniony jedynie na "opłacona"  
                lub "anulowana"');  
            END IF;  
        WHEN old_status = 'Z' THEN  
            IF ZMIEN_STATUS_REZERWACJI3.status <> 'A' THEN  
                RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji  
                "opłacona" może być zmieniony jedynie na "anulowana"');
```

```

        END IF;
    WHEN old_status = 'N' THEN
        IF ZMIEN_STATUS_REZERWACJI3.status <> 'Z' and
           ZMIEN_STATUS_REZERWACJI3.status <> 'P' and
           ZMIEN_STATUS_REZERWACJI3.status <> 'A' THEN
            RAISE_APPLICATION_ERROR(-20000, 'Status rezerwacji
            "nowa" może być zmieniony jedynie na "potwierdzona",
            "opłacona" lub "anulowana"');
        END IF;
    ELSE
        RAISE_APPLICATION_ERROR(-20000, 'CASE not found');
    END CASE;

    UPDATE REZERWACJE r
    SET r.STATUS = ZMIEN_STATUS_REZERWACJI3.status
    WHERE r.NR_REZERWACJI = ZMIEN_STATUS_REZERWACJI3.NRrezerwacji;
END;

```

9.7 Zmodyfikowana procedura wstawiająca dane zmien_liczbe_miejsc

```

CREATE OR REPLACE PROCEDURE ZMIEN_LICZBE_MIEJSC3(IDwycieczki INT,
miejsca INT)
AS
    trip_places INT;
    places_booked INT;
BEGIN
    SELECT w.LICZBA_MIEJSC INTO trip_places
    FROM WYCIECZKI w
    WHERE w.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC3.IDwycieczki;

    IF trip_places IS NULL THEN
        RAISE_APPLICATION_ERROR(-20000, 'Wycieczka o podanym ID nie
        istnieje');
    END IF;

    SELECT COUNT(*) INTO places_booked
    FROM REZERWACJE r
    WHERE r.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC3.IDwycieczki AND r.STATUS
    <> 'A';

    IF ZMIEN_LICZBE_MIEJSC3.miejsca < places_booked THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nie można zmniejszyć liczby
        miejsc poniżej liczby miejsc już zarezerwowanych');
    END IF;

```

```
IF ZMIEN_LICZBE_MIEJSC3.miejsca <= 0 THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nie można zmniejszyć liczby
    miejsc poniżej 1');
END IF;

UPDATE WYCIECZKI w
SET w.LICZBA_MIEJSC = ZMIEN_LICZBE_MIEJSC3.miejsca
WHERE w.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC3.IDwycieczki;
COMMIT;
END;
```