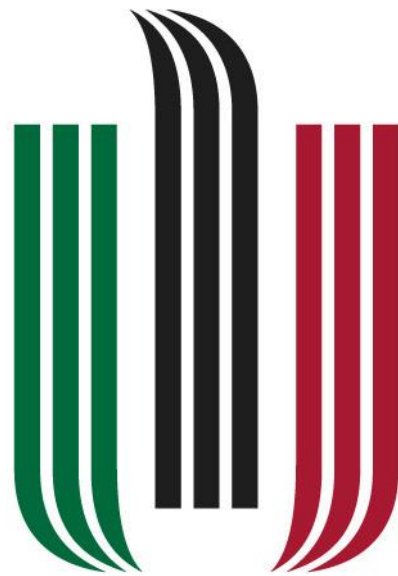


Laboratorium 1 : Oracle PL/SQL

Paweł Kocimski



AGH

1. Tabele

a) 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
)
```

b) Wycieczki

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

c) Rezerwacje

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

d) Rezerwacje_log

```
CREATE TABLE REZERWACJE_LOG
(
  ID INT GENERATED ALWAYS AS IDENTITY NOT NULL
, ID_REZERWACJI INT
, DATA DATE
, STATUS VARCHAR2(1)

, CONSTRAINT REZERWACJE_LOG PRIMARY KEY
(
  ID
)
ENABLE
);
```

e) 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;
ALTER TABLE REZERWACJE
ADD CONSTRAINT REZERWACJE_CHK1 CHECK
(status IN ('N', 'P', 'Z', 'A'))
ENABLE;

ALTER TABLE REZERWACJE_LOG
ADD CONSTRAINT REZERWACJE_LOG_FK1 FOREIGN KEY
(
  ID_REZERWACJI
)
REFERENCES REZERWACJE
(
  NR_REZERWACJI
)
ENABLE;
ALTER TABLE REZERWACJE_LOG
ADD CONSTRAINT REZERWACJE_LOG_CHK1 CHECK
(status IN ('N', 'P', 'Z', 'A'))
ENABLE;
```

2. Przykładowe dane

```
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: tel: 2312, dzwonić po 18.00');
INSERT INTO osoby (imie, nazwisko, pesel, kontakt)
VALUES('Paweł', 'Wójcik', '32154321', 'tel: 54352542');
INSERT INTO osoby (imie, nazwisko, pesel, kontakt)
VALUES('Michał', 'Wiśniewski', '12345438', 'tel: 24524522, dzwonić po 18.00');
VALUES('Adam', 'Noga', '32154321', 'tel: 5452442');
INSERT INTO osoby (imie, nazwisko, pesel, kontakt)
VALUES('Grzegorz', 'Szpak', '12376438', 'tel: 523424, dzwonić po 18.00');
VALUES('Kamil', 'Noga', '33143121', 'tel: 1312871');
INSERT INTO osoby (imie, nazwisko, pesel, kontakt)
VALUES('Karol', 'Gawron', '54389438', 'tel: 643231, dzwonić po 20.00');
VALUES('Anna', 'Lewska', '67981234', 'tel: 569823432');
INSERT INTO osoby (imie, nazwisko, pesel, kontakt)
VALUES('Diana', 'Mazak', '34124242', 'tel: 269834097, dzwonić po 19.00');
```

	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	Paweł	Wójcik	32154321	tel: 54352542
4	4	Michał	Wiśniewski	12345438	tel: 24524522, dzwonić po 18.00
5	5	Adam	Noga	32154321	tel: 5452442
6	6	Grzegorz	Szpak	12376438	tel: 523424, dzwonić po 18.00
7	7	Kamil	Noga	33143121	tel: 1312871
8	8	Karol	Gawron	54389438	tel: 643231, dzwonić po 20.00
9	9	Anna	Lewska	67981234	tel: 569823432
10	10	Diana	Mazak	34124242	tel: 269834097, dzwonić po 19.00

```
INSERT INTO wycieczki (nazwa, kraj, data, opis, liczba_miejsc)
VALUES ('Wycieczka do Paryża', 'Francja', TO_DATE('2019-12-09', 'YYYY-MM-DD'), 'Ciekawa wycieczka ', 31);
INSERT INTO wycieczki (nazwa, kraj, data, opis, liczba_miejsc)
VALUES ('Piękny Kraków', 'Polska', TO_DATE('2018-10-12', 'YYYY-MM-DD'), 'Najciekawsza wycieczka ', 54);
INSERT INTO wycieczki (nazwa, kraj, data, opis, liczba_miejsc)
VALUES ('Wieliczka', 'Polska', TO_DATE('2020-04-09', 'YYYY-MM-DD'), 'Zadziwiająca kopalnia ', 46);
INSERT INTO wycieczki (nazwa, kraj, data, opis, liczba_miejsc)
VALUES ('Warszawa', 'Polska', TO_DATE('2020-05-18', 'YYYY-MM-DD'), 'Stolica dla nas ', 42);
```

	ID_WYCIEZKI	NAZWA	KRAJ	DATA	OPIS	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC
1	4	Wycieczka do Paryża	Francja	2019-12-09 00:00:00	Ciekawa wycieczka	31	<null>
2	5	Piękny Kraków	Polska	2018-10-12 00:00:00	Najciekawsza wycieczka	54	<null>
3	6	Wieliczka	Polska	2020-04-09 00:00:00	Zadziwiająca kopalnia	35	<null>
4	7	Warszawa	Polska	2020-05-18 00:00:00	Stolica dla nas	42	<null>

```
INSERT INTO rezerwacje(id_wycieczki, id_osoby, status)
VALUES (4,1, 'N');
INSERT INTO rezerwacje(id_wycieczki, id_osoby, status)
VALUES (5,2, 'P');
```

```

INSERT INTO rezerwacje(id_wycieczki, id_osoby, status)
VALUES (6,3,'Z');
INSERT INTO rezerwacje(id_wycieczki, id_osoby, status)
VALUES (7,4,'A');
INSERT INTO rezerwacje(id_wycieczki, id_osoby, status)
VALUES (4,5,'N');
INSERT INTO rezerwacje(id_wycieczki, id_osoby, status)
VALUES (5,6,'P');
INSERT INTO rezerwacje(id_wycieczki, id_osoby, status)
VALUES (6,7,'Z');
INSERT INTO rezerwacje(id_wycieczki, id_osoby, status)
VALUES (7,8,'Z');
INSERT INTO rezerwacje(id_wycieczki, id_osoby, status)
VALUES (4,9,'A');
INSERT INTO rezerwacje(id_wycieczki, id_osoby, status)
VALUES (5,10,'P');

```

	NR_REZERWACJI	ID_WYCIECZKI	ID_OSOBY	STATUS
1	61	7	2	N
2	2	4	1	N
3	3	5	2	P
4	4	6	3	Z
5	5	7	4	A
6	6	4	5	N
7	7	5	6	Z
8	8	6	7	Z
9	9	7	8	Z
10	10	4	9	A
11	11	5	10	P
12	21	6	1	N
13	41	6	2	N

3. Widoki

a) RezerwacjeWszystkie

```

CREATE VIEW RezerwacjeWszystkie
AS
SELECT
w.ID_WYCIECZKI,
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;

```

	ID_WYCIECZKI	NAZWA	KRAJ	DATA	IMIE	NAZWISKO	STATUS
1	4	Wycieczka do Paryza	Francja	2019-12-09 00:00:00	Adam	Kowalski	N
2	6	Wieliczka	Polska	2020-04-09 00:00:00	Adam	Kowalski	N
3	5	Piękny Kraków	Polska	2018-10-12 00:00:00	Jan	Nowak	P
4	6	Wieliczka	Polska	2020-04-09 00:00:00	Jan	Nowak	N
5	7	Warszawa	Polska	2020-05-18 00:00:00	Jan	Nowak	N
6	6	Wieliczka	Polska	2020-04-09 00:00:00	Paweł	Wójcik	Z
7	7	Warszawa	Polska	2020-05-18 00:00:00	Michał	Wiśniewski	A
8	4	Wycieczka do Paryza	Francja	2019-12-09 00:00:00	Adam	Noga	N
9	5	Piękny Kraków	Polska	2018-10-12 00:00:00	Grzegorz	Szpak	Z
10	6	Wieliczka	Polska	2020-04-09 00:00:00	Kamil	Noga	Z
11	7	Warszawa	Polska	2020-05-18 00:00:00	Karol	Gawron	Z
12	4	Wycieczka do Paryza	Francja	2019-12-09 00:00:00	Anna	Lewska	A
13	5	Piękny Kraków	Polska	2018-10-12 00:00:00	Diana	Mazak	P

b) RezerwacjePotwierdzone

```

CREATE VIEW RezerwacjePotwierdzone
AS
SELECT
w.KRAJ,
w.DATA,
w.NAZWA,
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 = 'Z' OR r.STATUS = 'P';

```

	KRAJ	DATA	NAZWA	IMIE	NAZWISKO	STATUS
1	Polska	2018-10-12 00:00:00	Piękny Kraków	Jan	Nowak	P
2	Polska	2020-04-09 00:00:00	Wieliczka	Paweł	Wójcik	Z
3	Polska	2018-10-12 00:00:00	Piękny Kraków	Grzegorz	Szpak	Z
4	Polska	2020-04-09 00:00:00	Wieliczka	Kamil	Noga	Z
5	Polska	2020-05-18 00:00:00	Warszawa	Karol	Gawron	Z
6	Polska	2018-10-12 00:00:00	Piękny Kraków	Diana	Mazak	P

c) RezerwacjeWPrzyszłości

```

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

```

	KRAJ	DATA	NAZWA	IMIE	NAZWISKO	STATUS
1	Polska	2020-04-09 00:00:00	Wieliczka	Adam	Kowalski	N
2	Polska	2020-04-09 00:00:00	Wieliczka	Jan	Nowak	N
3	Polska	2020-05-18 00:00:00	Warszawa	Jan	Nowak	N
4	Polska	2020-04-09 00:00:00	Wieliczka	Paweł	Wójcik	Z
5	Polska	2020-05-18 00:00:00	Warszawa	Michał	Wiśniewski	A
6	Polska	2020-04-09 00:00:00	Wieliczka	Kamil	Noga	Z
7	Polska	2020-05-18 00:00:00	Warszawa	Karol	Gawron	Z

d)WycieczkiMiejsca

```
CREATE VIEW WycieczkiMiejsca
AS
SELECT
w.ID_WYCIECZKI,
w.NAZWA,
w.KRAJ,
w.DATA,
w.LICZBA_MIEJSC,
(w.LICZBA_MIEJSC -
(SELECT COUNT(*) FROM REZERWACJE r WHERE r.ID_WYCIECZKI = w.ID_WYCIECZKI AND
r.status <> 'A' ))
AS LICZBA_WOLNYCH_MIEJSC
FROM WYCIECZKI w
GROUP BY w. ID_WYCIECZKI, w.NAZWA, w.KRAJ, w.DATA, w.LICZBA_MIEJSC;
```

	ID_WYCIECZKI	NAZWA	KRAJ	DATA	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC
1	4	Wycieczka do Paryża	Francja	2019-12-09 00:00:00	31	29
2	5	Piękny Kraków	Polska	2018-10-12 00:00:00	54	51
3	6	Wieliczka	Polska	2020-04-09 00:00:00	35	31
4	7	Warszawa	Polska	2020-05-18 00:00:00	42	40

e)WycieczkiDostępne

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

```
CREATE VIEW RezerwacjeDoAnulowania
AS
SELECT w.ID_WYCIECZKI,
w.NAZWA,
r.STATUS,
o.IMIE,
```

```

o.NAZWISKO
FROM REZERWACJE r
JOIN OSOBY o ON r.ID_OSOBY = o.ID_OSOBY
JOIN WYCIECZKI w ON r.ID_WYCIECZKI = w.ID_WYCIECZKI
WHERE r.STATUS = 'N' AND w.DATA - CURRENT_DATE < 14;

```

	ID_WYCIECZKI	KRAJ	DATA	NAZWA	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC
1	6	Polska	2020-04-09 00:00:00	Wieliczka	35	31
2	7	Polska	2020-05-18 00:00:00	Warszawa	42	40

4. Obiekty

a) Wycieczka

```

CREATE OR REPLACE TYPE WYCIECZKA AS OBJECT
(
    id_wycieczki NUMBER,
    nazwa varchar2(100),
    kraj varchar2(50),
    data date,
    opis varchar2(200),
    liczba_miejsc NUMBER
);

CREATE OR REPLACE TYPE WYCIECZKI_TABELA AS TABLE OF WYCIECZKA;

```

b) Uczestnik

```

create TYPE UCZESTNIK AS OBJECT
(
    NAZWA          NVARCHAR2(100),
    KRAJ           NVARCHAR2(50),
    DATA          DATE,
    IMIE           NVARCHAR2(50),
    NAZWISKO       NVARCHAR2(50),
    STATUS         CHAR(1)
);

CREATE OR REPLACE TYPE UCZETNICY_TABELA AS TABLE OF UCZESTNIK;

```

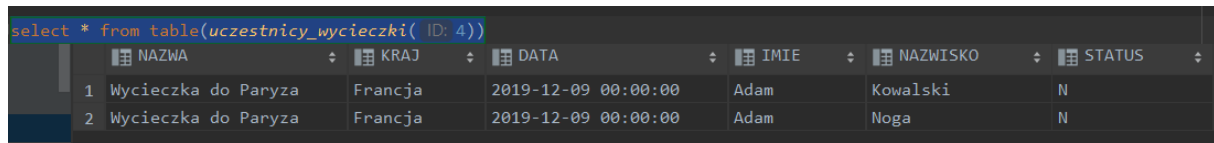
5. Funkcje

a) uczestnicy_wycieczki

```
CREATE OR REPLACE
FUNCTION uczestnicy_wycieczki(id INT)
return uczestnicy_tabela as v_ret uczestnicy_tabela;
istnieje
integer;
BEGIN
    SELECT COUNT(*) INTO istnieje FROM WYCIECZKI WHERE WYCIECZKI.ID_WYCIECZKI =
id;

    IF istnieje = 0 THEN
        raise_application_error(-20001, 'Nie znaleziono wycieczki o podanym id');
    END IF;

    SELECT UCZESTNIK(w.NAZWA,w.KRAJ, w.DATA, o.IMIE,
                    o.NAZWISKO, r.STATUS)
        BULK COLLECT INTO v_ret
    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 v_ret;
END uczestnicy_wycieczki;
```



The screenshot shows a SQL query result in a dark-themed interface. The query is `select * from table(uczestnicy_wycieczki(ID: 4))`. The result is a table with 7 columns: NAZWA, KRAJ, DATA, IMIE, NAZWISKO, and STATUS. There are two rows of data.

	NAZWA	KRAJ	DATA	IMIE	NAZWISKO	STATUS
1	Wycieczka do Paryza	Francja	2019-12-09 00:00:00	Adam	Kowalski	N
2	Wycieczka do Paryza	Francja	2019-12-09 00:00:00	Adam	Noga	N

b) Rezerwacje_osoby

```
CREATE OR REPLACE
FUNCTION rezerwacje_osoby(id_osoby INT)
return uczestnicy_tabela as v_ret uczestnicy_tabela;
istnieje
integer;
BEGIN
    SELECT COUNT(*) INTO istnieje FROM OSOBY WHERE OSOBY.ID_OSOBY =
rezerwacje_osoby.id_osoby;

    IF istnieje = 0 THEN
        raise_application_error(-20002, 'Nie znaleziono osoby o podanym id');
    END IF;

    SELECT uczestnik(w.NAZWA,w.KRAJ, w.DATA, o.IMIE,
                    o.NAZWISKO, r.STATUS)
        BULK COLLECT INTO v_ret
    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 = rezerwacje_osoby.id_osoby AND r.STATUS <> 'A'
    AND w.LICZBA_MIEJSC > (
        SELECT COUNT(*)
        FROM REZERWACJE
        WHERE r.STATUS <> 'A' AND r.ID_WYCIECZKI = w.ID_WYCIECZKI);
    return v_ret;
end rezerwacje_osoby;
```

```
select * from table(rezerwacje_osoby( ID_OSOBY: 6));
```

	NAZWA	KRAJ	DATA	IMIE	NAZWISKO	STATUS
1	Piękny Kraków	Polska	2018-10-12 00:00:00	Grzegorz	Szpak	Z

c)Dostępne_wycieczki

```
CREATE OR REPLACE
FUNCTION dostępne_wycieczki(kraj varchar2, data_od date, data_do date)
return WYCIECZKI_TABELA as v_ret WYCIECZKI_TABELA;

istnieje          Integer;
BEGIN
    SELECT COUNT(*) INTO istnieje FROM WYCIECZKI
    WHERE WYCIECZKI.KRAJ = dostępne_wycieczki.kraj AND WYCIECZKI.DATA >
dostępne_wycieczki.data_od AND
    WYCIECZKI.DATA < dostępne_wycieczki.data_do;

    IF istnieje = 0 THEN
        raise_application_error(-20002, 'Nie znaleziono wycieczki o podanym
id');
    END IF;

    SELECT WYCIECZKA(w.ID_WYCIECZKI, w.NAZWA,w.KRAJ, w.DATA, w.OPIS ,
        w.LICZBA_MIEJSC)
        BULK COLLECT INTO v_ret
    FROM WYCIECZKI w
    WHERE w.KRAJ = dostępne_wycieczki.kraj AND w.DATA >
dostępne_wycieczki.data_od AND
        w.DATA < dostępne_wycieczki.data_do;
    return v_ret;
end dostępne_wycieczki;
```

```
select * from table(DOSTĘPNE_WYCIECZKI( KRAJ: 'Polska', DATA_OD: TO_DATE('2019-11-09','YYYY-MM-DD'), DATA_DO: TO_DATE('2020-12-09','YYYY-MM-DD')));
```

1	6 Wieliczka	Polska	2020-04-09 00:00:00	Zadziwiająca kopalnia	35
2	7 Warszawa	Polska	2020-05-18 00:00:00	Stolica dla nas	42

6.Procedury

a)DODAJ_REZERWACJE

```
CREATE OR REPLACE PROCEDURE
DODAJ_REZERWACJE(ID_wycieczki WYCIECZKI.ID_WYCIECZKI%TYPE,
ID_osoby OSOBY.ID_OSOBY%TYPE)
AS
istnieje_osoba      INT;
dostępna_wycieczka  INT;
istnieje_rezerwacja INT;
ID_nowej_rezerwacji INT;
BEGIN
    -- sprawdź czy osoba istnieje, jeśli tak to istnieje_osoba > 0
    SELECT COUNT(*) INTO istnieje_osoba FROM OSOBY WHERE OSOBY.ID_OSOBY =
DODAJ_REZERWACJE.ID_osoby;

    IF istnieje_osoba = 0 THEN
        RAISE_APPLICATION_ERROR(-20002, 'Nie znaleziono osoby z tym ID.');
```

```
END IF;
    -- sprawdź czy istnieje wycieczka (czy jest w przyszłości i czy są wolne
```

miejsca)

```
SELECT COUNT(*)
INTO dostępna_wycieczka
FROM WYCIEZKIDOSTĘPNE
WHERE WYCIEZKIDOSTĘPNE.ID_WYCIEZKI = DODAJ_REZERWACJE.ID_wycieczki;

IF dostępna_wycieczka = 0 THEN
    RAISE_APPLICATION_ERROR(-20003, 'Wycieczka z tym ID nie jest dostępna');
END IF;

-- sprawdź czy istnieje rezerwacja
SELECT COUNT(*)
INTO istnieje_rezerwacja
FROM REZERWACJE
WHERE REZERWACJE.ID_WYCIEZKI = DODAJ_REZERWACJE.ID_wycieczki
    AND REZERWACJE.ID_OSOBY = DODAJ_REZERWACJE.ID_osoby;

IF istnieje_rezerwacja > 0 THEN
    RAISE_APPLICATION_ERROR(-20004, 'Rezerwacja owycieczki o podanym id
istnieje dla osoby o podanym id');
END IF;
SET TRANSACTION READ WRITE;

INSERT INTO REZERWACJE (ID_WYCIEZKI, ID_OSOBY, STATUS)
VALUES (DODAJ_REZERWACJE.ID_wycieczki, DODAJ_REZERWACJE.ID_osoby, 'N')
RETURNING NR_REZERWACJI INTO ID_nowej_rezerwacji;

INSERT INTO REZERWACJE_LOG (ID_REZERWACJI, DATA, STATUS)
VALUES (ID_nowej_rezerwacji, CURRENT_DATE, 'N');

COMMIT;
ROLLBACK;
END;
```

BEGIN

DODAJ_REZERWACJE (ID_WYCIEZKI: 7, ID_OSOBY: 3);

END;

	NR_REZERWACJI	ID_WYCIEZKI	ID_OSOBY	STATUS
1	61	7	2	N
2	2	4	1	N
3	3	5	2	P
4	4	6	3	Z
5	5	7	4	A
6	6	4	5	N
7	7	5	6	Z
8	8	6	7	Z
9	9	7	8	Z
10	10	4	9	A
11	11	5	10	P
12	21	6	1	N
13	41	6	2	N
14	81	7	3	N

b) ZMIEN_STATUS_REZERWACJI

```

CREATE OR REPLACE PROCEDURE
    ZMIEN_STATUS_REZERWACJI(ID_rezerwacji REZERWACJE.NR_REZERWACJI%TYPE,
                             status REZERWACJE.STATUS%TYPE)
AS
    stary_status REZERWACJE.STATUS%TYPE;
    wycieczka_istnieje INT;
BEGIN
    --sprawdź czy wycieczka istnieje
    SELECT COUNT(*)
    INTO wycieczka_istnieje
    FROM WYCIECZKIDOSTĘPNE wp
        JOIN REZERWACJE r ON wp.ID_WYCIECZKI = r.ID_WYCIECZKI
    WHERE r.ID_WYCIECZKI = ZMIEN_STATUS_REZERWACJI.ID_rezerwacji;

    IF wycieczka_istnieje = 0 THEN
        RAISE_APPLICATION_ERROR(-20005, 'Nie znaleziono wycieczki z tym ID');
    END IF;

    SELECT STATUS INTO stary_status FROM REZERWACJE r WHERE r.NR_REZERWACJI =
    ZMIEN_STATUS_REZERWACJI.ID_REZERWACJI;

    -- status 'N' (nowy) może być zmieniony na każdy status
    -- status 'P' (potwierdzony) może być zmieniony na status 'Z' lub 'A'
    -- status 'Z' (potwierdzony i zapłacony) może być zmieniony na status 'A'
    -- status 'A' nie może być zmieniony na inny status
    CASE
        WHEN stary_status IS NULL
        THEN RAISE_APPLICATION_ERROR(-20005, 'Rezerwacja z tym ID nie
istnieje');

        WHEN stary_status = 'A'
        THEN RAISE_APPLICATION_ERROR(-20006, 'Status A odwołanej rezerwacji
nie może być zmieniony');

        WHEN stary_status = 'P'
        THEN IF (ZMIEN_STATUS_REZERWACJI.status <> 'Z'
            AND ZMIEN_STATUS_REZERWACJI.status <> 'A') THEN
            RAISE_APPLICATION_ERROR(-20006,
                'Status potwierdzonej rezerwacji może być
zmieniony' ||
                'na "Z" (potwierdzona i zapłacona) lub "A"
(odwołana).');
        END IF;

        WHEN stary_status = 'Z'
        THEN IF ZMIEN_STATUS_REZERWACJI.status <> 'A' THEN
            RAISE_APPLICATION_ERROR(-20006,
                'Status potwierdzonej i zapłaconej
rezerwacji ("Z") może być zmieniony tylko' ||
                'na "A" (odwołany).');
        END IF;

    ELSE
        RAISE_APPLICATION_ERROR(-20999, 'Błąd wewnętrzny aplikacji');
    END CASE;

    SET TRANSACTION READ WRITE;

```

```

UPDATE REZERWACJE
SET STATUS = ZMIEN_STATUS_REZERWACJI.status
WHERE NR_REZERWACJI = ZMIEN_STATUS_REZERWACJI.ID_rezerwacji;

INSERT INTO REZERWACJE_LOG (ID_REZERWACJI, DATA, STATUS)
VALUES (ZMIEN_STATUS_REZERWACJI.ID_rezerwacji, CURRENT_DATE,
ZMIEN_STATUS_REZERWACJI.status);

COMMIT;
ROLLBACK;
END;

```

```
BEGIN
  ZMIEN_STATUS_REZERWACJI ( ID_REZERWACJI: 7, STATUS: 'A');
END;
```

1	61	7	2	N
2	2	4	1	N
3	3	5	2	P
4	4	6	3	Z
5	5	7	4	A
6	6	4	5	N
7	7	5	6	A
8	8	6	7	Z
9	9	7	8	Z
10	10	4	9	A
11	11	5	10	P
12	21	6	1	N
13	41	6	2	N
14	81	7	3	N

c) ZMIEN_LICZBE_MIEJSC

```

CREATE OR REPLACE PROCEDURE
  ZMIEN_LICZBE_MIEJSC(ID_wycieczki WYCIECZKI.ID_WYCIECZKI%TYPE,
                      nowa_liczba_miejsc WYCIECZKI.LICZBA_MIEJSC%TYPE)
AS
  zarezerwowane_miejsca INT;
BEGIN
  --jeśli wycieczka nie istnieje rzuć błąd
  SELECT w.LICZBA_MIEJSC - w.LICZBA_WOLNYCH_MIEJSC
  INTO zarezerwowane_miejsca
  FROM WYCIECZKIMIEJSCA w
  WHERE w.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC.ID_wycieczki;

  IF nowa_liczba_miejsc < 0 OR zarezerwowane_miejsca > nowa_liczba_miejsc
  THEN
    raise_application_error(-20007,
                          'Nowa liczba miejsc jest za mała (mniejsza od 0
lub mniejsza niż liczba zarezerwowanych miejsc)');
  END IF;
  SET TRANSACTION READ WRITE;
  UPDATE WYCIECZKI
  SET LICZBA_MIEJSC = ZMIEN_LICZBE_MIEJSC.nowa_liczba_miejsc

```

```

WHERE ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC.ID_wycieczki;

COMMIT;
ROLLBACK;

-- złap wyjątek i wypisz komunikat o błędzie
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono wycieczki z tym ID');
END;

```

```

BEGIN
    ZMIEN_LICZBE_MIEJSC( ID_WYCIECZKI: 6, NOWA_LICZBA_MIEJSC: 99);
END;

```

	ID_WYCIECZKI	NAZWA	KRAJ	DATA	OPIS	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC
1	4	Wycieczka do Paryża	Francja	2019-12-09 00:00:00	Ciekawa wycieczka	31	<nu
2	5	Piękny Kraków	Polska	2018-10-12 00:00:00	Najciekawa wycieczka	54	<nu
3	6	Wieliczka	Polska	2020-04-09 00:00:00	Zadziwiająca kopalnia	99	<nu
4	7	Warszawa	Polska	2020-05-18 00:00:00	Stolica dla nas	42	<nu

7. Tabela Rezerwacje_log i redundantne pole

```

CREATE TABLE REZERWACJE_LOG
(
    ID INT GENERATED ALWAYS AS IDENTITY NOT NULL
    , ID_REZERWACJI INT
    , DATA DATE
    , STATUS VARCHAR2(1)

    , CONSTRAINT REZERWACJE_LOG PRIMARY KEY
    (
        ID
    )
    ENABLE
);

```

	ID	ID_REZERWACJI	DATA	STATUS
1	21	61	2020-03-06 13:24:25	N
2	1	41	2020-03-05 21:21:10	N
3	41	81	2020-03-06 20:26:00	N
4	42	81	2020-03-06 20:26:00	N
5	43	7	2020-03-06 20:31:51	A
6	44	7	2020-03-06 20:31:51	A

```

ALTER TABLE WYCIECZKI
    ADD LICZBA_WOLNYCH_MIEJSC INT;

```

8. Procedura przelicz

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

	ID_WYCIECZKI	NAZWA	KRAJ	DATA	OPIS	LICZBA_MIEJSC	LICZBA_WOLNYCH_MIEJSC
1	4	Wycieczka do Paryża	Francja	2019-12-09 00:00:00	Ciekawa wycieczka	31	28
2	5	Piękny Kraków	Polska	2018-10-12 00:00:00	Najciekawsza wycieczka	54	51
3	7	Warszawa	Polska	2020-05-18 00:00:00	Stolica dla nas	42	38
4	6	Wieliczka	Polska	2020-04-09 00:00:00	Zadziwiająca kopalnia	99	96

9. Zmienione procedury

a) DODAJ_REZERWACJE2

```
CREATE OR REPLACE PROCEDURE
DODAJ_REZERWACJE2(ID_wycieczki WYCIECZKI.ID_WYCIECZKI%TYPE,
ID_osoby OSOBY.ID_OSOBY%TYPE)
AS
    istnieje_osoba      INT;
    dostępna_wycieczka  INT;
    istnieje_rezerwacja INT;
    ID_nowej_rezerwacji INT;
BEGIN
    -- sprawdź czy osoba istnieje, jeśli tak to istnieje_osoba > 0
    SELECT COUNT(*) INTO istnieje_osoba FROM OSOBY WHERE OSOBY.ID_OSOBY =
DODAJ_REZERWACJE2.ID_osoby;

    IF istnieje_osoba = 0 THEN
        RAISE_APPLICATION_ERROR(-20002, 'Nie znaleziono osoby z tym ID.');
```

```

        RAISE_APPLICATION_ERROR(-20004, 'Rezerwacja owycieczki o podanym id
istnieje dla osoby o podanym id');
    END IF;
    SET TRANSACTION READ WRITE;

    INSERT INTO REZERWACJE (ID_WYCIEZKI, ID_OSOBY, STATUS)
    VALUES (DODAJ_REZERWACJE2.ID_wycieczki, DODAJ_REZERWACJE2.ID_osoby, 'N')
    RETURNING NR_REZERWACJI INTO ID_nowej_rezerwacji;

    INSERT INTO REZERWACJE_LOG (ID_REZERWACJI, DATA, STATUS)
    VALUES (ID_nowej_rezerwacji, CURRENT_DATE, 'N');

    UPDATE WYCIEZKI
    SET LICZBA_WOLNYCH_MIEJSC = LICZBA_WOLNYCH_MIEJSC -1
    WHERE ID_WYCIEZKI = DODAJ_REZERWACJE2.ID_wycieczki;
    COMMIT;
    ROLLBACK;
END;

```

b) ZMIEN_STATUS_REZERWACJI

```

CREATE OR REPLACE PROCEDURE
    ZMIEN_STATUS_REZERWACJI2(ID_rezerwacji REZERWACJE.NR_REZERWACJI%TYPE,
                             status REZERWACJE.STATUS%TYPE)
AS
    stary_status REZERWACJE.STATUS%TYPE;
    wycieczka_istnieje INT;
    nowe_miejsce INT;
BEGIN
    --sprawdź czy wycieczka istnieje
    SELECT COUNT(*)
    INTO wycieczka_istnieje
    FROM WYCIEZKIDOSTĘPNE wp
        JOIN REZERWACJE r ON wp.ID_WYCIEZKI = r.ID_WYCIEZKI
    WHERE r.ID_WYCIEZKI = ZMIEN_STATUS_REZERWACJI2.ID_rezerwacji;

    IF wycieczka_istnieje = 0 THEN
        RAISE_APPLICATION_ERROR(-20005, 'Nie znaleziono wycieczki z tym ID');
    END IF;

    SELECT STATUS INTO stary_status FROM REZERWACJE r WHERE r.NR_REZERWACJI =
    ZMIEN_STATUS_REZERWACJI2.ID_REZERWACJI;

    -- status 'N' (nowy) może być zmieniony na każdy status
    -- status 'P' (potwierdzony) może być zmieniony na status 'Z' lub 'A'
    -- status 'Z' (potwierdzony i zapłacony) może być zmieniony na status 'A'
    -- status 'A' nie może być zmieniony na inny status
    CASE
        WHEN stary_status IS NULL
        THEN RAISE_APPLICATION_ERROR(-20005, 'Rezerwacja z tym ID nie
istnieje');

        WHEN stary_status = 'A'
        THEN RAISE_APPLICATION_ERROR(-20006, 'Status A odwołanej rezerwacji
nie może być zmieniony');

        WHEN stary_status = 'P'
        THEN IF (ZMIEN_STATUS_REZERWACJI2.status <> 'Z'

```



```

        AND ZMIEN_STATUS_REZERWACJI2.status <> 'A') THEN
        RAISE_APPLICATION_ERROR(-20006,
                                'Status potwierdzonej rezerwacji może być
zmieniony' ||
                                'na "Z" (potwierdzona i zapłacona) lub "A"
(odwołana).');
    END IF;

    WHEN stary_status = 'Z'
    THEN IF ZMIEN_STATUS_REZERWACJI2.status <> 'A' THEN
        RAISE_APPLICATION_ERROR(-20006,
                                'Status potwierdzonej i zapłaconej
rezerwacji ("Z") może być zmieniony tylko' ||
                                'na "A" (odwołany).');
    END IF;

    ELSE
        RAISE_APPLICATION_ERROR(-20999, 'Błąd wewnętrzny aplikacji');
    END CASE;

    IF ZMIEN_STATUS_REZERWACJI2.status = 'A' THEN
        nowe_miejsce := 1;
    ELSE
        nowe_miejsce := 0;
    END IF;

    SET TRANSACTION READ WRITE;
    UPDATE REZERWACJE
    SET STATUS = ZMIEN_STATUS_REZERWACJI2.status
    WHERE NR_REZERWACJI = ZMIEN_STATUS_REZERWACJI2.ID_rezerwacji;

    INSERT INTO REZERWACJE_LOG (ID_REZERWACJI, DATA, STATUS)
    VALUES (ZMIEN_STATUS_REZERWACJI2.ID_rezerwacji, CURRENT_DATE,
ZMIEN_STATUS_REZERWACJI2.status);

    UPDATE WYCIECZKI w
    SET LICZBA_WOLNYCH_MIEJSC = LICZBA_WOLNYCH_MIEJSC + nowe_miejsce
    WHERE ID_WYCIECZKI = (SELECT ID_WYCIECZKI FROM REZERWACJE r
                           WHERE r.NR_REZERWACJI =
ZMIEN_STATUS_REZERWACJI2.ID_rezerwacji);

    COMMIT;
    ROLLBACK;
END;

```

c) ZMIEN_LICZBĘ_MIEJSC

```

CREATE OR REPLACE PROCEDURE
    ZMIEN_LICZBE_MIEJSC2(ID_wycieczki WYCIECZKI.ID_WYCIECZKI%TYPE,
                        nowa_liczba_miejsc WYCIECZKI.LICZBA_MIEJSC%TYPE)
AS
    zarezerwowane_miejsca INT;
BEGIN
    --jeśli wycieczka nie istnieje rzuć błąd
    SELECT w.LICZBA_MIEJSC - w.LICZBA_WOLNYCH_MIEJSC
    INTO zarezerwowane_miejsca
    FROM WYCIECZKI w

```

```

WHERE w.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC2.ID_wycieczki;

IF nowa_liczba_miejsc < 0 OR zarezerwowane_miejsca > nowa_liczba_miejsc
THEN
    raise_application_error(-20007,
        'Nowa liczba miejsc jest za mała (mniejsza od 0
lub mniejsza niż liczba zarezerwowanych miejsc)');
END IF;
SET TRANSACTION READ WRITE;
UPDATE WYCIECZKI
SET LICZBA_MIEJSC = ZMIEN_LICZBE_MIEJSC2.nowa_liczba_miejsc,
    LICZBA_WOLNYCH_MIEJSC = LICZBA_WOLNYCH_MIEJSC +
(ZMIEN_LICZBE_MIEJSC2.nowa_liczba_miejsc - LICZBA_MIEJSC)
WHERE ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC2.ID_wycieczki;

COMMIT;
ROLLBACK;

-- złap wyjątek i wypisz komunikat o błędzie
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono wycieczki z tym ID');
END;

```

10. Zmieniona funkcje

a)Dostępne_wycieczki2

```

CREATE OR REPLACE
FUNCTION dostępne_wycieczki2(kraj varchar2, data_od date, data_do date)
return WYCIECZKI_TABELA as v_ret WYCIECZKI_TABELA;

istnieje Integer;
BEGIN
    SELECT COUNT(*) INTO istnieje FROM WYCIECZKI
    WHERE WYCIECZKI.KRAJ = dostępne_wycieczki2.kraj AND WYCIECZKI.DATA >
dostępne_wycieczki2.data_od AND
        WYCIECZKI.DATA < dostępne_wycieczki2.data_do;

    IF istnieje = 0 THEN
        raise_application_error(-20002, 'Nie znaleziono osoby o podanym id');
    END IF;

    SELECT WYCIECZKA(w.ID_WYCIECZKI, w.NAZWA,w.KRAJ, w.DATA, w.OPIS ,
        w.LICZBA_MIEJSC)
        BULK COLLECT INTO v_ret
    FROM WYCIECZKI w
    WHERE w.KRAJ = dostępne_wycieczki2.kraj AND w.DATA >
dostępne_wycieczki2.data_od AND
        w.DATA < dostępne_wycieczki2.data_do AND w.LICZBA_WOLNYCH_MIEJSC >
0;

    return v_ret;
end dostępne_wycieczki2;

```

11. Zmienione Widoki

a)WycieczkiMiejsca2

```
CREATE VIEW WycieczkiMiejsca2
AS
SELECT
W.ID_WYCIECZKI,
W.NAZWA,
W.KRAJ,
W.DATA,
W.LICZBA_MIEJSC,
W.LICZBA_WOLNYCH_MIEJSC
FROM WYCIECZKI w;
```

b)WycieczkiDostępne2

```
CREATE VIEW WycieczkiDostępne2
AS
SELECT
W.ID_WYCIECZKI,
W.KRAJ,
W.DATA,
W.NAZWA,
W.LICZBA_MIEJSC,
W.LICZBA_WOLNYCH_MIEJSC
FROM WYCIECZKI w
WHERE w.DATA > CURRENT_DATE;
```

12. Triggery

a)DODANIE_REZERWACJI_TRIGGER

```
CREATE OR REPLACE TRIGGER DODANIE_REZERWACJI_TRIGGER
AFTER INSERT
ON REZERWACJE
FOR EACH ROW
BEGIN
INSERT INTO REZERWACJE_LOG (ID_REZERWACJI, DATA, STATUS)
VALUES (:NEW.NR_REZERWACJI, CURRENT_DATE, :NEW.STATUS);

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

b) ZMIANA_STATUSU_TRIGGER

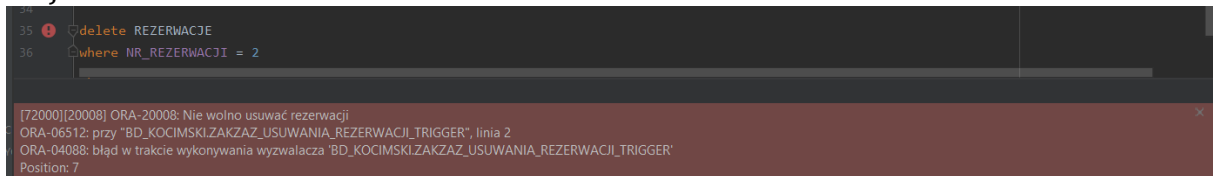
```
CREATE OR REPLACE TRIGGER ZMIANA_STATUSU_TRIGGER
AFTER UPDATE
ON REZERWACJE
FOR EACH ROW
DECLARE
    nowe_miejsce INT;
BEGIN
    INSERT INTO REZERWACJE_LOG (ID_REZERWACJI, DATA, STATUS)
    VALUES (:NEW.NR_REZERWACJI, CURRENT_DATE, :NEW.STATUS);

    --jeśli zmieniamy status na 'A' to zwiększa się liczba wolnych miejsc
    -- w pozostałych przypadkach liczba wolnych miejsc pozostaje bez zmian
    IF :NEW.STATUS = 'A' THEN
        nowe_miejsce := 1;
    ELSE
        nowe_miejsce := 0;
    END IF;

    UPDATE WYCIECZKI w
    SET LICZBA_WOLNYCH_MIEJSC = LICZBA_WOLNYCH_MIEJSC + nowe_miejsce
    WHERE w.ID_WYCIECZKI = :NEW.ID_WYCIECZKI;
END zmiana_statusu_trigger;
```

c) ZAKAZ_USUWANIA_REZERWACJI_TRIGGER

```
CREATE OR REPLACE TRIGGER ZAKAZ_USUWANIA_REZERWACJI_TRIGGER
BEFORE DELETE
ON REZERWACJE
FOR EACH ROW
BEGIN
    raise_application_error(-20008, 'Nie wolno usuwać rezerwacji');
END;
```



13. Zmienione procedury

a) DODAJ_REZERWACJE3

```
CREATE OR REPLACE PROCEDURE
DODAJ_REZERWACJE3(ID_wycieczki WYCIECZKI.ID_WYCIECZKI%TYPE,
ID_osoby OSOBY.ID_OSOBY%TYPE)
AS
    istnieje_osoba INT;
    dostępna_wycieczka INT;
    istnieje_rezerwacja INT;
BEGIN
    -- sprawdź czy osoba istnieje, jeśli tak to istnieje_osoba > 0
    SELECT COUNT(*) INTO istnieje_osoba FROM OSOBY WHERE OSOBY.ID_OSOBY =
DODAJ_REZERWACJE3.ID_osoby;
```

```

IF istnieje_osoba = 0 THEN
    RAISE_APPLICATION_ERROR(-20002, 'Nie znaleziono osoby z tym ID.');
```

-- sprawdź czy istnieje wycieczka (czy jest w przyszłości i czy są wolne miejsca)

```

SELECT COUNT(*)
INTO dostępna_wycieczka
FROM WYCIECZKIDOSTĘPNE
WHERE WYCIECZKIDOSTĘPNE.ID_WYCIECZKI = DODAJ_REZERWACJE3.ID_wycieczki;

IF dostępna_wycieczka = 0 THEN
    RAISE_APPLICATION_ERROR(-20003, 'Wycieczka z tym ID nie jest dostępna');
```

-- sprawdź czy istnieje rezerwacja

```

SELECT COUNT(*)
INTO istnieje_rezerwacja
FROM REZERWACJE
WHERE REZERWACJE.ID_WYCIECZKI = DODAJ_REZERWACJE3.ID_wycieczki
    AND REZERWACJE.ID_OSOBY = DODAJ_REZERWACJE3.ID_osoby;

IF istnieje_rezerwacja > 0 THEN
    RAISE_APPLICATION_ERROR(-20004, 'Rezerwacja owycieczki o podanym id
istnieje dla osoby o podanym id');
```

```

END IF;
SET TRANSACTION READ WRITE;

INSERT INTO REZERWACJE (ID_WYCIECZKI, ID_OSOBY, STATUS)
VALUES (DODAJ_REZERWACJE3.ID_wycieczki, DODAJ_REZERWACJE3.ID_osoby, 'N');

COMMIT;
ROLLBACK;
END;
```

b) ZMIEN_STATUS_REZERWACJI3

```

CREATE OR REPLACE PROCEDURE
    ZMIEN_STATUS_REZERWACJI3(ID_rezerwacji REZERWACJE.NR_REZERWACJI%TYPE,
                             status REZERWACJE.STATUS%TYPE)
AS
    stary_status REZERWACJE.STATUS%TYPE;
    wycieczka_istnieje INT;
    nowe_miejsce INT;
BEGIN
    -- sprawdź czy wycieczka istnieje
    SELECT COUNT(*)
    INTO wycieczka_istnieje
    FROM WYCIECZKIDOSTĘPNE wp
        JOIN REZERWACJE r ON wp.ID_WYCIECZKI = r.ID_WYCIECZKI
    WHERE r.ID_WYCIECZKI = ZMIEN_STATUS_REZERWACJI3.ID_rezerwacji;

    IF wycieczka_istnieje = 0 THEN
        RAISE_APPLICATION_ERROR(-20005, 'Nie znaleziono wycieczki z tym ID');
```

```

    END IF;

    SELECT STATUS INTO stary_status FROM REZERWACJE r WHERE r.NR_REZERWACJI =
```

```

ZMIEN_STATUS_REZERWACJI3.ID_REZERWACJI;

-- status 'N' (nowy) może być zmieniony na każdy status
-- status 'P' (potwierdzony) może być zmieniony na status 'Z' lub 'A'
-- status 'Z' (potwierdzony i zapłacony) może być zmieniony na status 'A'
-- status 'A' nie może być zmieniony na inny status
CASE
    WHEN stary_status IS NULL
    THEN RAISE_APPLICATION_ERROR(-20005, 'Rezerwacja z tym ID nie
istnieje');

    WHEN stary_status = 'A'
    THEN RAISE_APPLICATION_ERROR(-20006, 'Status A odwołanej rezerwacji
nie może być zmieniony');

    WHEN stary_status = 'P'
    THEN IF (ZMIEN_STATUS_REZERWACJI3.status <> 'Z'
    AND ZMIEN_STATUS_REZERWACJI3.status <> 'A') THEN
        RAISE_APPLICATION_ERROR(-20006,
        'Status potwierdzonej rezerwacji może być
zmieniony' ||
        'na "Z" (potwierdzona i zapłacona) lub "A"
(odwołana).');
    END IF;

    WHEN stary_status = 'Z'
    THEN IF ZMIEN_STATUS_REZERWACJI3.status <> 'A' THEN
        RAISE_APPLICATION_ERROR(-20006,
        'Status potwierdzonej i zapłaconej
rezerwacji ("Z") może być zmieniony tylko' ||
        'na "A" (odwołany).');
    END IF;

ELSE
    RAISE_APPLICATION_ERROR(-20999, 'Błąd wewnętrzny aplikacji');
END CASE;

IF ZMIEN_STATUS_REZERWACJI3.status = 'A' THEN
    nowe_miejsce := 1;
ELSE
    nowe_miejsce := 0;
END IF;

SET TRANSACTION READ WRITE;
UPDATE REZERWACJE
SET STATUS = ZMIEN_STATUS_REZERWACJI3.status
WHERE NR_REZERWACJI = ZMIEN_STATUS_REZERWACJI3.ID_rezerwacji;

COMMIT;
ROLLBACK;
END;

```

C) ZMIEN_LICZBE_MIEJSC3

```

CREATE OR REPLACE PROCEDURE
    ZMIEN_LICZBE_MIEJSC3(ID_wycieczki WYCIECZKI.ID_WYCIECZKI%TYPE,
        nowa_liczba_miejsc WYCIECZKI.LICZBA_MIEJSC%TYPE)
AS

```

```

zarezerwowane_miejsca INT;
BEGIN
    --jeśli wycieczka nie istnieje rzuć błąd
    SELECT w.LICZBA_MIEJSC - w.LICZBA_WOLNYCH_MIEJSC
    INTO zarezerwowane_miejsca
    FROM WYCIECZKI w
    WHERE w.ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC3.ID_wycieczki;

    IF ZMIEN_LICZBE_MIEJSC3.nowa_liczba_miejsc < 0 OR zarezerwowane_miejsca >
    ZMIEN_LICZBE_MIEJSC3.nowa_liczba_miejsc
    THEN
        raise_application_error(-20007,
        'Nowa liczba miejsc jest za mała (mniejsza od 0
lub mniejsza niż liczba zarezerwowanych miejsc');
    END IF;
    SET TRANSACTION READ WRITE;
    UPDATE WYCIECZKI
    SET LICZBA_MIEJSC = ZMIEN_LICZBE_MIEJSC3.nowa_liczba_miejsc
    WHERE ID_WYCIECZKI = ZMIEN_LICZBE_MIEJSC3.ID_wycieczki;

    COMMIT;
    ROLLBACK;

    -- złap wyjątek i wypisz komunikat o błędzie
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nie znaleziono wycieczki z tym ID');
END;

```