

# Sprawozdanie – Oracle

Autor: Adrian Żerebiec

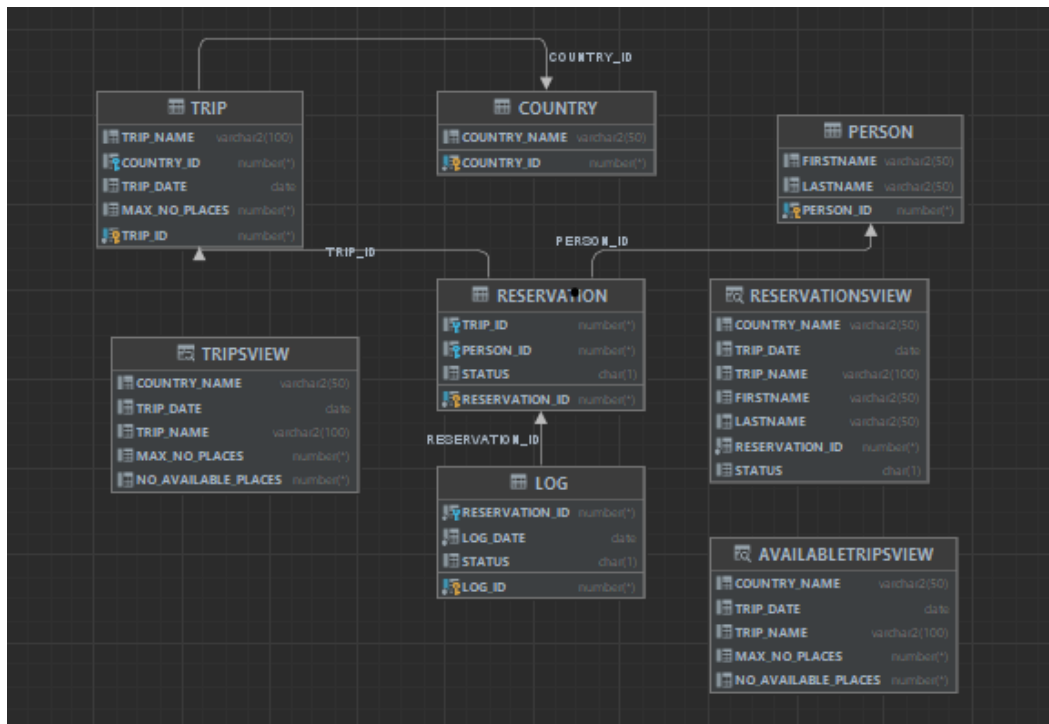
## Spis treści

1. Schemat bazy danych .....	4
2. Zadania 1 oraz zadanie 2 .....	4
2.1 Tworzenie Tabel .....	4
2.1.1 PERSON .....	4
2.1.2 TRIP .....	4
2.1.3 RESERVATION .....	5
2.2 Warunki Intergralnościowe .....	5
2.2.1 TRIP .....	5
2.2.2 RESERVATION .....	5
2.3 Wstawianie danych .....	5
2.3.1 COUNTRY .....	5
2.3.2 TRIPS .....	6
2.3.3 PERSON .....	7
2.3.4 RESERVATION .....	8
3. Zadanie 3 .....	9
3.1 RESERVATIONVIEW .....	9
3.2 TRIPVIEW .....	10
3.3 AVAILABLETRIPS .....	10
4. Zadanie 4 .....	11
4.1 TRIPPARTICIPANTS .....	11
4.2 PERSONREVERVATIONS .....	12
4.3AVAILABLETRIPS .....	13
*Informacja dodatkowa .....	14
5. Zadanie 5 .....	14
5.1 ADDRESERVATION .....	14
5.2 MODIFYRESERVATIONSTATUS .....	15
5.3 MODIFYNOPLACES .....	17
6. Zadanie 6 .....	17
6.1 Tworzenie tabeli LOG .....	17
6.2 Warunki Intergralnościowe .....	18

6.3 Dodawanie danych .....	18
6.4 Procedury .....	19
6.4.1 ADDRESERVATION .....	19
6.4.2 MODIFYRESERVATIONSTATUS .....	20
6.5 Przykładowe wpisy do dziennika po kilku operacjach.....	21
7. Zadanie 7 .....	21
7.1 Triggery.....	21
7.1.1 TGADDINGRESERVATION .....	21
7.1.2 TGMODIFYRESERVATION .....	22
7.1.3 TGDELETESTOP .....	22
7.2 Zmodyfikowane procedury modyfikujące.....	22
7.2.1 ADDRESERVATION_2 .....	22
7.2.2 MODIFYRESERVATIONSTATUS_2 .....	23
8. Zadanie 8 .....	24
8.1 Triggery.....	24
8.1.1 TGCHECKAVAILABLEPLACES .....	24
8.1.2 TGCHECKCHANGESTATUS .....	24
8.2 Zmodyfikowane procedury modyfikujące dane.....	25
8.2.1 ADDRESERVATION_3 .....	25
8.2.2 MODIFYRESERVATIONSTATUS_3 .....	25
9. Zadanie 9 .....	26
9.1 Dodanie pola NO_AVAILABLE_PLACES.....	26
9.2 Widoki.....	26
9.2.1 RESERVATIONVIEW_4 .....	26
9.2.1 TRIPSVIEW_4 .....	27
9.2.3 AVAILABLETRIPS_4 .....	27
9.3 Obliczanie ilości miejsc .....	28
9.4 Zmodyfikowane procedury .....	28
9.4.1 ADDRESERVATION_4 .....	28
9.4.2 MODIFYRESERVATIONSTATUS_4 .....	29
9.4.3 MODIFYNOPLACES_4 .....	30
10. Zadanie 10 .....	30
10.1 Triggery.....	30
10.1.1 TGUPDATEAVAILABLE.....	30
10.1.2 TGCHANGINGSTATUS.....	30
10.1.3 Zmiana liczby miejsc podczas dodania rezerwacji .....	31

10.2 Zmodyfikowane procedury .....	31
10.2.1 ADDRESERVATION_5 .....	31
10.2.2 MODIFYRESERVATIONSTATUS_5 .....	32
10.2.3 MODIFYNOPLACES_5.....	33

## 1. Schemat bazy danych



## 2. Zadania 1 oraz zadanie 2

### 2.1 Tworzenie Tabel

#### 2.1.1 PERSON

```
CREATE TABLE PERSON
(
    PERSON_ID INT GENERATED ALWAYS AS IDENTITY NOT NULL,
    FIRSTNAME VARCHAR(50),
    LASTNAME VARCHAR(50),
    CONSTRAINT PERSON_PK PRIMARY KEY (PERSON_ID) ENABLE
);
```

#### 2.1.2 TRIP

```
CREATE TABLE TRIP
(
    TRIP_ID INT GENERATED ALWAYS AS IDENTITY NOT NULL,
    TRIP_NAME VARCHAR(100),
    COUNTRY_ID INT,
    TRIP_DATE DATE,
    MAX_NO_PLACES INT,
    CONSTRAINT TRIP_PK PRIMARY KEY (TRIP_ID) ENABLE
);
```

### 2.1.3 RESERVATION

```
CREATE TABLE RESERVATION
(
    RESERVATION_ID INT GENERATED ALWAYS AS IDENTITY NOT NULL,
    TRIP_ID        INT,
    PERSON_ID      INT,
    STATUS         CHAR(1),
    CONSTRAINT RESERVATION_PK PRIMARY KEY (RESERVATION_ID)
ENABLE
);
```

## 2.2 Warunki Integralnościowe

### 2.2.1 TRIP

```
ALTER TABLE TRIP
    ADD CONSTRAINT TRIP_FK1 FOREIGN KEY
        (COUNTRY_ID) REFERENCES COUNTRY (COUNTRY_ID) ENABLE;
ALTER TABLE TRIP
    ADD CONSTRAINT TRIP_FK1 FOREIGN KEY
        (COUNTRY_ID) REFERENCES COUNTRY (COUNTRY_ID) ENABLE;
```

### 2.2.2 RESERVATION

```
ALTER TABLE RESERVATION
    ADD CONSTRAINT RESERVATION_FK1 FOREIGN KEY
        (PERSON_ID) REFERENCES PERSON (PERSON_ID) ENABLE;

ALTER TABLE RESERVATION
    ADD CONSTRAINT RESERVATION_FK2 FOREIGN KEY
        (TRIP_ID) REFERENCES TRIP (TRIP_ID) ENABLE;

ALTER TABLE RESERVATION
    ADD CONSTRAINT RESERVATION_CHK1 CHECK
        (STATUS IN ('N', 'P', 'C')) ENABLE;
```

## 2.3 Wstawianie danych

### 2.3.1 COUNTRY



```
INSERT INTO COUNTRY (COUNTRY_NAME)
VALUES ('FRANCJA');

INSERT INTO COUNTRY (COUNTRY_NAME)
VALUES ('POLSKA');

INSERT INTO COUNTRY (COUNTRY_NAME)
VALUES ('HISZPANIA');

INSERT INTO COUNTRY (COUNTRY_NAME)
VALUES ('WŁOCHY');
```

Wynik:

	 COUNTRY_ID ▾	 COUNTRY_NAME ▾
1	1	Francja
2	2	Polska
3	3	Hiszpania
4	4	Włochy

### 2.3.2 TRIPS

```
INSERT INTO TRIP(TRIP_NAME, COUNTRY_ID, TRIP_DATE,
MAX_NO_PLACES)
VALUES ('WYCIECZKA DO PARYZA', 1, TO_DATE('2023-09-12', 'YYYY-
MM-DD'), 30);






INSERT INTO TRIP(TRIP_NAME, COUNTRY_ID, TRIP_DATE,
MAX_NO_PLACES)
VALUES ('PIĘKNY KRAKÓW', 2, TO_DATE('2023-07-03', 'YYYY-MM-
DD'), 25);

INSERT INTO TRIP(TRIP_NAME, COUNTRY_ID, TRIP_DATE,
MAX_NO_PLACES)
VALUES ('ZNÓW DO FRANCJI', 1, TO_DATE('2023-05-01', 'YYYY-MM-
DD'), 21);

INSERT INTO TRIP(TRIP_NAME, COUNTRY_ID, TRIP_DATE,
MAX_NO_PLACES)
VALUES ('HEL', 2, TO_DATE('2023-03-01', 'YYYY-MM-DD'), 15);

INSERT INTO TRIP(TRIP_NAME, COUNTRY_ID, TRIP_DATE,
MAX_NO_PLACES)
VALUES ('KOŁOSEUM', 4, TO_DATE('2023-06-25', 'YYYY-MM-DD'),
85);
```

Wynik:

	 TRIP_ID ▾	 TRIP_NAME ▾	 COUNTRY_ID ▾	 TRIP_DATE ▾	 MAX_NO_PLACES ▾
1	1	Wycieczka do Paryza	1	2023-09-12	30
2	2	Piękny Kraków	2	2023-07-03	25
3	3	Znów do Francji	1	2023-05-01	21
4	4	Hel	2	2023-03-01	20

### 2.3.3 PERSON

```
INSERT INTO PERSON (FIRSTNAME, LASTNAME)
VALUES ('JAN', 'NOWAK');

INSERT INTO PERSON (FIRSTNAME, LASTNAME)
VALUES ('JAN', 'KOWALSKI');

INSERT INTO PERSON (FIRSTNAME, LASTNAME)
VALUES ('JAN', 'NOWAKOWSKI');

INSERT INTO PERSON (FIRSTNAME, LASTNAME)
VALUES ('ADAM', 'KOWALSKI');

INSERT INTO PERSON (FIRSTNAME, LASTNAME)
VALUES ('NOVAK', 'NOWAK');

INSERT INTO PERSON (FIRSTNAME, LASTNAME)
VALUES ('PIOTR', 'PIOTROWSKI');


INSERT INTO PERSON (FIRSTNAME, LASTNAME)
VALUES ('MAREK', 'ADAMOWSKI');

INSERT INTO PERSON (FIRSTNAME, LASTNAME)
VALUES ('SZYMON', 'RYT');

INSERT INTO PERSON (FIRSTNAME, LASTNAME)
VALUES ('KAROL', 'KOWALSKI');

INSERT INTO PERSON (FIRSTNAME, LASTNAME)
VALUES ('ARTUR', 'KAROLAK');
```

Wynik:

	 PERSON_ID ▾	 FIRSTNAME ▾	 LASTNAME ▾
1	1	Jan	Nowak
2	2	Jan	Kowalski
3	3	Jan	Nowakowski
4	4	Adam	Kowalski
5	5	Novak	Nowak
6	6	Piotr	Piotrowski
7	7	Marek	Adamowski
8	8	Szymon	Ryt
9	9	Karol	Kowalski
10	10	Artur	Karolak

### 2.3.4 RESERVATION

```
INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (1, 1, 'P');

INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (1, 2, 'N');

INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (2, 10, 'C');

INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (3, 7, 'P');

INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (1, 6, 'N');

INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (4, 8, 'C');





INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (1, 9, 'N');

INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (2, 4, 'C');

INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (2, 4, 'P');

INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (3, 9, 'P');
```

Wynik:

	 RESERVATION_ID ▾	 TRIP_ID ▾	 PERSON_ID ▾	 STATUS ▾
1	1	1	1	P
2	2	1	2	N
3	3	2	10	C
4	4	3	7	P
5	5	1	6	N
6	6	4	8	C
7	7	1	9	N
8	8	2	4	C
9	9	2	4	P
10	10	3	9	P



## 3. Zadanie 3

### 3.1 RESERVATIONVIEW

```
CREATE OR REPLACE VIEW RESERVATIONVIEW
AS
SELECT C.COUNTRY_NAME,
       T.TRIP_DATE,
       T.TRIP_NAME,
       P.FIRSTNAME,
       P.LASTNAME,
       R.RESERVATION_ID,
       R.STATUS
FROM TRIP T
      INNER JOIN RESERVATION R
        ON R.TRIP_ID = T.TRIP_ID
      INNER JOIN PERSON P
        ON P.PERSON_ID = R.PERSON_ID
      INNER JOIN COUNTRY C
        ON C.COUNTRY_ID = T.COUNTRY_ID;
```

**Wynik:**

	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	FIRSTNAME	LASTNAME	RESERVATION_ID	STATUS
1	Francja	2023-09-12	Wycieczka do Paryza	Jan	Nowak	1	P
2	Francja	2023-09-12	Wycieczka do Paryza	Jan	Kowalski	2	N
3	Francja	2023-09-12	Wycieczka do Paryza	Jan	Kowalski	21	C
4	Polska	2023-07-03	Piękny Kraków	Jan	Kowalski	22	N
5	Polska	2023-07-03	Piękny Kraków	Adam	Kowalski	8	C
6	Polska	2023-07-03	Piękny Kraków	Adam	Kowalski	9	P
7	Francja	2023-09-12	Wycieczka do Paryza	Piotr	Piotrowski	5	N
8	Francja	2023-05-01	Znów do Francji	Marek	Adamowski	4	P
9	Polska	2023-03-01	Hel	Szymon	Ryt	6	C
10	Francja	2023-09-12	Wycieczka do Paryza	Karol	Kowalski	7	N
11	Francja	2023-05-01	Znów do Francji	Karol	Kowalski	10	P
12	Polska	2023-07-03	Piękny Kraków	Artur	Karolak	3	C

**Widok wyświetla wszystkie ważne informacje dotyczące rezerwacji.**

## 3.2 TRIPSVIEW

```
CREATE OR REPLACE VIEW TRIPSVIEW
AS
SELECT COUNTRY_NAME,
       TRIP_DATE,
       TRIP_NAME,
       MAX_NO_PLACES,
       MAX_NO_PLACES - (SELECT COUNT(*)
                        FROM RESERVATION R
                        WHERE R.STATUS = 'P'
                        AND R.TRIP_ID = T.TRIP_ID)
       AS NO_AVAILABLE_PLACES
FROM TRIP T
     INNER JOIN COUNTRY C
     ON T.COUNTRY_ID = C.COUNTRY_ID;
```

Wynik:

	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	Francja	2023-05-01	Znów do Francji	21	19
2	Francja	2023-09-12	Wycieczka do Paryża	30	29
3	Polska	2023-03-01	Hel	20	20
4	Polska	2023-07-03	Piękny Kraków	25	24

Widok wyświetla ważne informacje o wszystkich wycieczkach w tym liczbę obecnie dostępnych miejsc, za zarezerwowane uznajemy tylko te już opłacone.

## 3.3 AVAILABLETRIPS

```
CREATE OR REPLACE VIEW AVAILABLETRIPSVIEW
AS
SELECT COUNTRY_NAME,
       TRIP_DATE,
       TRIP_NAME,
       MAX_NO_PLACES,
       NO_AVAILABLE_PLACES
FROM TRIPSVIEW
WHERE TRIP_DATE > SYSDATE
     AND NO_AVAILABLE_PLACES > 0;
```

Przykładowe wykonanie:

	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	Francja	2023-09-12	Wycieczka do Paryża	30	29
2	Francja	2023-05-01	Znów do Francji	21	19
3	Polska	2023-07-03	Piękny Kraków	25	24

Widok pokazuje tylko te wycieczki, które są obecnie dostępne, czyli te w których są miejsca oraz data jest z przyszłości.

## 4. Zadanie 4

### 4.1 TRIPPARTICIPANTS

```
CREATE OR REPLACE TYPE TRIPPARTICIPANTSROW AS OBJECT
(
    COUNTRY_NAME    VARCHAR2(50),
    TRIP_DATE       DATE,
    TRIP_NAME       VARCHAR2(100),
    FIRSTNAME       VARCHAR2(50),
    LASTNAME        VARCHAR2(50),
    RESERVATION_ID  INT,
    STATUS          CHAR(1)
);

CREATE OR REPLACE TYPE TRIPPARTICIPANTSTABLE AS TABLE OF
TRIPPARTICIPANTSROW;

CREATE OR REPLACE FUNCTION TRIPPARTICIPANTS(
    SEARCHED_TRIP_ID INT
)
RETURN TRIPPARTICIPANTSTABLE
IS
    RETURNTABLE TRIPPARTICIPANTSTABLE;
BEGIN
    SELECT TRIPPARTICIPANTSROW(
        RV.COUNTRY_NAME,
        RV.TRIP_DATE,
        RV.TRIP_NAME,
        FIRSTNAME,
        LASTNAME,
        RESERVATION_ID,
        STATUS) BULK COLLECT
    INTO RETURNTABLE
    FROM RESERVATIONSVIEW RV
        INNER JOIN TRIP T ON RV.TRIP_NAME = T.TRIP_NAME
    WHERE T.TRIP_ID = SEARCHED_TRIP_ID;
    RETURN RETURNTABLE;
END;
```

**Przykładowe wykonanie:**

```
SELECT *
FROM TABLE (TRIPPARTICIPANTS(1));
```

**Wynik:**

	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	FIRSTNAME	LASTNAME	RESERVATION_ID	STATUS
1	FRANCJA	2023-09-12	WYCIECZKA DO PARYZA	JAN	NOWAK	1	P
2	FRANCJA	2023-09-12	WYCIECZKA DO PARYZA	JAN	KOWALSKI	2	N
3	FRANCJA	2023-09-12	WYCIECZKA DO PARYZA	PIOTR	PIOTROWSKI	5	N

**Procedura pokazuje uczestników wycieczki, numer rezerwacji i jej status**

## 4.2 PERSONREVERVATIONS

```
CREATE OR REPLACE FUNCTION PERSONREVERVATIONS (
    SEARCHED_PERSON_ID INT
)
RETURN TRIPPARTICIPANTSTABLE
IS
    RETURNTABLE TRIPPARTICIPANTSTABLE;
BEGIN
    SELECT TRIPPARTICIPANTSROW (RV.COUNTRY_NAME,
                                RV.TRIP_DATE,
                                TRIP_NAME,
                                RV.FIRSTNAME,
                                RV.LASTNAME,
                                RESERVATION_ID,
                                STATUS) BULK COLLECT
        INTO RETURNTABLE
    FROM RESERVATIONSVIEW RV
        INNER JOIN PERSON P ON P.FIRSTNAME = RV.FIRSTNAME
    AND P.LASTNAME = RV.LASTNAME
    WHERE P.PERSON_ID = SEARCHED_PERSON_ID;
    RETURN RETURNTABLE;
END;
```

Przykładowe wykonanie:

```
SELECT *
FROM TABLE ( PERSONREVERVATIONS (4) );
```

Wynik:

	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	FIRSTNAME	LASTNAME	RESERVATION_ID	STATUS
1	POLSKA	2023-07-03	PIĘKNY KRAKÓW	ADAM	KOWALSKI	8	C
2	POLSKA	2023-03-01	HEL	ADAM	KOWALSKI	9	P

Procedura pokazuje rezerwacje wycieczek, datę wycieczki, dane osoby, która dokonała rezerwacji oraz ID rezerwacji i jej status

### 4.3 AVAILABLETRIPS

```
CREATE OR REPLACE TYPE AVAILABLETRIPSROW AS OBJECT
(
    COUNTRY                VARCHAR2(50),
    TRIP_DATE              DATE,
    TRIP_NAME              VARCHAR2(100),
    NO_AVAILABLE_PLACES    INT
);

CREATE OR REPLACE TYPE AVAILABLETRIPSTABLE AS TABLE OF
AVAILABLETRIPSROW;

CREATE OR REPLACE FUNCTION AVAILABLETRIPS(
    SEARCHED_COUNTRY VARCHAR,
    SEARCHED_DATE_FROM DATE,
    SEARCHED_DATE_TO DATE
)
RETURN AVAILABLETRIPSTABLE
IS
    RETURNTABLE AVAILABLETRIPSTABLE;
BEGIN
    SELECT AVAILABLETRIPSROW(COUNTRY_NAME,
                            TRIP_DATE,
                            TRIP_NAME,
                            NO_AVAILABLE_PLACES) BULK COLLECT
    INTO RETURNTABLE
    FROM TRIPVIEW TV
    WHERE TV.COUNTRY_NAME = SEARCHED_COUNTRY
        AND (TV.TRIP_DATE BETWEEN SEARCHED_DATE_FROM AND
SEARCHED_DATE_TO);
    RETURN RETURNTABLE;
END;
```

#### Przykładowe wykonanie:

```
SELECT *
FROM TABLE (AVAILABLETRIPS('FRANCJA', TO_DATE('2022/08/15',
'YYYY/MM/DD'), TO_DATE('2023/08/29', 'YYYY/MM/DD ')) );
```

#### Wynik:

	COUNTRY	TRIP_DATE	TRIP_NAME	NO_AVAILABLE_PLACES
1	FRANCJA	2023-05-01	ZNÓW DO FRANCJI	19

Procedura pokazuje wycieczki, które spełniają wszystkie warunki, to jest: kraj, oraz daty

## \*Informacja dodatkowa

Przyjąłem, iż jako zajęte miejsce liczymy opłacone rezerwacje czyli Paid. Rezerwacje New nie są wliczane przeze mnie jako zajęte miejsce, dopiero po opłaceniu zostają one wliczane jako miejsce zajęte stąd np. w widoku TRIPSVIEW od maksymalnej liczby miejsc odejmuje tylko te opłacone przez osobę. Również pokazuję działanie procedur tylko w zadaniu 5, gdyż ich działanie później jest analogiczne.

## 5. Zadanie 5

### 5.1 ADDRESERVATION

```
CREATE OR REPLACE PROCEDURE ADDRESERVATION(
    TRIP_ID_ADD IN INT, PERSON_ID_ADD IN INT
) AS
    NEW_TRIP_DATE      DATE;
    NO_AVAILABLE       INT;
    NEW_RESERVATION_ID INT;
BEGIN
    SELECT TRIP_DATE
    INTO NEW_TRIP_DATE
    FROM TRIP T
    WHERE T.TRIP_ID = TRIP_ID_ADD;

    SELECT NO_AVAILABLE_PLACES
    INTO NO_AVAILABLE
    FROM TRIPSVIEW VT
    INNER JOIN TRIP T2 ON VT.TRIP_NAME = T2.TRIP_NAME
    WHERE T2.TRIP_ID = TRIP_ID_ADD;

    IF NEW_TRIP_DATE > TRUNC(SYSDATE) AND NO_AVAILABLE > 0
    THEN
        INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (TRIP_ID_ADD, PERSON_ID_ADD, 'N');
        SELECT MAX(RESERVATION_ID)
        INTO NEW_RESERVATION_ID
        FROM RESERVATION;
        INSERT INTO LOG(RESERVATION_ID, LOG_DATE, STATUS)
VALUES (NEW_RESERVATION_ID, TRUNC(SYSDATE), 'N');
    END IF;
END;
```

#### Przykładowe wykonanie:

```
DECLARE
    TRIP_ID_ADD NUMBER := 2;
    PERSON_ID_ADD NUMBER := 2;
BEGIN
    ADDRESERVATION(
        TRIP_ID_ADD => TRIP_ID_ADD,
        PERSON_ID_ADD => PERSON_ID_ADD
    );
END;
```

## Wynik:

	RESERVATION_ID	TRIP_ID	PERSON_ID	STATUS
1	41	1	3	P
2	42	5	2	N
3	61	5	8	P
4	1	1	1	P
5	2	1	2	N
6	3	2	10	C
7	4	3	7	P
8	5	1	6	N
9	6	4	8	C
10	7	3	9	N
11	8	2	4	C
12	9	4	4	P
13	10	3	9	P
14	21	2	2	N

Procedura dodaje rezerwację dla danej osoby, jeśli data wycieczki jest późniejsza niż obecna oraz liczba miejsc jest większa od 0. Jak widać rezerwacja z ID 21 to dodana przez procedurę rezerwacja.

## 5.2 MODIFYRESERVATIONSTATUS

```
CREATE OR REPLACE PROCEDURE MODIFYRESERVATIONSTATUS (
    RESERVATION_ID_MOD IN INT, STATUS_CHANGE IN CHAR
) AS
    NO_AVAILABLE     INT;
    TRIP_ID_AC       INT;
    CURRENT_STATUS   CHAR;
BEGIN
    SELECT TRIP_ID
    INTO TRIP_ID_AC
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;

    SELECT NO_AVAILABLE_PLACES
    INTO NO_AVAILABLE
    FROM TRIPSVIEW VT
        INNER JOIN TRIP T2 ON VT.TRIP_NAME = T2.TRIP_NAME
    WHERE T2.TRIP_ID = TRIP_ID_AC;

    SELECT STATUS
    INTO CURRENT_STATUS
```

```

FROM RESERVATION
WHERE RESERVATION_ID = RESERVATION_ID_MOD;

IF (CURRENT_STATUS = 'N' AND STATUS_CHANGE = 'P' AND
NO_AVAILABLE > 0) OR
((CURRENT_STATUS = 'N' OR CURRENT_STATUS = 'P' OR
CURRENT_STATUS = 'C') AND STATUS_CHANGE = 'C') OR
((CURRENT_STATUS = 'C') AND STATUS_CHANGE = 'P' AND
NO_AVAILABLE > 0) THEN
UPDATE RESERVATION
SET STATUS = STATUS_CHANGE
WHERE RESERVATION_ID = RESERVATION_ID_MOD;
END IF;
END;

```





## Wywołanie:

```

DECLARE
RESERVATION_ID_MOD1  NUMBER := 21;
STATUS_CHANGE1 CHAR(1) := 'P';
BEGIN
MODIFYRESERVATIONSTATUS(
    RESERVATION_ID_MOD => RESERVATION_ID_MOD1,
    STATUS_CHANGE => STATUS_CHANGE1
);
END;

```

## Wynik:

	 RESERVATION_ID ▾	 TRIP_ID ▾	 PERSON_ID ▾	 STATUS ▾
1	41	1	3	P
2	42	5	2	N
3	61	5	8	P
4	1	1	1	P
5	2	1	2	N
6	3	2	10	C
7	4	3	7	P
8	5	1	6	N
9	6	4	8	C
10	7	3	9	N
11	8	2	4	C
12	9	4	4	P
13	10	3	9	P
14	21	2	2	P



Procedura zmienia status rezerwacji ale pod pewnymi warunkami: zawsze można zmienić na Cancelled, z Cancelled można zmienić na Paid jeśli jest odpowiednia liczba miejsc oraz New można zmienić na Paid jeśli są jeszcze wolne miejsca.

### 5.3 MODIFYNOPLACES

```
CREATE OR REPLACE PROCEDURE MODIFYNOPLACES(TRIP_ID_MOD IN INT,
NEW_NUMBER_OF_PLACES IN INT) AS NO_RESERVED INT;
BEGIN
    SELECT T.MAX_NO_PLACES - TV.NO_AVAILABLE_PLACES
    INTO NO_RESERVED
    FROM TRIPVIEW TV
    INNER JOIN TRIP T ON TV.TRIP_NAME = T.TRIP_NAME
    WHERE T.TRIP_ID = TRIP_ID_MOD;
    IF NEW_NUMBER_OF_PLACES >= NO_RESERVED THEN
        UPDATE TRIP
        SET MAX_NO_PLACES = NEW_NUMBER_OF_PLACES
        WHERE TRIP_ID = TRIP_ID_MOD;
    END IF;
END;
```

**Przykładowe wykonanie:**

```
DECLARE
    S_TRIP_ID      NUMBER := 5;
    NEW_NO_PLACES  NUMBER := 50;
BEGIN
    MODIFYNOPLACES(
        TRIP_ID_MOD => S_TRIP_ID,
        NEW_NUMBER_OF_PLACES => NEW_NO_PLACES
    );
END;
```

**Wynik:**

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES
1	1	WYCIECZKA DO PARYZA	1	2023-09-12	30
2	2	PIĘKNY KRAKÓW	2	2023-07-03	25
3	3	ZNÓW DO FRANCJI	1	2023-05-01	21
4	4	HEL	2	2023-03-01	15
5	5	KOLOSEUM	4	2023-06-25	50

Procedura zmienia maksymalną liczbę miejsc w wybranej wycieczce

## 6. Zadanie 6

### 6.1 Tworzenie tabeli LOG

```
CREATE TABLE LOG
(
    LOG_ID          INT GENERATED ALWAYS AS IDENTITY NOT NULL,
    RESERVATION_ID  INT                                NOT NULL,
```

```

LOG_DATE          DATE                                NOT NULL,
STATUS            CHAR(1),
CONSTRAINT LOG_PK PRIMARY KEY (LOG_ID) ENABLE
);

```

## 6.2 Warunki Intergralnościowe

```

ALTER TABLE LOG
  ADD CONSTRAINT LOG_CHK1 CHECK
    (STATUS IN ('N', 'P', 'C')) ENABLE;

ALTER TABLE LOG
  ADD CONSTRAINT LOG_FK1 FOREIGN KEY
    (RESERVATION_ID) REFERENCES RESERVATION
    (RESERVATION_ID) ENABLE;

```

## 6.3 Dodawanie danych

```

INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (1, '2023-02-01', 'P');

INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (2, '2023-01-03', 'N');

INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (3, '2023-01-01', 'P');

INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (4, '2022-12-01', 'C');

INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (5, '2023-02-07', 'P');

INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (6, '2023-01-12', 'C');

INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (7, '2023-03-17', 'P');

INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (8, '2023-02-07', 'N');

INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (9, '2023-03-10', 'C');

INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (10, '2023-01-23', 'N');

```

## Wynik:

	LOG_ID	RESERVATION_ID	LOG_DATE	STATUS
1	1	1	2023-02-01	P
2	2	2	2023-01-03	N
3	3	3	2023-01-01	P
4	4	4	2022-12-01	C
5	5	5	2023-02-07	P
6	6	6	2023-01-12	C
7	7	7	2023-03-17	P
8	8	8	2023-02-07	N
9	9	9	2023-03-10	C
10	10	10	2023-01-23	N

Tabela zawiera informacje o zmianach w rezerwacjach

## 6.4 Procedury

### 6.4.1 ADDRESERVATION

```
CREATE OR REPLACE PROCEDURE ADDRESERVATION(
    TRIP_ID_ADD IN INT, PERSON_ID_ADD IN INT
) AS
    NEW_TRIP_DATE      DATE;
    NO_AVAILABLE       INT;
    NEW_RESERVATION_ID INT;
BEGIN
    SELECT TRIP_DATE
    INTO NEW_TRIP_DATE
    FROM TRIP T
    WHERE T.TRIP_ID = PERSON_ID_ADD;

    SELECT NO_AVAILABLE_PLACES
    INTO NO_AVAILABLE
    FROM TRIPVIEW VT
    INNER JOIN TRIP T2 ON VT.TRIP_NAME = T2.TRIP_NAME
    WHERE T2.TRIP_ID = PERSON_ID_ADD;

    IF NEW_TRIP_DATE > TRUNC(SYSDATE) AND NO_AVAILABLE > 0
    THEN
        INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
        VALUES (TRIP_ID_ADD, PERSON_ID_ADD, 'N');
        SELECT MAX(RESERVATION_ID)
        INTO NEW_RESERVATION_ID
        FROM RESERVATION;
        INSERT INTO LOG(RESERVATION_ID, LOG_DATE, STATUS)
        VALUES (NEW_RESERVATION_ID, TRUNC(SYSDATE), 'N');
    END IF;
END;
```

Dodałem tylko linijkę odpowiadającą za wpis do dziennika

#### 6.4.2 MODIFYRESERVATIONSTATUS

```
CREATE OR REPLACE PROCEDURE MODIFYRESERVATIONSTATUS (
    RESERVATION_ID_MOD IN INT, STATUS_CHANGE IN CHAR
) AS
    NO_AVAILABLE      INT;
    TRIP_ID_AC        INT;
    CURRENT_STATUS    CHAR;
BEGIN
    SELECT TRIP_ID
    INTO TRIP_ID_AC
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;

    SELECT NO_AVAILABLE
    INTO NO_AVAILABLE
    FROM TRIPVIEW VT
        INNER JOIN TRIP T2 ON VT.TRIP_NAME = T2.TRIP_NAME
    WHERE T2.TRIP_ID = TRIP_ID_AC;

    SELECT STATUS
    INTO CURRENT_STATUS
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;

    IF (CURRENT_STATUS = 'N' AND STATUS_CHANGE = 'P' AND
NO_AVAILABLE > 0) OR
        ((CURRENT_STATUS = 'N' OR CURRENT_STATUS = 'P' OR
CURRENT_STATUS = 'C') AND STATUS_CHANGE = 'C') OR
        ((CURRENT_STATUS = 'C') AND STATUS_CHANGE = 'P' AND
NO_AVAILABLE > 0) THEN
        UPDATE RESERVATION
        SET STATUS = STATUS_CHANGE
        WHERE RESERVATION_ID = RESERVATION_ID_MOD;
        INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
VALUES (RESERVATION_ID_MOD, TRUNC(SYSDATE), STATUS_CHANGE);
    END IF;
END;
```

Dodałem tylko linijkę odpowiadającą za wpis do dziennika

## 6.5 Przykładowe wpisy do dziennika po kilku operacjach

	LOG_ID	RESERVATION_ID	LOG_DATE	STATUS
1	41	41	2023-04-04	N
2	42	42	2023-04-04	N
3	61	41	2023-04-04	P
4	62	61	2023-04-04	N
5	63	61	2023-04-04	P
6	64	21	2023-04-04	P
7	1	1	2023-02-01	P
8	2	2	2023-01-03	N
9	3	3	2023-01-01	P
10	4	4	2022-12-01	C
11	5	5	2023-02-07	P
12	6	6	2023-01-12	C
13	7	7	2023-03-17	P
14	8	8	2023-02-07	N
15	9	9	2023-03-10	C
16	10	10	2023-01-23	N
17	21	21	2023-04-02	N

## 7. Zadanie 7

### 7.1 Triggery

#### 7.1.1 TGADDINGRESERVATION

```
CREATE OR REPLACE TRIGGER TGADDINGRESERVATION
  AFTER INSERT
  ON RESERVATION
  FOR EACH ROW
DECLARE
  RES_ID INT;
  STAT CHAR;
BEGIN
  RES_ID := :NEW.RESERVATION_ID;
  STAT := :NEW.STATUS;
  INSERT INTO LOG VALUES (DEFAULT, RES_ID, TRUNC(SYSDATE),
STAT);
END;
```

Trigger dodaje zapis do dziennika po dodaniu rezerwacji

### 7.1.2 TGMODIFYRESERVATION

```
CREATE OR REPLACE TRIGGER TGMODIFYRESERVATION
  AFTER UPDATE OF STATUS ON RESERVATION
  FOR EACH ROW
DECLARE
  V_RESERVATION_ID RESERVATION.RESERVATION_ID%TYPE;
BEGIN
  V_RESERVATION_ID := :NEW.RESERVATION_ID;
  IF :OLD.STATUS != :NEW.STATUS THEN
    INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
      VALUES (V_RESERVATION_ID, TRUNC(SYSDATE),
:NEW.STATUS);
  END IF;
END;
```

Trigger dodaje wpis do dziennika po zmianie statusu w tabeli RESERVATION

### 7.1.3 TGDELETESTOP

```
CREATE OR REPLACE TRIGGER TGDELETESTOP
  BEFORE DELETE
  ON RESERVATION
BEGIN
  RAISE_APPLICATION_ERROR(-20001, 'RESERVATIONS CANT BE
DELETED');
END;
```

Trigger zapobiega usuwaniu rezerwacji

## 7.2 Zmodyfikowane procedury modyfikujące

### 7.2.1 ADDRESERVATION\_2

```
CREATE OR REPLACE PROCEDURE ADDRESERVATION_2(
  TRIP_ID_ADD IN INT, PERSON_ID_ADD IN INT
) AS
  NEW_TRIP_DATE      DATE;
  NO_AVAILABLE       INT;
  NEW_RESERVATION_ID INT;
BEGIN
  SELECT TRIP_DATE
  INTO NEW_TRIP_DATE
  FROM TRIP T
  WHERE T.TRIP_ID = TRIP_ID_ADD;

  SELECT NO_AVAILABLE_PLACES
  INTO NO_AVAILABLE
  FROM TRIPVIEW VT
    INNER JOIN TRIP T2 ON VT.TRIP_NAME = T2.TRIP_NAME
  WHERE T2.TRIP_ID = TRIP_ID_ADD;
```

```

        IF NEW_TRIP_DATE > TRUNC(SYSDATE) AND NO_AVAILABLE > 0
THEN
        INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (TRIP_ID_ADD, PERSON_ID_ADD, 'N');
        SELECT MAX(RESERVATION_ID)
        INTO NEW_RESERVATION_ID
        FROM RESERVATION;
    ELSE
        RAISE_APPLICATION_ERROR(-20001, 'OUT OF DATE');
    END IF;
END;

```

Procedura dodaje rezerwacje dla wybranej wycieczki jeśli są wolne miejsca i jeśli wycieczka jeszcze się nie odbyła. Trigger dopisuje do dziennika działanie

## 7.2.2 MODIFYRESERVATIONSTATUS\_2

```

CREATE OR REPLACE PROCEDURE MODIFYRESERVATIONSTATUS_2(
    RESERVATION_ID_MOD IN INT, STATUS_CHANGE IN CHAR
) AS
    NO_AVAILABLE      INT;
    TRIP_ID_AC        INT;
    CURRENT_STATUS     CHAR;
BEGIN
    SELECT TRIP_ID
    INTO TRIP_ID_AC
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;

    SELECT NO_AVAILABLE_PLACES
    INTO NO_AVAILABLE
    FROM TRIPSVIEW VT
        INNER JOIN TRIP T2 ON VT.TRIP_NAME = T2.TRIP_NAME
    WHERE T2.TRIP_ID = TRIP_ID_AC;

    SELECT STATUS
    INTO CURRENT_STATUS
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;
    IF (CURRENT_STATUS = 'N' AND STATUS_CHANGE = 'P' AND
NO_AVAILABLE > 0) OR
        ((CURRENT_STATUS = 'N' OR CURRENT_STATUS = 'P' OR
CURRENT_STATUS = 'C') AND STATUS_CHANGE = 'C') OR
        ((CURRENT_STATUS = 'C') AND STATUS_CHANGE = 'P' AND
NO_AVAILABLE > 0) THEN
        UPDATE RESERVATION
        SET STATUS = STATUS_CHANGE
        WHERE RESERVATION_ID = RESERVATION_ID_MOD;
    END IF;
END;

```

Procedura sprawdza czy można dokonać zmiany, jeśli została dokonana to trigger powoduje wpis do dziennika.

## 8. Zadanie 8

### 8.1 Triggery

#### 8.1.1 TGCHECKAVAILABLEPLACES

```
CREATE OR REPLACE TRIGGER TGCHECKAVAILABLEPLACES
  BEFORE INSERT
  ON RESERVATION
  FOR EACH ROW
DECLARE
  V_MAX_PLACES      TRIP.MAX_NO_PLACES%TYPE;
  V_RESERVED_PLACES INT;
BEGIN
  SELECT MAX_NO_PLACES INTO V_MAX_PLACES FROM TRIP WHERE
TRIP_ID = :NEW.TRIP_ID;
  SELECT COUNT(*) INTO V_RESERVED_PLACES FROM RESERVATION
WHERE TRIP_ID = :NEW.TRIP_ID AND STATUS = 'P';

  IF V_MAX_PLACES - V_RESERVED_PLACES < 1 THEN
    RAISE_APPLICATION_ERROR(-20001, 'THERE ARE NO
AVAILABLE PLACES FOR THIS TRIP.');
```

Trigger sprawdza czy są dostępne miejsca i tylko wtedy można zmienić status na P, gdyż jak przyjąłem tylko status P jest liczony jako rezerwacja już dokonana.

#### 8.1.2 TGCHECKCHANGESTATUS

```
CREATE OR REPLACE TRIGGER TGCHECKCHANGESTATUS
  BEFORE UPDATE OF STATUS ON RESERVATION
  FOR EACH ROW
DECLARE
  TRIP_MAX_PLACES INT;
  NO_RESERVED INT;
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
  SELECT MAX_NO_PLACES INTO TRIP_MAX_PLACES
FROM TRIP
WHERE TRIP_ID = :NEW.TRIP_ID;

  SELECT COUNT(*) INTO NO_RESERVED
FROM RESERVATION
WHERE TRIP_ID = :NEW.TRIP_ID AND STATUS = 'P';

  IF ((:OLD.STATUS = 'N' AND :NEW.STATUS = 'P' AND NO_RESERVED
```



```

< TRIP_MAX_PLACES)
    OR (:OLD.STATUS IN ('N', 'P', 'C') AND :NEW.STATUS =
'C')
    OR (:OLD.STATUS = 'C' AND :NEW.STATUS = 'P' AND
NO_RESERVED < TRIP_MAX_PLACES))
    THEN
        NULL;
    ELSE
        RAISE_APPLICATION_ERROR(-20001, 'STATUS CHANGE NOT
ALLOWED');
    END IF;
    COMMIT;
END;

```

Trigger sprawdza czy możliwa jest zmiana statusu rezerwacji

## 8.2 Zmodyfikowane procedury modyfikujące dane

### 8.2.1 ADDRESERVATION\_3

```

CREATE OR REPLACE PROCEDURE ADDRESERVATION_3(
    TRIP_ID_ADD IN INT, PERSON_ID_ADD IN INT
) AS
    NEW_TRIP_DATE      DATE;
    NEW_RESERVATION_ID INT;
BEGIN
    SELECT TRIP_DATE
    INTO NEW_TRIP_DATE
    FROM TRIP T
    WHERE T.TRIP_ID = PERSON_ID_ADD;

    IF NEW_TRIP_DATE > TRUNC(SYSDATE) THEN
        INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (TRIP_ID_ADD, PERSON_ID_ADD, 'N');
        SELECT MAX(RESERVATION_ID)
        INTO NEW_RESERVATION_ID
        FROM RESERVATION;
    END IF;
END;

```

Procedura dodaje rezerwacje, sprawdzanie oraz wpis do dziennika realizowane są przez trigger.

### 8.2.2 MODIFYRESERVATIONSTATUS\_3

```

CREATE OR REPLACE PROCEDURE MODIFYRESERVATIONSTATUS_3(
    RESERVATION_ID_MOD IN INT, STATUS_CHANGE IN CHAR
) AS
    TRIP_ID_AC      INT;
    CURRENT_STATUS CHAR;
BEGIN
    SELECT TRIP_ID

```

```

    INTO TRIP_ID_AC
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;

    SELECT STATUS
    INTO CURRENT_STATUS
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;

    UPDATE RESERVATION
    SET STATUS = STATUS_CHANGE
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;
END;

```

Procedura zmienia status wycieczki, za sprawdzenie poprawności oraz wpis do dziennika odpowiadają triggerzy.

## 9. Zadanie 9

### 9.1 Dodanie pola NO\_AVAILABLE\_PLACES

```

ALTER TABLE TRIP
    ADD NO_AVAILABLE_PLACES INT;

```

Wynik:

TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	WYCIECZKA DO PARYZA	1	2023-09-12	30	<null>
2	PIĘKNY KRAKÓW	2	2023-07-03	25	25
3	ZNÓW DO FRANCJI	1	2023-05-01	21	<null>
4	HEL	2	2023-03-01	20	<null>
5	KOLOSEUM	4	2023-06-25	85	<null>

### 9.2 Widoki

#### 9.2.1 RESERVATIONVIEW\_4

```

CREATE OR REPLACE VIEW RESERVATIONVIEW_4
AS
SELECT C.COUNTRY_NAME,
       T.TRIP_DATE,
       T.TRIP_NAME,
       P.FIRSTNAME,
       P.LASTNAME,
       R.RESERVATION_ID,
       R.STATUS
FROM TRIP T
     INNER JOIN RESERVATION R
        ON R.TRIP_ID = T.TRIP_ID
     INNER JOIN PERSON P
        ON P.PERSON_ID = R.PERSON_ID
     INNER JOIN COUNTRY C
        ON C.COUNTRY_ID = T.COUNTRY_ID;

```

## Wynik:

	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	FIRSTNAME	LASTNAME	RESERVATION_ID	STATUS
1	FRANCJA	2023-09-12	WYCIECZKA DO PARYŻA	JAN	NOWAK	1	P
2	FRANCJA	2023-09-12	WYCIECZKA DO PARYŻA	JAN	KOWALSKI	2	N
3	POLSKA	2023-07-03	PIĘKNY KRAKÓW	JAN	KOWALSKI	21	N
4	POLSKA	2023-07-03	PIĘKNY KRAKÓW	ADAM	KOWALSKI	8	C
5	POLSKA	2023-03-01	HEL	ADAM	KOWALSKI	9	P
6	FRANCJA	2023-09-12	WYCIECZKA DO PARYŻA	PIOTR	PIOTROWSKI	5	N
7	FRANCJA	2023-05-01	ZNÓW DO FRANCJI	MAREK	ADAMOWSKI	4	P
8	POLSKA	2023-03-01	HEL	SZYMON	RYT	6	C
9	FRANCJA	2023-05-01	ZNÓW DO FRANCJI	KAROL	KOWALSKI	7	N
10	FRANCJA	2023-05-01	ZNÓW DO FRANCJI	KAROL	KOWALSKI	10	P
11	POLSKA	2023-07-03	PIĘKNY KRAKÓW	ARTUR	KAROLAK	3	C

## Analogiczny widok do wcześniejszego

### 9.2.1 TRIPSVIEW\_4

```
CREATE OR REPLACE VIEW TRIPSVIEW_4
AS
SELECT COUNTRY_NAME,
       TRIP_DATE,
       TRIP_NAME,
       MAX_NO_PLACES,
       NO_AVAILABLE_PLACES
FROM TRIP T
     INNER JOIN COUNTRY C
       ON T.COUNTRY_ID = C.COUNTRY_ID;
```

## Wynik:

	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	FRANCJA	2023-09-12	WYCIECZKA DO PARYŻA	30	<null>
2	FRANCJA	2023-05-01	ZNÓW DO FRANCJI	21	<null>
3	POLSKA	2023-03-01	HEL	20	<null>
4	POLSKA	2023-07-03	PIĘKNY KRAKÓW	25	25
5	WŁOCHY	2023-06-25	KOLOSEUM	85	<null>

Nie obliczamy jak w TRIPSVIEW ilości miejsc ale pokazujemy dodane pole

### 9.2.3 AVAILABLETRIPS\_4

```
CREATE OR REPLACE VIEW AVAILABLETRIPSVIEW_4
AS
SELECT COUNTRY_NAME,
       TRIP_DATE,
       TRIP_NAME,
       MAX_NO_PLACES,
       NO_AVAILABLE_PLACES
FROM TRIPSVIEW
WHERE TRIP_DATE > SYSDATE
     AND NO_AVAILABLE_PLACES > 0;
```

## Wynik:

	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	FRANCJA	2023-09-12	WYCIEZKA DO PARYZA	30	29
2	FRANCJA	2023-05-01	ZNÓW DO FRANCJI	21	19
3	POLSKA	2023-07-03	PIĘKNY KRAKÓW	25	25
4	WŁOCHY	2023-06-25	KOLOSEUM	85	85

Pokazujemy tylko te które są dostępne

## 9.3 Obliczanie ilości miejsc

```
CREATE OR REPLACE PROCEDURE CALCULATEAVAILABLEPLACES (
    S_TRIP_ID INT
)
AS
    NO_RESERVED_PLACES INT;
BEGIN
    SELECT COUNT(*)
    INTO NO_RESERVED_PLACES
    FROM RESERVATIONVIEW_4 RV
        INNER JOIN TRIP T ON T.TRIP_NAME = RV.TRIP_NAME
    WHERE T.TRIP_ID = S_TRIP_ID AND STATUS = 'P';

    UPDATE TRIP
    SET TRIP.NO_AVAILABLE_PLACES = TRIP.MAX_NO_PLACES -
NO_RESERVED_PLACES
    WHERE TRIP_ID = S_TRIP_ID;
END;
```

Tak jak wcześniej przyjąłem jako miejsce już zajęte, to jest zarezerwowane przyjąłem tylko te, które są już opłacone, czyli te o statusie Paid.

## 9.4 Zmodyfikowane procedury

### 9.4.1 ADDRESERVATION\_4

```
CREATE OR REPLACE PROCEDURE ADDRESERVATION_4 (
    TRIP_ID_ADD IN INT, PERSON_ID_ADD IN INT
) AS
    NEW_TRIP_DATE DATE;
    NEW_RESERVATION_ID INT;
    NO_AVAILABLE INT;
BEGIN
    SELECT TRIP_DATE
    INTO NEW_TRIP_DATE
    FROM TRIP T
    WHERE T.TRIP_ID = TRIP_ID_ADD;

    SELECT NO_AVAILABLE_PLACES
    INTO NO_AVAILABLE
    FROM TRIP T
```

```

WHERE T.TRIP_ID = TRIP_ID_ADD;

IF NEW_TRIP_DATE > TRUNC(SYSDATE) AND NO_AVAILABLE>0 THEN
    INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (TRIP_ID_ADD, PERSON_ID_ADD, 'N');
    SELECT MAX(RESERVATION_ID)
    INTO NEW_RESERVATION_ID
    FROM RESERVATION;
    CALCULATEAVAILABLEPLACES(TRIP_ID_ADD);
ELSE
    RAISE_APPLICATION_ERROR(-20001, 'OUT OF DATE');
END IF;
END;

```

Dodano obliczanie miejsc i bezpośrednie pobieranie ilości z tabeli TRIP, wpisy do dziennika realizuje trigger dlatego niepotrzebne jest wykonywanie tego w procedurze. Podobnie jest z ilością miejsc, tak na prawdę w tym przypadku nie trzeba sprawdzać, gdyż robi to trigger.

#### 9.4.2 MODIFYRESERVATIONSTATUS\_4

```

CREATE OR REPLACE PROCEDURE MODIFYRESERVATIONSTATUS_4 (
    RESERVATION_ID_MOD IN INT, STATUS_CHANGE IN CHAR
) AS
    TRIP_ID_AC          INT;
    CURRENT_STATUS      CHAR;
BEGIN
    SELECT TRIP_ID
    INTO TRIP_ID_AC
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;

    SELECT STATUS
    INTO CURRENT_STATUS
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;

    UPDATE RESERVATION
    SET STATUS = STATUS_CHANGE
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;
    CALCULATEAVAILABLEPLACES(TRIP_ID_AC);
END;

```

Procedura zmienia status wycieczki. Dodatkowo na końcu oblicza nową liczbę miejsc dostępnych

### 9.4.3 MODIFYNOPLACES\_4

```
CREATE OR REPLACE PROCEDURE MODIFYNOPLACES_4
  (TRIP_ID_MOD IN INT, NEW_NUMBER_OF_PLACES IN INT) AS
  NO_RESERVED INT;
BEGIN
  SELECT T2.MAX_NO_PLACES - VT.NO_AVAILABLE_PLACES
  INTO NO_RESERVED
  FROM TRIPVIEW_4 VT
       INNER JOIN TRIP T2 ON VT.TRIP_NAME = T2.TRIP_NAME
  WHERE T2.TRIP_ID = TRIP_ID_MOD;

  IF NEW_NUMBER_OF_PLACES >= NO_RESERVED THEN
    UPDATE TRIP
    SET MAX_NO_PLACES = NEW_NUMBER_OF_PLACES
    WHERE TRIP_ID = TRIP_ID_MOD;
    CALCULATEAVAILABLEPLACES(TRIP_ID_MOD);
  END IF;
END;
```

Procedura tym razem dodatkowo oblicza liczbę pozostałych miejsc po zmianie

## 10. Zadanie 10

### 10.1 Triggery

#### 10.1.1 TGUPDATEAVAILABLE

```
CREATE OR REPLACE TRIGGER TGUPDATEAVAILABLE
  BEFORE UPDATE OF MAX_NO_PLACES
  ON TRIP
  FOR EACH ROW
BEGIN
  :NEW.NO_AVAILABLE_PLACES := :OLD.NO_AVAILABLE_PLACES +
  :NEW.MAX_NO_PLACES - :OLD.MAX_NO_PLACES;
END;
```

Trigger aktualizuje liczbę obecnie dostępnych miejsc

#### 10.1.2 TGCHANGINGSTATUS

```
CREATE OR REPLACE TRIGGER TGCHANGINGSTATUS
  BEFORE UPDATE OF STATUS ON RESERVATION
  FOR EACH ROW
DECLARE
  TRIP_PLACES INT;
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
  SELECT NO_AVAILABLE_PLACES INTO TRIP_PLACES
  FROM TRIP
```

```

WHERE TRIP_ID = :NEW.TRIP_ID;

IF ((:OLD.STATUS = 'N' AND :NEW.STATUS = 'P' AND
TRIP_PLACES>0)
    OR (:OLD.STATUS IN ('N', 'P', 'C') AND :NEW.STATUS =
'C')
    OR (:OLD.STATUS = 'C' AND :NEW.STATUS = 'P' AND
TRIP_PLACES>0))
THEN
    IF :NEW.STATUS = 'P' THEN
        UPDATE TRIP
            SET TRIP.NO_AVAILABLE_PLACES =
TRIP.NO_AVAILABLE_PLACES-1
            WHERE TRIP.TRIP_ID = :NEW.TRIP_ID;
        ELSE
            NULL;
        END IF;
    IF :NEW.STATUS IN ('C', 'N') AND :OLD.STATUS = 'P' THEN
        UPDATE TRIP
            SET TRIP.NO_AVAILABLE_PLACES =
TRIP.NO_AVAILABLE_PLACES+1
            WHERE TRIP.TRIP_ID = :NEW.TRIP_ID;
        END IF;
    ELSE
        RAISE_APPLICATION_ERROR(-20001, 'STATUS CHANGE NOT
ALLOWED');
    END IF;
    COMMIT;
END;

```

Trigger sprawdza poprawność zmiany statusu i jeśli nowy status to P, zmniejsza liczbę obecnie dostępnych miejsc w Trip, a jeśli zmiana jest z P na inny to zwiększa liczbę

### 10.1.3 Zmiana liczby miejsc podczas dodania rezerwacji

W związku z przyjętą decyzją, iż jako miejsce zajęte liczymy tylko te, które zostały opłacone tzn. mają status P, nie potrzebny jest taki Trigger. Wszystkie rezerwacje, które początkowo dodajemy uznawane są jako New a one nie wpływają na liczbę miejsc.

## 10.2 Zmodyfikowane procedury

### 10.2.1 ADDRESERVATION\_5

```

CREATE OR REPLACE PROCEDURE ADDRESERVATION_5(
    TRIP_ID_ADD IN INT, PERSON_ID_ADD IN INT
) AS
    NEW_TRIP_DATE      DATE;
    NEW_RESERVATION_ID INT;
    NO_AVAILABLE       INT;
BEGIN

```

```

SELECT TRIP_DATE
INTO NEW_TRIP_DATE
FROM TRIP T
WHERE T.TRIP_ID = TRIP_ID_ADD;

SELECT NO_AVAILABLE_PLACES
INTO NO_AVAILABLE
FROM TRIP T
WHERE T.TRIP_ID = TRIP_ID_ADD;

IF NEW_TRIP_DATE > TRUNC(SYSDATE) AND NO_AVAILABLE > 0
THEN
    INSERT INTO RESERVATION(TRIP_ID, PERSON_ID, STATUS)
VALUES (TRIP_ID_ADD, PERSON_ID_ADD, 'N');
    SELECT MAX(RESERVATION_ID)
    INTO NEW_RESERVATION_ID
    FROM RESERVATION;
    CALCULATEAVAILABLEPLACES(TRIP_ID_ADD);
ELSE
    RAISE_APPLICATION_ERROR(-20001, 'OUT OF DATE');
END IF;
END;

```

Procedura nie ulega zmianie w porównaniu do poprzedniej.

### 10.2.2 MODIFYRESERVATIONSTATUS\_5

```

CREATE OR REPLACE PROCEDURE MODIFYRESERVATIONSTATUS_5(
    RESERVATION_ID_MOD IN INT, STATUS_CHANGE IN CHAR
) AS
    TRIP_ID_AC          INT;
    CURRENT_STATUS      CHAR;
BEGIN
    SELECT TRIP_ID
    INTO TRIP_ID_AC
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;

    SELECT STATUS
    INTO CURRENT_STATUS
    FROM RESERVATION
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;

    UPDATE RESERVATION
    SET STATUS = STATUS_CHANGE
    WHERE RESERVATION_ID = RESERVATION_ID_MOD;
END;

```

Usunięte jest zliczanie miejsc, gdyż za to odpowiada Trigger



### 10.2.3 MODIFYNOPLACES\_5

```
CREATE OR REPLACE PROCEDURE MODIFYNOPLACES_5
  (TRIP_ID_MOD IN INT, NEW_NUMBER_OF_PLACES IN INT) AS
  NO_RESERVED INT;
BEGIN
  SELECT T2.MAX_NO_PLACES - VT.NO_AVAILABLE_PLACES
  INTO NO_RESERVED
  FROM TRIPSVIEW_4 VT
        INNER JOIN TRIP T2 ON VT.TRIP_NAME = T2.TRIP_NAME
  WHERE T2.TRIP_ID = TRIP_ID_MOD;

  IF NEW_NUMBER_OF_PLACES >= NO_RESERVED THEN
    UPDATE TRIP
      SET MAX_NO_PLACES = NEW_NUMBER_OF_PLACES
      WHERE TRIP_ID = TRIP_ID_MOD;
  END IF;
END;
```

Usunięte jest zliczanie, gdyż za to odpowiada Trigger.