Adam Biśta, 28.03.2023r.

Spis treści

Rozdział 1 Struktura tabel

Dodane zostały 4 tabele z zadania: **trip**, **log**, **reservation**, **person** oraz dodatkowa tabela **countries.**

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
);


alter table trip
add constraint trip_fk1 foreign key
```

```
( country_id ) references countries ( country_id ) enable;
```

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
);

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;
```

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
);
```

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
);


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;
```

Countries:

```
create table countries
(
    country_id int generated always as identity not null,
    country_name varchar(50),
    constraint country_pk primary key ( country_id ) enable
);
```

Rozdział 2 Uzupełnienie tabel o przykładowe dane

Poniżej przedstawię screeny z DataGripa przedstawiające przykładowe dane zawarte w przedstawionych wcześniej tabelach:

Log:

	LOG_ID	RESERVATION_ID	LOG_DATE	STATUS
1	1	1	2022-02-01	N
2	2	1	2022-02-02	P
3	3	2	2022-02-01	N
4	4	2	2022-02-03	P
5	5	3	2022-02-02	N
6	6	3	2022-02-05	C
7	7	4	2022-02-03	N
8	8	5	2022-02-04	N
9	9	5	2022-02-07	P
10	10	6	2022-02-05	N
11	11	6	2022-02-06	P
12	12	7	2022-02-06	N
13	13	7	2022-02-09	P
14	14	8	2022-02-07	N
15	15	8	2022-02-11	C
16	16	9	2022-02-08	N
17	17	9	2022-02-10	C
18	18	10	2022-02-09	N

Person:

	PERSON_ID	FIRSTNAME	LASTNAME
1	1	Jan	Kowalski
2	2	Maria	Nowak
3	3	Andrzej	Lewandowski
4	4	Katarzyna	Wójcik
5	5	Piotr	Mazurek
6	6	Agnieszka	Szymańska
7	7	Tomasz	Jankowski
8	8	Barbara	Kamińska
9	9	Wojciech	Kaczmarek
10	10	Magdalena	Zając

Reservation:

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

Trip:

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES
1	1	Wakacje w Polsce	1	2022-07-01	20
2	2	Wakacje we Francji	2	2023-08-15	15
3	3	Wycieczka po Włoszech	3	2023-06-01	30
4	4	Wakacje w Hiszpanii	4	2023-07-15	25

Countries:

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

Rozdział 3 Widoki

Reservations

Widok w DataGripie:

	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	FIRSTNAME	LASTNAME	RESERVATION_ID	STATUS
1	Polska	2022-07-01	Wakacje w Polsce	Jan	Kowalski	1	P
2	Polska	2022-07-01	Wakacje w Polsce	Maria	Nowak	2	P
3	Francja	2023-08-15	Wakacje we Francji	Maria	Nowak	5	P
4	Polska	2022-07-01	Wakacje w Polsce	Andrzej	Lewandowski	3	C
5	Francja	2023-08-15	Wakacje we Francji	Andrzej	Lewandowski	6	P
6	Włochy	2023-06-01	Wycieczka po Włoszech	Katarzyna	Wójcik	9	C
7	Włochy	2023-06-01	Wycieczka po Włoszech	Agnieszka	Szymańska	8	C
8	Francja	2023-08-15	Wakacje we Francji	Tomasz	Jankowski	4	N
9	Włochy	2023-06-01	Wycieczka po Włoszech	Wojciech	Kaczmarek	7	P
10	Hiszpania	2023-07-15	Wakacje w Hiszpanii	Wojciech	Kaczmarek	10	N

Kod:

```
CREATE VIEW Reservations AS
```

```sql
SELECT countries.country_name,
       trip.trip_date,
       trip.trip_name,
       person.firstname,
       person.lastname,
       reservation.reservation_id,
       reservation.status
FROM reservation
INNER JOIN trip ON reservation.trip_id = trip.trip_id
INNER JOIN person ON reservation.person_id = person.person_id
INNER JOIN countries ON trip.country_id = countries.country_id;
```

Reservations to widok wyświetlający rezerwacje, łączy tabele reservation, trip, person oraz countries. Wyświetla nazwy kraju, datę wycieczki, nazwę wycieczki, imię i nazwisko osoby, identyfikator rezerwacji oraz status.

Trips

Widok w DataGripie:

	COUNTRY	TRIP_DATE	TRIP_NAME	TRIP_ID	NO_PLACES	NO_AVAILABLE_PLACES
1	Polska	2022-07-01	Wakacje w Polsce	1	20	18
2	Francja	2023-08-15	Wakacje we Francji	2	15	12
3	Włochy	2023-06-01	Wycieczka po Włoszech	3	30	29
4	Hiszpania	2023-07-15	Wakacje w Hiszpanii	4	25	24

Kod:

```sql
CREATE VIEW Trips AS
SELECT
    countries.country_name AS country,
    trip.trip_date,
    trip.trip_name,
    trip.trip_id,
    trip.max_no_places AS no_places,
    trip.max_no_places - COUNT(reservation.reservation_id) AS
no_available_places
FROM
    trip
    INNER JOIN countries ON trip.country_id = countries.country_id
    LEFT JOIN reservation ON trip.trip_id = reservation.trip_id AND
reservation.status in ('P','N')
```

```
--LEFT JOIN sprawia, ze jesli nie ma powiazania wycieczki z rezerwacjami, tzn
ze wszystkie wycieczki są dostepne

GROUP BY
    countries.country_name,
    trip.trip_date,
    trip.trip_name,
    trip.trip_id,
    trip.max_no_places;
```

Trips to widok wyświetlający wycieczki, łączy tabele trip, countries i reservation. Wyświetla nazwy kraju, datę wycieczki, nazwę wycieczki, identyfikator wycieczki, maksymalną liczbę miejsc oraz liczbę dostępnych miejsc.

Available_Trips

Widok w DataGripie:

	COUNTRY	TRIP_DATE	TRIP_NAME	TRIP_ID	NO_PLACES	NO_AVAILABLE_PLACES
1	Francja	2023-08-15	Wakacje we Francji	2	15	12
2	Włochy	2023-06-01	Wycieczka po Włoszech	3	30	29
3	Hiszpania	2023-07-15	Wakacje w Hiszpanii	4	25	24

Kod:

```
CREATE VIEW Available_Trips AS
SELECT * FROM TRIPS t
WHERE t.no_available_places > 0 AND t.trip_date > SYSDATE;
```

Available_Trips to widok wyświetlający dostępne wycieczki z wykorzystaniem widoku Trips. Wyświetla dane z wycieczek, które mają więcej niż 0 dostępnych miejsc i ich data jest późniejsza niż dzisiejsza.

Rozdział 4 Dodatkowe widoki:

Upcoming_Trips

Widok w DataGripie:

	COUNTRY	TRIP_DATE	TRIP_NAME	TRIP_ID	NO_PLACES	NO_AVAILABLE_PLACES
1	Francja	2023-08-15	Wakacje we Francji	2	15	12
2	Włochy	2023-06-01	Wycieczka po Włoszech	3	30	29
3	Hiszpania	2023-07-15	Wakacje w Hiszpanii	4	25	24

Kod:

```
CREATE VIEW Upcoming_Trips AS
SELECT * FROM TRIPS t
WHERE t.trip_date > SYSDATE;
```

Upcoming_Trips to widok który różni się od Available_trips tylko tym, że pokazuje wycieczki które dopiero się odbędą bez względu na ich dostępność.

Reservation_Log

	LOG_ID	LOG_DATE	TRIP_NAME	COUNTRY_NAME	FIRSTNAME	LASTNAME	INNER
1	1	2022-02-01	Wakacje w Polsce	Polska	Jan	Kowalski	N
2	2	2022-02-02	Wakacje w Polsce	Polska	Jan	Kowalski	P
3	3	2022-02-01	Wakacje w Polsce	Polska	Maria	Nowak	N
4	4	2022-02-03	Wakacje w Polsce	Polska	Maria	Nowak	P
5	5	2022-02-02	Wakacje w Polsce	Polska	Andrzej	Lewandowski	N
6	6	2022-02-05	Wakacje w Polsce	Polska	Andrzej	Lewandowski	C
7	7	2022-02-03	Wakacje we Francji	Francja	Tomasz	Jankowski	N
8	8	2022-02-04	Wakacje we Francji	Francja	Maria	Nowak	N
9	9	2022-02-07	Wakacje we Francji	Francja	Maria	Nowak	P
10	10	2022-02-05	Wakacje we Francji	Francja	Andrzej	Lewandowski	N
11	11	2022-02-06	Wakacje we Francji	Francja	Andrzej	Lewandowski	P
12	12	2022-02-06	Wycieczka po Włoszech	Włochy	Wojciech	Kaczmarek	N
13	13	2022-02-09	Wycieczka po Włoszech	Włochy	Wojciech	Kaczmarek	P
14	14	2022-02-07	Wycieczka po Włoszech	Włochy	Agnieszka	Szymańska	N
15	15	2022-02-11	Wycieczka po Włoszech	Włochy	Agnieszka	Szymańska	C
16	16	2022-02-08	Wycieczka po Włoszech	Włochy	Katarzyna	Wójcik	N
17	17	2022-02-10	Wycieczka po Włoszech	Włochy	Katarzyna	Wójcik	C
18	18	2022-02-09	Wakacje w Hiszpanii	Hiszpania	Wojciech	Kaczmarek	N

```
CREATE VIEW Reservation_Log AS
SELECT l.log_id,l.log_date,trip.trip_name, countries.country_name,
       person.firstname, person.lastname, l.status
INNER FROM log l
INNER JOIN reservation ON l.reservation_id = reservation.reservation_id
INNER JOIN trip ON reservation.trip_id = trip.trip_id
INNER JOIN countries ON trip.country_id = countries.country_id
INNER JOIN person ON reservation.person_id = person.person_id
```

Reservation_Log to widok wyświetlający historię zmian statusu rezerwacji. Łączy tabele log, reservation, trip, countries oraz person.

Wyświetla identyfikator logu, datę logu, nazwę wycieczki, nazwę kraju, imię i nazwisko osoby oraz status.

Country_Statistics

	COUNTRY_ID	COUNTRY_NAME	NO_TRIPS	NO_TOTAL_PLACES	NO_AVAILABLE_PLACES
1	1	Polska	1	20	0
2	2	Francja	1	15	15
3	3	Włochy	1	30	30
4	4	Hiszpania	1	25	25

```
CREATE VIEW Country_Statistics AS
SELECT countries.COUNTRY_ID,countries.country_name, COUNT(trip.trip_id) AS
no_trips,
       SUM(trip.max_no_places) AS no_total_places,
       SUM(CASE WHEN trip.trip_date > SYSDATE THEN trip.max_no_places ELSE 0
END) AS no_available_places
FROM countries
INNER JOIN trip ON countries.country_id = trip.country_id
GROUP BY countries.COUNTRY_ID,countries.country_name;
```

Country_Statistics to widok wyświetlający statystyki dla krajów, łączy tabele countries i trip. Wyświetla identyfikator kraju, nazwę kraju, liczbę wycieczek, sumaryczną liczbę miejsc oraz sumaryczną liczbę dostępnych miejsc.

Most_Popular_Countries_Last_Year

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

```
CREATE VIEW Most_Popular_Countries_Last_Year
AS
```

```sql
SELECT countries.country_id,countries.country_name, COUNT(trip.trip_id) AS
trip_count
FROM countries
INNER JOIN trip ON countries.country_id = trip.country_id
INNER JOIN reservation ON trip.trip_id = reservation.trip_id
WHERE reservation.status in ('P','N') AND trip.trip_date >= SYSDATE - INTERVAL
'1' YEAR
GROUP BY countries.country_id,countries.country_name
ORDER BY trip_count DESC
FETCH FIRST 5 ROWS ONLY;
```

Most_Popular_Countries_Last_Year to widok wyświetlający 5 najbardziej popularnych krajów w ostatnim roku, łączy tabele countries, trip oraz reservation. Wyświetla identyfikator kraju, nazwę kraju oraz liczbę wycieczek.

NO_Pait_Reservations_Per_Person

	LASTNAME	FIRSTNAME	NO_PAID_RESERVATIONS
1	Nowak	Maria	2
2	Zając	Magdalena	0
3	Mazurek	Piotr	0
4	Wójcik	Katarzyna	0
5	Kowalski	Jan	1
6	Jankowski	Tomasz	0
7	Kaczmarek	Wojciech	1
8	Kamińska	Barbara	0
9	Szymańska	Agnieszka	0
10	Lewandowski	Andrzej	1

```sql
CREATE VIEW NO_Paid_Reservations_Per_Person AS
SELECT person.lastname , person.firstname,COUNT(reservation.RESERVATION_ID) as
NO_paid_reservations
FROM person
LEFT JOIN reservation ON person.person_id = reservation.person_id  and
reservation.status = 'P'
GROUP BY person.lastname , person.firstname
```

NO_Paid_Reservations_Per_Person to widok wyświetlający liczbę opłaconych rezerwacji na osobę, łączy tabele person oraz reservation. Wyświetla imię i nazwisko osoby oraz liczbę opłaconych rezerwacji.

Cancelled_Reservations

	TRIP_ID	TRIP_NAME	NO_CANCELLED_RESERVATIONS
1	1	Wakacje w Polsce	1
2	2	Wakacje we Francji	0
3	3	Wycieczka po Włoszech	2
4	4	Wakacje w Hiszpanii	0

```
CREATE VIEW CANCELLED_RESERVATIONS AS
SELECT trip.trip_id,trip.TRIP_NAME , COUNT(reservation.RESERVATION_ID) as
NO_cancelled_reservations
FROM trip
LEFT JOIN reservation ON trip.trip_id =  reservation.trip_id and
reservation.status = 'C'
GROUP BY trip.trip_id,trip.TRIP_NAME
```

CANCELLED_RESERVATIONS to widok wyświetlający liczbę odwołanych rezerwacji na wycieczkę, łączy tabele trip oraz reservation. Wyświetla identyfikator wycieczki, nazwę wycieczki oraz liczbę odwołanych rezerwacji.

Rozdział 5 Tworzenie procedur/funkcji pobierających dane

Funkcja Trip_Participants_Func:

```
create or replace type trip_participant as OBJECT
(
reservation_id int,
person_id int,
firstname varchar2(50),
Lastname varchar2(50)
);
create or replace type trip_participant_table is table of trip_participant;

create or replace function Trip_Participants_Func(trip_id int)
return trip_participant_table
```

```
as
result trip_participant_table;
tmp char(1);
no_data exception;
begin

if Trip_Participants_Func.trip_id is null then
raise_application_error(-20002,'przeslana wartosc to null');
end if;
    -- sprawdzenie, czy istnieje wycieczka o podanym trip_id
select 1 into tmp from trip where trip.TRIP_ID =
Trip_Participants_Func.trip_id;

select trip_participant(r.reservation_id, p.person_id,
p.firstname, p.lastname) bulk collect
into result
from reservation r
join person p
on r.person_id = p.person_id
where r.trip_id = Trip_Participants_Func.trip_id and r.status = 'P';
return result;
EXCEPTION
WHEN NO_DATA_FOUND THEN
    raise_application_error(-20001,'Nie znaleziono osoby o id: ' ||
Trip_Participants_Func.trip_id);
    RETURN NULL;
WHEN others THEN
    raise_application_error(-20003,'NIEPRAWIDLOWE DZIALANIE');
    RETURN NULL;
END;
```

Funkcja ta zwraca tablicę z danymi uczestników wycieczki o zadanym przez nas id.

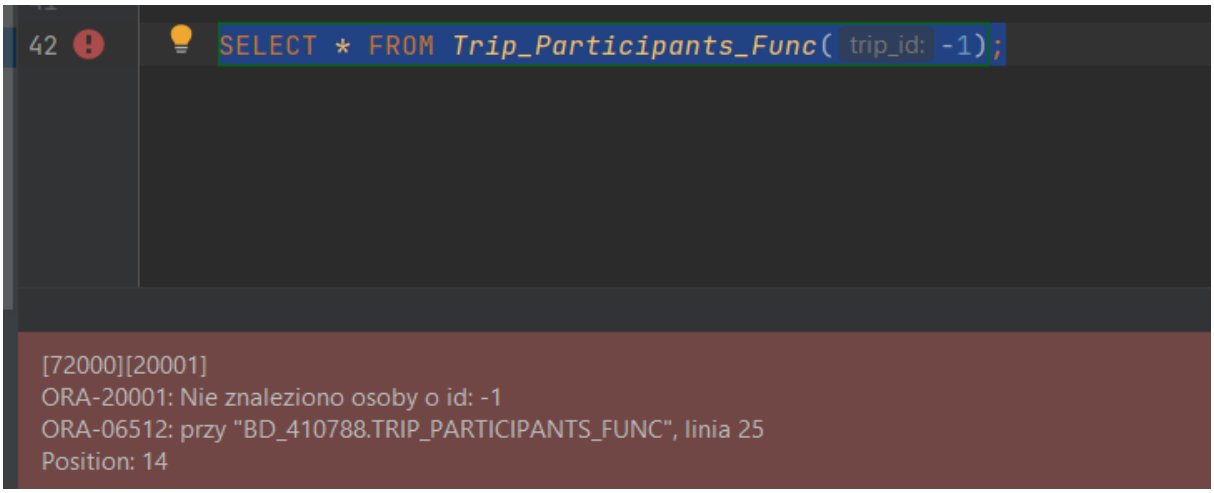
Przykładowe wywołanie:

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

	RESERVATION_ID	PERSON_ID	FIRSTNAME	LASTNAME
1	1	1	Jan	Kowalski
2	2	2	Maria	Nowak

Wywołanie w którym podanym id które nie istnieje w bazie:

```
SELECT * FROM Trip_Participants_Func(-1);
```



Funkcja Person_Reservations_Func:

```
create or replace type person_reservations as OBJECT
(
reservation_id int,
reservation_status char(1),
trip_date DATE,
trip_name varchar2(50),
country_name varchar2(50)
);
create or replace type person_reservations_table is table of
person_reservations;

create or replace function Person_Reservations_Func(person_id int)
return person_reservations_table
as
result person_reservations_table;
tmp char(1);
null_arg exception;
begin
```

```
if Person_Reservations_Func.person_id is null then
raise_application_error(-20002,'argument jest nullem!');
end if;
-- sprawdzenie, czy istnieje osoba o podanym person_id
select 1 into tmp from person where person.person_id =
Person_Reservations_Func.person_id;

select person_reservations(r.reservation_id, r.status,
t.TRIP_DATE, t.TRIP_NAME,c.COUNTRY_NAME) bulk collect
into result
from reservation r
join trip t on t.trip_id = r.TRIP_ID
join countries c on c.COUNTRY_ID = t.COUNTRY_ID
where r.PERSON_ID = Person_Reservations_Func.PERSON_ID and r.status != 'C';
return result;
EXCEPTION
WHEN NO_DATA_FOUND THEN
raise_application_error(-20001,'Nie znaleziono zadnej rezerwacji dla osoby
o id: ' || Person_Reservations_Func.PERSON_ID);
RETURN NULL;
WHEN others THEN
raise_application_error(-20003,'Wystapil blad!');
RETURN NULL;
END;
```

Ta funkcja przyjmuje jako argument ID osoby i zwraca wszystkie jej aktywne rezerwacje na wycieczki w formie tabeli typu person_reservations_table.

Przykładowe wywołanie:

```
SELECT * FROM Person_Reservations_Func(2);
```

	RESERVATION_ID	RESERVATION_STATUS	TRIP_DATE	TRIP_NAME	COUNTRY_NAME
1	2	P	2022-07-01	Wakacje w Polsce	Polska
2	5	P	2023-08-15	Wakacje we Francji	Francja

Wywołanie w którym podanym id które nie istnieje w bazie:

```
SELECT * FROM Person_Reservations_Func(-1);
```

```
[72000][20001]
ORA-20001: Nie znaleziono zadnej rezerwacji dla osoby o id: -1
ORA-06512: przy "BD_410788.PERSON_RESERVATIONS_FUNC", linia 25
Position: 14
```

Funkcja Trip_Participants_Func:

```
create or replace type available_trips_obj as OBJECT
(
trip_date DATE,
trip_name varchar(100),
NO_places int,
NO_available_places int
);
```

```
create or replace type available_trips_table is table of
available_trips_obj;

create or replace function available_Trips_Func(country_name varchar,
date_from DATE, date_to DATE)
return available_trips_table
AS
result available_trips_table;
null_arg exception;
wrong_order exception ;
tmp char(1);
begin
IF country_name is null or date_from is null or date_to is null then
raise_application_error(-20003,'przeslana wartosc to null');
END IF;

IF date_to < date_from THEN
raise_application_error(-20002,'from_data wieksze od to_data');
END IF;
-- sprawdzenie, czy istnieje wycieczka o podanym country_name i czasie
select 1 into tmp from Available_Trips avt
where avt.country = available_Trips_Func.country_name
AND avt.trip_date BETWEEN available_Trips_Func.date_from AND
available_Trips_Func.date_to;

--zapytanie wsadzone do result
select available_trips_obj(avt.trip_date,
avt.trip_name, avt.no_places, avt.no_available_places)
bulk collect into result
from Available_Trips avt
where avt.country = available_Trips_Func.country_name
AND avt.trip_date BETWEEN available_Trips_Func.date_from AND
available_Trips_Func.date_to;
--zwracanie wyniku
return result;
EXCEPTION
WHEN NO_DATA_FOUND THEN
raise_application_error(-20001,'wycieczka z podanym country_name oraz datą
nie istnieje lub się odbyła');
RETURN NULL;
WHEN others THEN
raise_application_error(-20004,'Wystapil blad!');
RETURN NULL;
END;
```

Ta funkcja przyjmuje trzy argumenty: country_name, date_from i date_to. Zwraca tablicę, która reprezentuje wycieczki spełniające określone kryteria, czyli wycieczki w określonym kraju i pomiędzy określonymi datami, które mają wolne miejsca.

Przykładowe wywołanie

```
SELECT * FROM available_Trips_Func('Francja','2021-01-01', '2023-12-31');
```

	TRIP_DATE	TRIP_NAME	NO_PLACES	NO_AVAILABLE_PLACES
1	2023-08-15	Wakacje we Francji	15	12

Wywołanie, które nie znajduje żadnych wycieczek:

```
SELECT * FROM available_Trips_Func('Polska','2021-01-01', '2023-12-31');
```

```
[72000][20001]
ORA-20001: wycieczka z podanym country_name oraz datą nie istnieje lub się odbyła
ORA-06512: przy "BD_410788.AVAILABLE_TRIPS_FUNC", linia 32
Position: 14
```

Jeśli uwzględnimy fakt, że funkcja może zwrócić pustą tabelę, to należy usunąć W Exception wyjątek NO_DATA_FOUND. Zdecydowałem się jednak na zakomunikowanie, że nie istnieją wycieczki.

Rozdział 6 Tworzenie procedur modyfikujących dane oraz dodawanie wpisów do dzienniczka

Postanowiłem od razu połączyć zadanie 5. z 6., gdyż w 6. w procedurach jedynie dodaję kolejną właściwość jaką jest zapis do tabeli log zmian statusów rezerwacji wycieczek.

Add_Reservation:

```
create or replace PROCEDURE Add_Reservation(trip_id int, person_id int)
IS
tmp_person char(1);
tmp_trip char(1);
not_exist inT;
id int;
max_places_was_already int;
begin
    --sprawdzenie wystapienia null
    if trip_id is null or person_id is null then
        raise_application_error(-20002, 'Przesłana wartość jest równa
NULL');
    end if;
    --sprawdzenie czy osoba o podanym id istnieje
    select 1 into tmp_person FROM PERSON p WHERE Add_Reservation.person_id
= p.person_id;
```

```
    --sprawdzenie czy wycieczka o podanym id istnieje
    select 1 into tmp_trip FROM TRIP t WHERE Add_Reservation.trip_id =
t.trip_id;
    --sprawdzenie czy osoba o podanym id nie ma juz wybranej wycieczki o
podanym id i czy status rezerwacji != C
    select case
        when exists(select * FROM RESERVATION r
                            WHERE r.PERSON_ID = Add_Reservation.person_id
                           AND r.trip_id = Add_Reservation.trip_id) then 0
        else 1
        end
    into not_exist from dual;

    if not_exist = 0 then
        raise_application_error(-20003, 'Osoba z podaną rezerwacją już
istnieje, Możesz zmodyfikować tą rezerwację');
    end if;
    --sprawdzamy czy wycieczka sie juz nie odbyla
   --sprawdzamy czy są wolne miejsca
    select case
            when exists(select * FROM AVAILABLE_TRIPS t
                                WHERE t.trip_id = Add_Reservation.trip_id
) then 1
            else 0
            end
        into max_places_was_already from dual;
        if max_places_was_already = 0 then
            raise_application_error(-20004, 'Nie można dodać rezerwacji dla
wycieczki, która już się odbyła lub nie ma wolnych miejsc');
        end if;

   --dodajemy osobę
    INSERT INTO RESERVATION(trip_id,person_id,status)
VALUES(Add_Reservation.trip_id,Add_Reservation.person_id,'N')
    returning reservation_id into id;
    dbms_output.PUT_LINE(id);

    INSERT INTO LOG(RESERVATION_ID,LOG_DATE,status)
VALUES(id,TO_CHAR(SYSDATE, 'YYYY-MM-DD'),'N');
    COMMIT;

EXCEPTION
   WHEN NO_DATA_FOUND THEN
        raise_application_error(-20001, 'Nie znaleziono wycieczki lub osoby
z podanym ID');
END;
/
```

Procedura ta dodaje rezerwację przyjmując parametry trip_id oraz person_id. Sprawdza ona czy istnieje podane trip_id, podane person_id oraz czy osoba już nie dokonała pewnej rezerwacji. Po dodaniu rezerwacji następuje zapis do tabeli log aktualnego statusu tej rezerwacji.

Przykładowe wywołanie:

```
DECLARE
v_person_id INT := 8;
v_trip_id INT := 2;
begin
    Add_Reservation(v_trip_id,v_person_id);

end;
```

	RESERVATION_ID	TRIP_ID	PERSON_ID	STATUS
1	1	1	1	P
2	2	1	2	P
3	3	1	3	C
4	4	2	7	N
5	5	2	2	P
6	6	2	3	P
7	7	3	9	P
8	8	3	6	C
9	9	3	4	C
10	10	4	9	N
11	21	2	8	N

W tabel z logami na dole również się pojawia nowy element:

	LOG_ID	RESERVATION_ID	LOG_DATE	STATUS
6	6	3	2022-02-05	C
7	7	4	2022-02-03	N
8	8	5	2022-02-04	N
9	9	5	2022-02-07	P
10	10	6	2022-02-05	N
11	11	6	2022-02-06	P
12	12	7	2022-02-06	N
13	13	7	2022-02-09	P
14	14	8	2022-02-07	N
15	15	8	2022-02-11	C
16	16	9	2022-02-08	N
17	17	9	2022-02-10	C
18	18	10	2022-02-09	N
19	21	21	2023-03-29	N

Wywołanie w którym podamy niepoprawne id wycieczki:

```
DECLARE
v_person_id INT := 2;
v_trip_id INT := 8;
begin
    Add_Reservation(v_trip_id,v_person_id);

end;
```

```
[72000][20001]
ORA-20001: Nie znaleziono wycieczki lub osoby z podanym ID
ORA-06512: przy "BD_410788.ADD_RESERVATION", linia 51
ORA-06512: przy linia 5
Position: 0
```

Wywołanie, w którym osoba już posiada rezerwację z statusem w bazie danych:

```
DECLARE
v_person_id INT := 8;
v_trip_id INT := 2;
begin
    Add_Reservation(v_trip_id,v_person_id);

end;
```

```
[72000][20003]
ORA-20003: Osoba z podaną rezerwacją już istnieje, Możesz zmodyfikować tą rezerwację
ORA-06512: przy "BD_410788.ADD_RESERVATION", linia 27
ORA-06512: przy linia 5
Position: 0
```

Modify_NO_Places

```
create or replace PROCEDURE Modify_No_Places(trip_id INT, no_places INT) AS
tmp_reserv0 INT;
reserved_places INT;
BEGIN
  if trip_id is null or trip_id < 0 or no_places is null or no_places < 0
then
        raise_application_error(-20001, 'Nie mozna podawać wartosci
nullowej lub mniejszej od 0');
 END IF;
    -- Sprawdzenie, czy wycieczka istnieje
  SELECT 1 INTO tmp_reserv0 FROM trip t WHERE t.trip_id  =
Modify_No_Places.trip_id;
  -- wydobycie ilosci dostepnych miejsc
  SELECT uptr.NO_PLACES - uptr.NO_AVAILABLE_PLACES INTO reserved_places
FROM UPCOMING_TRIPS uptr WHERE uptr.TRIP_ID = Modify_No_Places.trip_id;


  IF (Modify_No_Places.no_places < reserved_places) THEN
    RAISE_APPLICATION_ERROR(-20002, 'Ilosc zarezerwowanych miejsc na daną
wycieczkę musi byc mniejsza od nowej ilosc miejsc');
  END IF;

  -- Aktualizacja liczby miejsc dla wycieczki
  UPDATE TRIP t SET t.MAX_NO_PLACES = Modify_No_Places.no_places WHERE
t.trip_id = Modify_No_Places.trip_id;

  COMMIT;

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Nie znaleziono wycieczki o zadanym
id,mozliwe ze wycieczka juz sie odbyla albo niepoprawne id');
END;
/
```

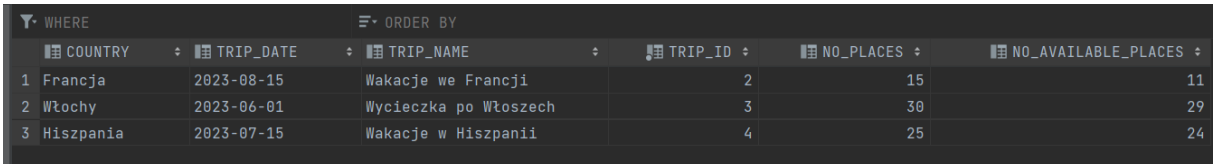
Procedura ta zmienia całkowitą ilość miejsc danej wycieczki. Przyjmuje dwa parametry: identyfikator wycieczki oraz nową ilość miejsc na tą wycieczkę. Zapobiega sytuacji, w której ilość dostępnych miejsc byłaby mniejsza od zera oraz nie możliwe jest ustalenie ilości miejsc dla wycieczek, które się już odbyły. Dlatego wykorzystałem dodatkowy widok Upcoming_Trips.

Przykładowe wywołanie:

```
DECLARE
   v_trip_id INT := 3;
   no_places INT := 20;
BEGIN
   Modify_No_Places(v_trip_id , no_places );
END;
/
```

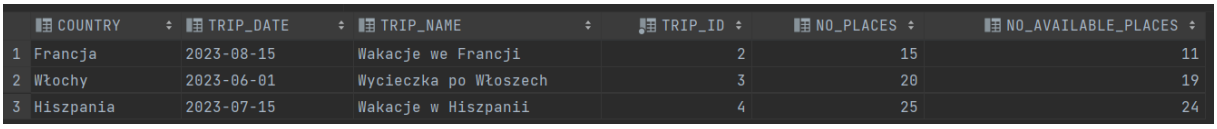
Dla ułatwienia posłużę się widokiem Upcoming_Trips by zasygnalizować zmiany:

Przed wywołaniem:



	COUNTRY	TRIP_DATE	TRIP_NAME	TRIP_ID	NO_PLACES	NO_AVAILABLE_PLACES
1	Francja	2023-08-15	Wakacje we Francji	2	15	11
2	Włochy	2023-06-01	Wycieczka po Włoszech	3	30	29
3	Hiszpania	2023-07-15	Wakacje w Hiszpanii	4	25	24

Po wywołaniu:



	COUNTRY	TRIP_DATE	TRIP_NAME	TRIP_ID	NO_PLACES	NO_AVAILABLE_PLACES
1	Francja	2023-08-15	Wakacje we Francji	2	15	11
2	Włochy	2023-06-01	Wycieczka po Włoszech	3	20	19
3	Hiszpania	2023-07-15	Wakacje w Hiszpanii	4	25	24

Niepoprawne wywołanie: spróbujmy zmienić max_NO_places wycieczki o id 2, z 15 na 3:

```
DECLARE
   v_trip_id INT := 2;
   no_places INT := 3;
BEGIN
   Modify_No_Places(v_trip_id , no_places );
END;
/
```

```
[72000][20002]
ORA-20002: Ilosc zarezerwowanych miejsc na daną wycieczkę musi byc mniejsza od nowej ilosc miejsc
ORA-06512: przy "BD_410788.MODIFY_NO_PLACES", linia 15
ORA-06512: przy linia 5
Position: 0
```

Modify_Reservation_Status:

```
create or replace PROCEDURE
Modify_Reservation_Status(modifying_reservation_id int, new_status char)
IS
   modif int;
    old_reserv_status char(1);
   modifying_trip_id int;
BEGIN
   SELECT r.status, r.trip_id
    INTO old_reserv_status, modifying_trip_id
    FROM RESERVATION r
    WHERE r.reservation_id = modifying_reservation_id;

   IF old_reserv_status IS NULL THEN
      RAISE_APPLICATION_ERROR(-20001, 'Error: trip not found.');
```

```
   END IF;

   IF old_reserv_status = new_status THEN
      RAISE_APPLICATION_ERROR(-20001, 'Error: You gave same status as
before.');
   END IF;

IF old_reserv_status = 'C' THEN
        SELECT CASE
                    WHEN EXISTS(SELECT *
                                FROM AVAILABLE_TRIPS avt
                                WHERE avt.trip_id = modifying_trip_id) THEN
0
                    ELSE 1
                    END
        INTO modif
        FROM DUAL;
        IF modif = 1 THEN
            RAISE_APPLICATION_ERROR(-20001, 'Error: the trip has already
taken place or there are no vacancies.');
        END IF;
    END IF;

   UPDATE RESERVATION SET status = new_status WHERE reservation_id =
modifying_reservation_id;
  -- dodanie nowego wpisu do LOG
INSERT INTO LOG(RESERVATION_ID, LOG_DATE, status)
VALUES(modifying_reservation_id, TO_CHAR(SYSDATE, 'YYYY-MM-
DD'),new_status);
  COMMIT;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Rezerwacja o podanym ID nie istnieje
lub niepoprawny status');
end;
/
```

Procedura ta pobiera od użytkownika id rezerwacji oraz jej nowy status. Uwzględnia ona przypadki skrajne, w których nie można już rezerwacji zmienić lub podane dane są niepoprawne dane.

Poprawne wywołanie:

```
DECLARE
   v_reservation_id INT := 8;
   status char(1) := 'P';
BEGIN
   Modify_Reservation_Status(v_reservation_id, status);
END;
/
```

Przed:

	RESERVATION_ID	TRIP_ID	PERSON_ID	STATUS
1	1	1	1	P
2	2	1	2	P
3	3	1	3	C
4	4	2	7	N
5	5	2	2	P
6	6	2	3	P
7	7	3	9	P
8	8	3	6	C
9	9	3	4	C
10	10	4	9	N
11	21	2	8	N

Po:

	RESERVATION_ID	TRIP_ID	PERSON_ID	STATUS
1	1	1	1	P
2	2	1	2	P
3	3	1	3	C
4	4	2	7	N
5	5	2	2	P
6	6	2	3	P
7	7	3	9	P
8	8	3	6	P
9	9	3	4	C
10	10	4	9	N
11	21	2	8	N

Tabela log:

	LOG_ID	RESERVATION_ID	LOG_DATE	STATUS
8	8	5	2022-02-04	N
9	9	5	2022-02-07	P
10	10	6	2022-02-05	N
11	11	6	2022-02-06	P
12	12	7	2022-02-06	N
13	13	7	2022-02-09	P
14	14	8	2022-02-07	N
15	15	8	2022-02-11	C
16	16	9	2022-02-08	N
17	17	9	2022-02-10	C
18	18	10	2022-02-09	N
19	21	21	2023-03-29	N
20	41	8	2023-03-29	P

Zauważmy że dla trip_id = 3 mamy 2 aktualne rezerwacje i 1 odwołaną. Zmieńmy ilość możliwych miejsc na 2 i spróbujmy zaktualizować rezerwację o id = 9 z ‘C’ na ‘N’:

```
DECLARE
   v_trip_id INT := 3;
   no_places INT := 2;
BEGIN
   Modify_No_Places(v_trip_id , no_places );
END;
/
```

Teraz zobaczmy przez widok Upcoming_trips jak się zmieniła liczba miejsc:

	COUNTRY	TRIP_DATE	TRIP_NAME	TRIP_ID	NO_PLACES	NO_AVAILABLE_PLACES
1	Francja	2023-08-15	Wakacje we Francji	2	15	11
2	Włochy	2023-06-01	Wycieczka po Włoszech	3	2	0
3	Hiszpania	2023-07-15	Wakacje w Hiszpanii	4	25	24

Zmodyfikujmy status:

```
DECLARE
   v_reservation_id INT := 9;
   status char(1) := 'P';
BEGIN
   Modify_Reservation_Status(v_reservation_id, status);
END;
/
```

```
[72000][20001]
ORA-20001: Error: the trip has already taken place or there are no vacancies.
ORA-06512: przy "BD_410788.MODIFY_RESERVATION_STATUS", linia 33
ORA-06512: przy linia 5
Position: 0
```

Rozdział 7 Zmiana strategii zapisywania do dziennika rezerwacji. Realizacja przy pomocy triggerów.

Zmodyfikujmy najpierw procedury opisane w poprzednim podpunkcie. Wystarczy, że w Add_Reservation usuniemy Insert do Log, to samo zróbmy dla Modify_Reservation_Status. Zróbmy nowe funkcje z suffixem _2:

Add_Reservation_2:

```
create or replace PROCEDURE Add_Reservation_2(trip_id int, person_id int)
IS
tmp_person char(1);
tmp_trip char(1);
not_exist inT;
id int;
max_places_was_already int;
begin
    --sprawdzenie wystapienia null
    if trip_id is null or person_id is null then
        raise_application_error(-20002, 'Przesłana wartość jest równa
NULL');
    end if;
    --sprawdzenie czy osoba o podanym id istnieje
    select 1 into tmp_person FROM PERSON p WHERE
Add_Reservation_2.person_id = p.person_id;
    --sprawdzenie czy wycieczka o podanym id istnieje
    select 1 into tmp_trip FROM TRIP t WHERE Add_Reservation_2.trip_id =
t.trip_id;
    --sprawdzenie czy osoba o podanym id nie ma juz wybranej wycieczki o
podanym id i czy status rezerwacji != C
    select case
        when exists(select * FROM RESERVATION r
                            WHERE r.PERSON_ID =
Add_Reservation_2.person_id
                           AND r.trip_id = Add_Reservation_2.trip_id) then
0
        else 1
        end
    into not_exist from dual;

    if not_exist = 0 then
        raise_application_error(-20003, 'Osoba z podaną nieodwołaną
rezerwacją już istnieje');
    end if;
    --sprawdzamy czy wycieczka sie juz nie odbyla
   --sprawdzamy czy są wolne miejsca
    select case
            when exists(select * FROM AVAILABLE_TRIPS t
```

```
                                WHERE t.trip_id =
Add_Reservation_2.trip_id ) then 1
            else 0
            end
        into max_places_was_already from dual;
        if max_places_was_already = 0 then
            raise_application_error(-20004, 'Nie można dodać rezerwacji dla
wycieczki, która już się odbyła lub nie ma wolnych miejsc');
        end if;

   --dodajemy osobę
    INSERT INTO RESERVATION(trip_id,person_id,status)
VALUES(Add_Reservation_2.trip_id,Add_Reservation_2.person_id,'N')
    returning reservation_id into id;
    dbms_output.PUT_LINE(id);
    COMMIT;

EXCEPTION
   WHEN NO_DATA_FOUND THEN
        raise_application_error(-20001, 'Nie znaleziono wycieczki lub osoby
z podanym ID');
END;
/
```

Modify_Reservation_Status_2:

```
create or replace PROCEDURE
Modify_Reservation_Status_2(modifying_reservation_id int, new_status char)
IS
   can_modify int;
    old_reserv_status char(1);
   modifying_trip_id int;
BEGIN
   if new_status NOT IN ('C','N','P') THEN
       RAISE_APPLICATION_ERROR(-20001,'Error: wrong status!');
   end if;

   SELECT r.status, r.trip_id
    INTO old_reserv_status, modifying_trip_id
    FROM RESERVATION r
    WHERE r.reservation_id = modifying_reservation_id;

   IF old_reserv_status IS NULL THEN
      RAISE_APPLICATION_ERROR(-20001, 'Error: trip not found.');
   END IF;

   IF old_reserv_status = new_status THEN
      RAISE_APPLICATION_ERROR(-20001, 'Error: You gave same status as
before.');
   END IF;

IF old_reserv_status = 'C' THEN
        SELECT CASE
                    WHEN EXISTS(SELECT *
                                FROM AVAILABLE_TRIPS avt
                                WHERE avt.trip_id = modifying_trip_id) THEN
1
                    ELSE 0
                    END
```

```
        INTO can_modify
        FROM DUAL;
        IF can_modify = 0 THEN
            RAISE_APPLICATION_ERROR(-20001, 'Error: the trip has already
taken place or there are no vacancies.');
        END IF;
    END IF;

   UPDATE RESERVATION
    SET status = new_status
    WHERE reservation_id = modifying_reservation_id;
  COMMIT;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Rezerwacja o podanym ID nie istnieje
lub niepoprawny status');
end;
/
```

Teraz zajmijmy się triggerami:

Trg_Add_reservation:

```
CREATE OR REPLACE TRIGGER trg_add_reservation
AFTER INSERT ON reservation
FOR EACH ROW
BEGIN
  INSERT INTO LOG (reservation_id, log_date, status)
  VALUES (:NEW.reservation_id, TO_CHAR(SYSDATE, 'YYYY-MM-DD'), :NEW.status);
END;
/
```

Trg_modify_reservation:

```

CREATE OR REPLACE TRIGGER trg_modify_reservation
AFTER UPDATE ON reservation
FOR EACH ROW
BEGIN
    INSERT INTO LOG (reservation_id, log_date, status)
    VALUES (:NEW.reservation_id, TO_CHAR(SYSDATE, 'YYYY-MM-DD'), :NEW.status);
END;
/
```

Trg_delete_reservation:

```
CREATE OR REPLACE TRIGGER trg_delete_reservation
BEFORE DELETE ON reservation
FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010, 'Deletion of reservations is not allowed');
END;
/
```

Zauważmy, że dwa pierwsze wykonają się po update/insert natomiast ostatni przed próbą usunięcia jakiejś rezerwacji.

Wywołania:

```
DECLARE
   trip_id INT := 3;
   person_id INT := 2;
BEGIN
   ADD_RESERVATION_2(trip_id, person_id);
END;
/
```

Reservation:

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

Log:

	LOG_ID	RESERVATION_ID	LOG_DATE	STATUS
1	1	1	2022-02-01	N
2	2	1	2022-02-02	P
3	3	2	2022-02-01	N
4	4	2	2022-02-03	P
5	5	3	2022-02-02	N
6	6	3	2022-02-05	C
7	7	4	2022-02-03	N
8	8	5	2022-02-04	N
9	9	5	2022-02-07	P
10	10	6	2022-02-05	N
11	11	6	2022-02-06	P
12	12	7	2022-02-06	N
13	13	7	2022-02-09	P
14	14	8	2022-02-07	N
15	15	8	2022-02-11	C
16	16	9	2022-02-08	N
17	17	9	2022-02-10	C
18	18	10	2022-02-09	N
19	21	21	2023-03-29	N
20	41	8	2023-03-29	P
21	61	41	2023-03-29	N

Jak widać powstała rezerwacja o id 41 i została wpisana w Log.

Wywołanie dla Modify_Reservation_Status_2:

```
DECLARE
   v_reservation_id INT := 8;
   status char(1) := 'C';
BEGIN
   Modify_Reservation_Status_2(v_reservation_id, status);
END;
/
```

Reservation:

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

Log:

8	18	10	2022-02-09	N
9	21	21	2023-03-29	N
0	41	8	2023-03-29	P
1	61	41	2023-03-29	N
2	62	8	2023-03-29	C

Podobnie jak dla dodawania rezerwacji, modyfikacja przy pomocy triggera również zadziałała.

```
DELETE FROM reservation WHERE reservation_id = 1;
```

```
[72000][20010]
ORA-20010: Deletion of reservations is not allowed
ORA-06512: przy "BD_410788.TRG_DELETE_RESERVATION", linia 2
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_410788.TRG_DELETE_RESERVATION'
Position: 12
```

Rozdział 8 Zmiana strategii kontroli dostępności miejsc. Realizacja przy pomocy triggerów

Dodałem nową funkcję pomocniczą free_places(). Funkcja ta korzysta tylko z tabel Trip oraz Reservation, zlicza ilość wolnych miejsc.

```
CREATE OR REPLACE FUNCTION free_places(trip_id IN NUMBER) RETURN NUMBER
IS
    total_places NUMBER;
    reserved_places NUMBER;
    available_places NUMBER;
BEGIN
    SELECT MAX_NO_PLACES INTO total_places FROM TRIP T WHERE t.TRIP_ID =
free_places.trip_id;
    SELECT COUNT(*) INTO reserved_places FROM RESERVATION r WHERE r.TRIP_ID
= free_places.trip_id AND STATUS IN ('N','P');
    available_places := total_places - reserved_places;
    RETURN available_places;
END;
/
```

Triggery:

Tr_check_before_add_reservation:

```
create or replace trigger TR_CHECK_BEFORE_ADD_RESERVATION
before insert
on RESERVATION
for each row
DECLARE PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
IF free_places(:NEW.trip_id) <= 0 THEN
RAISE_APPLICATION_ERROR(-20001, 'Trigger error: No available places!');
END IF;
END;
/
```

Jeśli ilość wolnych miejsc będzie <= 0 to nie zostanie dodana rezerwacja.

Tr_before_modified_reservation

```
create trigger TR_BEFORE_MODIFIED_RESERVATION
before update of STATUS
on RESERVATION
for each row
DECLARE PRAGMA AUTONOMOUS_TRANSACTION;
trip INT;
BEGIN
```

```
IF  (:NEW.STATUS = 'P' OR :NEW.STATUS = 'N') AND :OLD.STATUS = 'C'
THEN
SELECT TRIP_ID into trip FROM RESERVATION WHERE RESERVATION_ID =
:OLD.reservation_id;
IF free_places(trip) <= 0 THEN
RAISE_APPLICATION_ERROR(-20001, 'Trigger error: No available places!');
END IF;
END IF;
END;
/
```

Jeśli poprzedni status był równy ‘C’ i go zmieniamy, to trzeba sprawdzić ilosc miejsc. Jeśli jest <=0 to nie zostanie zmieniona.

Zauważmy, że triggery zdefiniowane w poprzednich zadaniach mogą zostać, będą dalej dodawać nowe wpisy do tabeli Log. W tych nowych triggerach sprawdzamy liczbę miejsc PRZED insert/update.

Nasze procedury Add_reservation i Modify_Reservation_Status możemy jeszcze pomniejszyć o sprawdzanie dostępności wycieczek.

Add_reservation_3:

```
create or replace PROCEDURE Add_Reservation_3(trip_id int, person_id int)
IS
tmp_person char(1);
tmp_trip char(1);
not_exist inT;
id int;
begin
    --sprawdzenie wystapienia null
    if trip_id is null or person_id is null then
        raise_application_error(-20002, 'Przesłana wartość jest równa
NULL');
    end if;
    --sprawdzenie czy osoba o podanym id istnieje
    select 1 into tmp_person FROM PERSON p WHERE
Add_Reservation_3.person_id = p.person_id;
    --sprawdzenie czy wycieczka o podanym id istnieje
    select 1 into tmp_trip FROM TRIP t WHERE Add_Reservation_3.trip_id =
t.trip_id;
    --sprawdzenie czy osoba o podanym id nie ma juz wybranej wycieczki o
podanym id i czy status rezerwacji != C
    select case
        when exists(select * FROM RESERVATION r
                            WHERE r.PERSON_ID =
Add_Reservation_3.person_id
                           AND r.trip_id = Add_Reservation_3.trip_id) then
0
        else 1
        end
    into not_exist from dual;

    if not_exist = 0 then
```

```
        raise_application_error(-20003, 'Osoba z podaną  rezerwacją już
istnieje. Jesli odwolala rezerwacje to sprobuj uzyc
ModifyReservationStatus()');
    end if;
   --dodajemy osobę
    INSERT INTO RESERVATION(trip_id,person_id,status)
VALUES(Add_Reservation_3.trip_id,Add_Reservation_3.person_id,'N')
    returning reservation_id into id;
    dbms_output.PUT_LINE(id);
    COMMIT;
EXCEPTION
   WHEN NO_DATA_FOUND THEN
        raise_application_error(-20001, 'Nie znaleziono wycieczki lub osoby
z podanym ID');
END;
/
```

Modify_Reservation_Status_3:

```
create or replace PROCEDURE
Modify_Reservation_Status_3(modifying_reservation_id int, new_status char)
IS
    old_reserv_status char(1);
   modifying_trip_id int;
BEGIN
   if new_status NOT IN ('C','N','P') THEN
       RAISE_APPLICATION_ERROR(-20001,'Error: wrong status!');
   end if;
    SELECT r.status, r.trip_id
    INTO old_reserv_status, modifying_trip_id
    FROM RESERVATION r
    WHERE r.reservation_id = modifying_reservation_id;

   IF old_reserv_status IS NULL THEN
      RAISE_APPLICATION_ERROR(-20001, 'Error: trip not found.');
   END IF;

   IF old_reserv_status = new_status THEN
      RAISE_APPLICATION_ERROR(-20001, 'Error: You gave same status as
before.');
   END IF;

   UPDATE RESERVATION
    SET status = new_status
    WHERE reservation_id = modifying_reservation_id;
  COMMIT;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Rezerwacja o podanym ID nie istnieje
lub niepoprawny status');
end;
/
```

Wywołania:

Funkcjonalnosc procedur się nie zmieniła, więc przetestujmy działanie naszych triggerów.

Widok Upcoming_trips:

	COUNTRY	TRIP_DATE	TRIP_NAME	TRIP_ID	NO_PLACES	NO_AVAILABLE_PLACES
1	Francja	2023-08-15	Wakacje we Francji	2	4	0
2	Włochy	2023-06-01	Wycieczka po Włoszech	3	20	18
3	Hiszpania	2023-07-15	Wakacje w Hiszpanii	4	25	24

Dodajmy za pomocą Add_reservation_3 nową rezerwację dla trip o id=2:

```
DECLARE
    v_trip_id INT := 2;
    v_person_id INT := 5;
BEGIN
   ADD_RESERVATION_3(v_trip_id, v_person_id);
END;
/
```

```
[72000][20001]
ORA-20001: Trigger error: No available places!
ORA-06512: przy "BD_410788.TR_CHECK_BEFORE_ADD_RESERVATION", linia 4
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_410788.TR_CHECK_BEFORE_ADD_RESERVATION'
ORA-06512: przy "BD_410788.ADD_RESERVATION_3", linia ...
```

Teraz przetestujmy drugi trigger:

Zmieńmy ilość miejsc dla trip o id = 2, niech rezerwacja o id = 5 będzie wynosić C:

```
DECLARE
   v_reservation_id INT := 5;
   status char(1) := 'C';
BEGIN
   Modify_Reservation_Status_3(v_reservation_id, status);
END;
/
```

Sprawdzmy przy okazji czy dodalo dokonaly się zmiany w tabelach

Reservation:

RESERVATION_ID	TRIP_ID	PERSON_ID	STATUS
1	1	1	P
2	1	2	P
3	1	3	C
4	2	7	N
5	2	2	C

Log:

61	41	2023-03-29	N
62	8	2023-03-29	C
81	5	2023-03-29	C

Zmiany ilości miejsc:

```
DECLARE
   v_trip_id INT := 2;
   no_places INT := 3;
BEGIN
   Modify_No_Places(v_trip_id , no_places );
END;
/
```

	COUNTRY	TRIP_DATE	TRIP_NAME	TRIP_ID	NO_PLACES	NO_AVAILABLE_PLACES
1	Francja	2023-08-15	Wakacje we Francji	2	3	0

Teraz staramy się zmienić status rezerwacji nr 5:

```
DECLARE
   v_reservation_id INT := 5;
   status char(1) := 'N';
BEGIN
   Modify_Reservation_Status_3(v_reservation_id, status);
END;
/
```

```
[72000][20001]
ORA-20001: Trigger error: No available places!
ORA-06512: przy "BD_410788.TR_BEFORE_MODIFIED_RESERVATION", linia 8
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_410788.TR_BEFORE_MODIFIED_RESERVATION'
ORA-06512: przy "BD_410788.MODIFY_RESERVATION_STATUS_3", linia ...
```

Rozdział 9 Zmiana struktury bazy danych, w tabeli wycieczki dodajemy redundantne pole no_available_places

Musimy wykonać polecenie:

```
ALTER TABLE Trip ADD no_available_places INTEGER;
```

Tworzymy nową procedurę o nazwie Przelicz:

```
ALTER TABLE Trip ADD no_available_places INTEGER;

CREATE OR REPLACE PROCEDURE przelicz
IS
BEGIN
    FOR trip_rec IN (SELECT * FROM TRIP)
    LOOP
        UPDATE TRIP
        SET no_available_places = trip_rec.max_no_places - (SELECT COUNT(*)
FROM RESERVATION WHERE RESERVATION.trip_id = trip_rec.trip_id AND
RESERVATION.status != 'C')
        WHERE trip_id = trip_rec.trip_id;
    END LOOP;
    COMMIT;
END;
/
```

Wywołanie:

```
begin
    przelicz();
end;
```

Teraz w tabeli Trip otrzymamy:

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	1	Wakacje w Polsce	1	2022-07-01	20	18
2	2	Wakacje we Francji	2	2023-08-15	3	0
3	3	Wycieczka po Włoszech	3	2023-06-01	20	18
4	4	Wakacje w Hiszpanii	4	2023-07-15	25	24

Otrzymaliśmy teraz przeliczone pole no_available_places dla każdego wiersza tabeli Trip.

Widoki które ulegną zmianie:

Trips_4:

```
CREATE VIEW Trips_4 AS
SELECT
    countries.country_name AS country,
    trip.trip_date,
    trip.trip_name,
   trip.trip_id,
    trip.max_no_places AS no_places,
    trip.no_available_places
FROM
    trip
    INNER JOIN countries ON trip.country_id = countries.country_id;
```

Available_Trips_4:

```
CREATE VIEW Available_Trips_4 AS
SELECT * FROM TRIPS_4 t
WHERE t.no_available_places > 0 AND t.trip_date > SYSDATE;
```

Upcoming_Trips_4:

```
CREATE VIEW Upcoming_Trips_4 AS
SELECT * FROM TRIPS_4 t
WHERE t.trip_date > SYSDATE;
```

Widok Trips_4 został uproszczony gdyż usunięte zostało grupowanie, teraz bezpośrednio się odwołuje do rekordów tabeli Trip.

Zmiany w funkcjach:

```
CREATE OR REPLACE FUNCTION free_places_5(trip_id IN NUMBER) RETURN NUMBER
IS
    available_places NUMBER;
BEGIN
    SELECT T.no_available_places INTO available_places FROM TRIP T WHERE
t.TRIP_ID = free_places_5.trip_id;
    RETURN available_places;
END;
/
```

Ta pomocnicza dla triggerów funkcja została uproszczona, teraz może bezwzględnie korzystać z pola w tabeli Trip. Ponieważ triggery dojdą w zadaniu 10, nadałem suffix 5.

Available_trips_func_4:

```
create or replace type available_trips_obj_4 as OBJECT
(
trip_date DATE,
trip_name varchar(100),
NO_places int,
NO_available_places int
);
create or replace type available_trips_table_4 is table of
available_trips4;

create or replace function available_Trips_Func_4(country_name varchar,
date_from DATE, date_to DATE)
return available_trips_table4
```

```
AS
result available_trips_table4;
null_arg exception;
wrong_order exception ;
tmp char(1);
begin
IF country_name is null or date_from is null or date_to is null then
raise_application_error(-20003,'przeslana wartosc to null');
END IF;

IF date_to < date_from THEN
raise_application_error(-20002,'from_data wieksze od to_data');
END IF;
-- sprawdzenie, czy istnieje wycieczka o podanym country_name i czasie
select 1 into tmp from Available_Trips_4 avt
where avt.country = available_Trips_Func_4.country_name
AND avt.trip_date BETWEEN available_Trips_Func_4.date_from AND
available_Trips_Func_4.date_to;

--zapytanie wsadzone do result
select available_trips4(avt.trip_date,
avt.trip_name, avt.no_places, avt.no_available_places)
bulk collect into result
from Available_Trips_4 avt
where avt.country = available_Trips_Func_4.country_name
AND avt.trip_date BETWEEN available_Trips_Func_4.date_from AND
available_Trips_Func_4.date_to;
--zwracanie wyniku
return result;
EXCEPTION
WHEN NO_DATA_FOUND THEN
raise_application_error(-20001,'wycieczka z podanym country_name oraz datą
nie istnieje lub się odbyła');
RETURN NULL;
WHEN others THEN
raise_application_error(-20004,'Wystapil blad!');
RETURN NULL;
END;
```

W tej funkcji jedynie zmieniły się nazwy widoków do których należy się odwołać tzn Available_Trips_4. Cała reszta zostaje taka sama.

Add_Reservation_4:

```
create or replace PROCEDURE Add_Reservation_4(trip_id int, person_id int)
IS
tmp_person char(1);
tmp_trip char(1);
not_exist inT;
id int;
max_places_was_already int;
begin
    --sprawdzenie wystapienia null
    if trip_id is null or person_id is null then
        raise_application_error(-20002, 'Przesłana wartość jest równa
NULL');
    end if;
    --sprawdzenie czy osoba o podanym id istnieje
    select 1 into tmp_person FROM PERSON p WHERE
Add_Reservation_4.person_id = p.person_id;
```

```
    --sprawdzenie czy wycieczka o podanym id istnieje
    select 1 into tmp_trip FROM TRIP t WHERE Add_Reservation_4.trip_id =
t.trip_id;
    --sprawdzenie czy osoba o podanym id nie ma juz wybranej wycieczki o
podanym id i czy status rezerwacji != C
    select case
        when exists(select * FROM RESERVATION r
                            WHERE r.PERSON_ID =
Add_Reservation_4.person_id
                           AND r.trip_id = Add_Reservation_4.trip_id) then
0
        else 1
        end
    into not_exist from dual;

    if not_exist = 0 then
        raise_application_error(-20003, 'Osoba z podaną nieodwołaną
rezerwacją już istnieje lub ma odwolaną rezerwację, skorzystaj z
modifyReservationStatus()');
    end if;
    --sprawdzamy czy wycieczka sie juz nie odbyla
   --sprawdzamy czy są wolne miejsca
    select case
            when exists(select * FROM AVAILABLE_TRIPS_4 t
                                WHERE t.trip_id =
Add_Reservation_4.trip_id ) then 1
            else 0
            end
        into max_places_was_already from dual;
        if max_places_was_already = 0 then
            raise_application_error(-20004, 'Nie można dodać rezerwacji dla
wycieczki, która już się odbyła lub nie ma wolnych miejsc');
        end if;

   --dodajemy osobę
    INSERT INTO RESERVATION(trip_id,person_id,status)
VALUES(Add_Reservation_4.trip_id,Add_Reservation_4.person_id,'N')
    returning reservation_id into id;
    dbms_output.PUT_LINE(id);
    INSERT INTO LOG(RESERVATION_ID,LOG_DATE,status)
VALUES(id,TO_CHAR(SYSDATE, 'YYYY-MM-DD'),'N');

    UPDATE TRIP t SET
    t.NO_AVAILABLE_PLACES = t.NO_AVAILABLE_PLACES - 1
    WHERE t.trip_id = Add_Reservation_4.trip_id;

    COMMIT;

EXCEPTION
   WHEN NO_DATA_FOUND THEN
        raise_application_error(-20001, 'Nie znaleziono wycieczki lub osoby
z podanym ID');
END;
/
```

Jest to przybliżona wersja Add_Reservation, jedynia zmiana, to występujący Update pola no_available_places w trip dla podanego trip_id

Wywolanie:

```
DECLARE
v_person_id INT := 8;
v_trip_id INT := 4;
begin
    Add_Reservation_4(v_trip_id,v_person_id);

end;
```

Analogicznie jak poprzednie wersje w tabeli Reservation i Log wystąpią nowe dane.

Tabela trip:

Przed:

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	1	Wakacje w Polsce	1	2022-07-01	20	18
2	2	Wakacje we Francji	2	2023-08-15	3	0
3	3	Wycieczka po Włoszech	3	2023-06-01	20	18
4	4	Wakacje w Hiszpanii	4	2023-07-15	25	24

Po:

WHERE ORDER BY

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	1	Wakacje w Polsce	1	2022-07-01	20	18
2	2	Wakacje we Francji	2	2023-08-15	3	0
3	3	Wycieczka po Włoszech	3	2023-06-01	20	18
4	4	Wakacje w Hiszpanii	4	2023-07-15	25	23

Modify_Reservation_Status_4:

```
create or replace PROCEDURE
Modify_Reservation_Status_4(modifying_reservation_id int, new_status char)
IS
   can_modify int;
    old_reserv_status char(1);
   modifying_trip_id int;
    v_value int;
BEGIN
   if new_status NOT IN ('C','N','P') THEN
       RAISE_APPLICATION_ERROR(-20001,'Error: wrong status!');
   end if;
   SELECT r.status, r.trip_id
    INTO old_reserv_status, modifying_trip_id
    FROM RESERVATION r
    WHERE r.reservation_id = modifying_reservation_id;

   IF old_reserv_status IS NULL THEN
      RAISE_APPLICATION_ERROR(-20001, 'Error: trip not found.');
   END IF;

   IF old_reserv_status = new_status THEN
      RAISE_APPLICATION_ERROR(-20001, 'Error: You gave same status as
before.');
```

```
   END IF;

IF old_reserv_status = 'C' THEN
        SELECT CASE
                    WHEN EXISTS(SELECT *
                                FROM AVAILABLE_TRIPS_4 avt
                                WHERE avt.trip_id = modifying_trip_id) THEN
1
                    ELSE 0
                    END
        INTO can_modify
        FROM DUAL;
        IF can_modify = 0 THEN
            RAISE_APPLICATION_ERROR(-20001, 'Error: the trip has already
taken place or there are no vacancies.');
        END IF;
    END IF;

   IF new_status = 'C' THEN
      v_value := 1;
   ELSE
       v_value := -1;
      END IF;
   IF new_status in ('N','P') and old_reserv_status in ('N','P') THEN
      v_value := 0;
    END IF;
   --update rezerwacji
   UPDATE RESERVATION
    SET status = new_status
    WHERE reservation_id = modifying_reservation_id;
   --update ilosci miejsc
   UPDATE TRIP t SET
    t.NO_AVAILABLE_PLACES = t.NO_AVAILABLE_PLACES + v_value
    WHERE t.trip_id = modifying_trip_id;
  -- dodanie nowego wpisu do LOG
  INSERT INTO LOG(RESERVATION_ID, LOG_DATE, status)
VALUES(modifying_reservation_id, TO_CHAR(SYSDATE, 'YYYY-MM-
DD'),new_status);
  COMMIT;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Rezerwacja o podanym ID nie istnieje
lub niepoprawny status');
end;
/
```

Ta procedura różni się od Modify_Reservation_Status tym, że tworzy zmienną przechowującą wartość 1 0 lub -1 w zależności od wartości starych i nowych statusów. Służy to do zmiany no_available_places z trip. Wykonuje się dodatkowo Update tabeli trip podobnie jak przy add_reservation_4.

Wywolanie:

```
DECLARE
   v_reservation_id INT := 8;
   status char(1) := 'N';
BEGIN
   Modify_Reservation_Status_4(v_reservation_id, status);
END;
/
```

Tabela trip:

Przed:

WHERE ORDER BY

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	1	Wakacje w Polsce	1	2022-07-01	20	18
2	2	Wakacje we Francji	2	2023-08-15	3	0
3	3	Wycieczka po Włoszech	3	2023-06-01	20	18
4	4	Wakacje w Hiszpanii	4	2023-07-15	25	23

Po:

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	1	Wakacje w Polsce	1	2022-07-01	20	18
2	2	Wakacje we Francji	2	2023-08-15	3	0
3	3	Wycieczka po Włoszech	3	2023-06-01	20	17
4	4	Wakacje w Hiszpanii	4	2023-07-15	25	23

Modify_NO_Places_4:

```
create or replace PROCEDURE Modify_No_Places_4(trip_id INT, no_places INT)
AS
tmp_reserv0 INT;
reserved_places INT;
new_available_place INT;
BEGIN
  if trip_id is null or trip_id < 0 or no_places is null or no_places < 0
then
        raise_application_error(-20000, 'Nie mozna podawać wartosci
nullowej lub mniejszej od 0');
 END IF;
    -- Sprawdzenie, czy wycieczka istnieje
  SELECT 1 INTO tmp_reserv0 FROM trip t WHERE t.trip_id  =
Modify_No_Places_4.trip_id;
  -- wydobycie ilosci dostepnych miejsc
  SELECT uptr.NO_PLACES - uptr.NO_AVAILABLE_PLACES,
         uptr.NO_AVAILABLE_PLACES - uptr.NO_PLACES +
Modify_No_Places_4.no_places
         INTO reserved_places,new_available_place FROM UPCOMING_TRIPS_4
uptr WHERE uptr.TRIP_ID = Modify_No_Places_4.trip_id;



  IF (Modify_No_Places_4.no_places < reserved_places) THEN
    RAISE_APPLICATION_ERROR(-20002, 'Ilosc zarezerwowanych miejsc na daną
```

```
wycieczkę musi byc mniejsza od nowej ilosc miejsc');
  END IF;

  -- Aktualizacja liczby miejsc dla wycieczki
  UPDATE TRIP t SET
    t.MAX_NO_PLACES = Modify_No_Places_4.no_places,t.NO_AVAILABLE_PLACES =
Modify_No_Places_4.new_available_place
    WHERE t.trip_id = Modify_No_Places_4.trip_id;
  COMMIT;

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Nie znaleziono wycieczki o zadanym
id,mozliwe ze wycieczka juz sie odbyla albo niepoprawne id');
END;
/
```

W tej procedurze musimy dodać Update dla tabeli trip w dwóch miejscach: dla maksymalnej ilości miejsc i liczby dostępnych miejsc. Poza tym, będzie działać analogicznie jak poprzednio.

Wywołanie:

```
DECLARE
   v_trip_id INT := 2;
   no_places INT := 20;
BEGIN
   Modify_No_Places_4(v_trip_id , no_places );
END;
/
```

Tabela Trip:

Przed:

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	1	Wakacje w Polsce	1	2022-07-01	20	18
2	2	Wakacje we Francji	2	2023-08-15	3	0
3	3	Wycieczka po Włoszech	3	2023-06-01	20	17
4	4	Wakacje w Hiszpanii	4	2023-07-15	25	23

Po:

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	1	Wakacje w Polsce	1	2022-07-01	20	18
2	2	Wakacje we Francji	2	2023-08-15	20	17
3	3	Wycieczka po Włoszech	3	2023-06-01	20	17
4	4	Wakacje w Hiszpanii	4	2023-07-15	25	23

Rozdział 10 Zmiana strategii obsługi redundantnego pola no_available_places, realizacja przy pomocy trigerów

Wykorzystamy fakt, że w zadaniu 7 zdefiniowaliśmy triggery dla dodawania nowych logów, więc nie trzeba będzie uwzględniać inserta nowego wiersza do tabeli log w podanych procedurach.

Triggery:

Tr_check_before_add_reservation_5:

```
create or replace trigger TR_CHECK_BEFORE_ADD_RESERVATION_5
before insert
on RESERVATION
for each row
DECLARE PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
IF free_places_5(:NEW.trip_id) <= 0 THEN
RAISE_APPLICATION_ERROR(-20001, 'No available places');
END IF;
END;
/
```

Tr_before_modified_reservation_5:

```
create trigger TR_BEFORE_MODIFIED_RESERVATION_5
before update of STATUS
on RESERVATION
for each row
DECLARE PRAGMA AUTONOMOUS_TRANSACTION;
trip INT;
BEGIN
IF (:NEW.STATUS = 'P' OR :NEW.STATUS = 'N') AND :OLD.STATUS = 'C'
THEN
SELECT TRIP_ID into trip FROM RESERVATION WHERE RESERVATION_ID =
:OLD.reservation_id;
IF free_places_5(trip) <= 0 THEN
RAISE_APPLICATION_ERROR(-20001, 'No more available places');
END IF;
END IF;
END;
/
```

Te dwa triggery nie zmienily się praktycznie wcale. Jedynie ich funkcja pomocnicza została zmieniona.

Tr_Modify_no_places_5:

```
create trigger TR_MODIFY_NOPLACES_5
before update of no_available_places
on TRIP
for each row
DECLARE PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
IF :NEW.no_available_places < 0 THEN
RAISE_APPLICATION_ERROR(-20001, 'trigger error: new nr of available places
will be less than zero if you do that');
END IF;

END;
/
```

Ten trigger wykonuje się przed updatem no_available_places, obsługuje błędy, jakie mogą wyniknąć, np. możliwość wystąpienia ujemnych możliwych miejsc.

Procedury w których dokonują się zmiany:

Add_Reservation_5:

```
create or replace PROCEDURE Add_Reservation_5(trip_id int, person_id int)
IS
tmp_person char(1);
tmp_trip char(1);
not_exist inT;
id int;
begin
    --sprawdzenie wystapienia null
    if trip_id is null or person_id is null then
        raise_application_error(-20002, 'Przesłana wartość jest równa
NULL');
    end if;
    --sprawdzenie czy osoba o podanym id istnieje
    select 1 into tmp_person FROM PERSON p WHERE
Add_Reservation_5.person_id = p.person_id;
    --sprawdzenie czy wycieczka o podanym id istnieje
    select 1 into tmp_trip FROM TRIP t WHERE Add_Reservation_5.trip_id =
t.trip_id;
    --sprawdzenie czy osoba o podanym id nie ma juz wybranej wycieczki o
podanym id i czy status rezerwacji != C
    select case
        when exists(select * FROM RESERVATION r
                            WHERE r.PERSON_ID =
Add_Reservation_5.person_id
                           AND r.trip_id = Add_Reservation_5.trip_id) then
0
        else 1
        end
    into not_exist from dual;

    if not_exist = 0 then
        raise_application_error(-20003, 'Osoba z podaną nieodwołaną
rezerwacją już istnieje lub ma odwolaną rezerwację, skorzystaj z
modifyReservationStatus()');
    end if;
   --dodajemy osobę
    INSERT INTO RESERVATION(trip_id,person_id,status)
```

```
VALUES(Add_Reservation_5.trip_id,Add_Reservation_5.person_id,'N')
    returning reservation_id into id;
    dbms_output.PUT_LINE(id);


    UPDATE TRIP t SET
    t.NO_AVAILABLE_PLACES = t.NO_AVAILABLE_PLACES - 1
    WHERE t.trip_id = Add_Reservation_5.trip_id;
    COMMIT;

EXCEPTION
   WHEN NO_DATA_FOUND THEN
        raise_application_error(-20001, 'Nie znaleziono wycieczki lub osoby
z podanym ID');
END;
/
```

Modify_Reservation_Status_5:

```
create or replace PROCEDURE
Modify_Reservation_Status_5(modifying_reservation_id int, new_status char)
IS
    old_reserv_status char(1);
   modifying_trip_id int;
    v_value int;
BEGIN
   if new_status NOT IN ('C','N','P') THEN
       RAISE_APPLICATION_ERROR(-20001,'Error: wrong status!');
   end if;
   SELECT r.status, r.trip_id
    INTO old_reserv_status, modifying_trip_id
    FROM RESERVATION r
    WHERE r.reservation_id = modifying_reservation_id;

   IF old_reserv_status IS NULL THEN
      RAISE_APPLICATION_ERROR(-20001, 'Error: trip not found.');
   END IF;

   IF old_reserv_status = new_status THEN
      RAISE_APPLICATION_ERROR(-20001, 'Error: You gave same status as
before.');
   END IF;

   IF new_status = 'C' THEN
      v_value := 1;
   ELSE
       v_value := -1;
      END IF;
   IF new_status in ('N','P') and old_reserv_status in ('N','P') THEN
      v_value := 0;
    END IF;
   --update rezerwacji
   UPDATE RESERVATION
    SET status = new_status
    WHERE reservation_id = modifying_reservation_id;
   --update ilosci miejsc
   UPDATE TRIP t SET
```

```
    t.NO_AVAILABLE_PLACES = t.NO_AVAILABLE_PLACES + v_value
    WHERE t.trip_id = modifying_trip_id;
  COMMIT;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Rezerwacja o podanym ID nie istnieje
lub niepoprawny status');
end;
/
```

Modify_No_Places_5

```
create or replace PROCEDURE Modify_No_Places_5(trip_id INT, no_places INT)
AS
tmp_reserv0 INT;
reserved_places INT;
new_available_place INT;
BEGIN
  if trip_id is null or trip_id < 0 or no_places is null or no_places < 0
then
        raise_application_error(-20000, 'Nie mozna podawać wartosci
nullowej lub mniejszej od 0');
 END IF;
    -- Sprawdzenie, czy wycieczka istnieje
  SELECT 1 INTO tmp_reserv0 FROM trip t WHERE t.trip_id  =
Modify_No_Places_5.trip_id;
  -- wydobycie ilosci dostepnych miejsc
  SELECT uptr.NO_PLACES - uptr.NO_AVAILABLE_PLACES,
         uptr.NO_AVAILABLE_PLACES - uptr.NO_PLACES +
Modify_No_Places_5.no_places
         INTO reserved_places,new_available_place FROM UPCOMING_TRIPS_4
uptr WHERE uptr.TRIP_ID = Modify_No_Places_5.trip_id;

  -- Aktualizacja liczby miejsc dla wycieczki
  UPDATE TRIP t SET
    t.MAX_NO_PLACES = Modify_No_Places_5.no_places,t.NO_AVAILABLE_PLACES =
Modify_No_Places_5.new_available_place
    WHERE t.trip_id = Modify_No_Places_5.trip_id;
  COMMIT;

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20001, 'Nie znaleziono wycieczki o zadanym
id,mozliwe ze wycieczka juz sie odbyla albo niepoprawne id');
END;
/
```

Przykładowe wywołania:

Trip:

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	1	Wakacje w Polsce	1	2022-07-01	20	18
2	2	Wakacje we Francji	2	2023-08-15	20	17
3	3	Wycieczka po Włoszech	3	2023-06-01	3	0
4	4	Wakacje w Hiszpanii	4	2023-07-15	25	23

```
DECLARE
   v_trip_id INT := 3;
   no_places INT := 2;
BEGIN
   Modify_No_Places_5(v_trip_id , no_places );
END;
/
```

```
[72000][20001]
ORA-20001: trigger error: new nr of available places will be less than zero if you do that
ORA-06512: przy "BD_410788.TR_MODIFY_NOPLACES_5", linia 4
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_410788.TR_MODIFY_NOPLACES_5'
ORA-06512: przy "BD_410788.MODIFY_NO_PLACES_5", linia ...
```

```
DECLARE
   v_trip_id INT := 3;
   v_person_id INT := 10;
BEGIN
   ADD_RESERVATION_5(v_trip_id , v_person_id );
END;
/
```

```
[72000][20001]
ORA-20001: No available places
ORA-06512: przy "BD_410788.TR_CHECK_BEFORE_ADD_RESERVATION_5", linia 4
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_410788.TR_CHECK_BEFORE_ADD_RESERVATION_5'
ORA-06512: przy "BD_410788.ADD_RESERVATION_5", linia ...
```

Tabela reservation:

	RESERVATION_ID	TRIP_ID	PERSON_ID	STATUS
1	1	1	1	P
2	2	1	2	P
3	3	1	3	C
4	4	2	7	N
5	5	2	2	C
6	6	2	3	P
7	7	3	9	P
8	8	3	6	N
9	9	3	4	C
10	10	4	9	N
11	21	2	8	N
12	41	3	2	N
13	81	4	8	N

```
DECLARE
   v_reservation_id INT := 9;
   status char(1) := 'N';
BEGIN
   Modify_Reservation_Status_5(v_reservation_id, status);
END;
/
```

```
[72000][20001]
ORA-20001: No more available places
ORA-06512: przy "BD_410788.TR_BEFORE_MODIFIED_RESERVATION_5", linia 8
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_410788.TR_BEFORE_MODIFIED_RESERVATION_5'
ORA-06512: przy "BD_410788.MODIFY_RESERVATION_STATUS_5", linia ...
```

Jak widać powyżej kontrola triggerów zadziałała dla tych trzech procedur.