






Mateusz Skowron

1. TABELE

a. **log** - tabelę dziennikującą zmiany statusu rezerwacji, oraz liczby zarezerwowanych miejsc.

```
create table log
(
  log_id int generated always as identity not null
, reservation_id int
, log_date date
, status char(1)
, no_places int
, constraint log_pk primary key
(
  log_id
)
enable
);

alter table log
add constraint log_chk1 check
(status in ('n','p','c'))
enable;
```

	 LOG_ID ↕	 RESERVATION_ID ↕	 LOG_DATE ↕	 STATUS ↕	 NO_PLACES ↕
1	2	4	2022-03-16 20:47:55	n	1
2	3	4	2022-03-16 20:48:04	n	2
3	4	4	2022-03-16 20:48:22	n	0
4	5	4	2022-03-16 20:48:38	n	2
5	6	4	2022-03-16 20:50:27	c	2
6	7	4	2022-03-16 20:50:40	p	2

2. WIDOKI

a. reservations – widok zawierający informacje o rezerwacjach.

```
create view reservations as
select
r.RESERVATION_ID,r.NO_PLACES,r.STATUS,p.PERSON_ID,p.FIRSTNAME,p.LASTNAME,t.TRIP_ID,t.NAME,t.COUNTRY,t.TRIP_DATE
from RESERVATION r
join TRIP t on r.TRIP_ID = t.TRIP_ID
join PERSON p on r.PERSON_ID = p.PERSON_ID;
```

	RESERVATION_ID	NO_PLACES	STATUS	PERSON_ID	FIRSTNAME	LASTNAME	TRIP_ID	NAME	COUNTRY	TRIP_DATE
1	1	1	n	1	adam	kowalski	1	wycieczka do paryza	francja	2021-09-03
2	2	2	p	2	jan	nowak	1	wycieczka do paryza	francja	2021-09-03
3	3	2	c	3	mateusz	skowron	1	wycieczka do paryza	francja	2021-09-03
4	4	2	p	4	alicja	krak	2	wycieczka do krakowa	polaska	2022-12-05
5	5	1	c	5	adam	mickiewicz	2	wycieczka do krakowa	polaska	2022-12-05
6	6	2	n	6	juliusz	słowacki	3	wycieczka na AGH	polaska	2022-05-05
7	7	1	n	7	mieszko	pierwszy	3	wycieczka na AGH	polaska	2022-05-05
8	8	3	c	8	karol	wojdyla	3	wycieczka na AGH	polaska	2022-05-05
9	9	1	p	9	piotr	kosmowski	4	wycieczka do bracelony	hiszpania	2023-05-05
10	10	1	c	10	olivia	brzeg	4	wycieczka do bracelony	hiszpania	2023-05-05

b. trips_no_reserved_places – widok zawierający informacje o liczbie obecnie zarezerwowanych miejsc na daną wycieczkę.

```
create view trips_no_reserved_places as
select TRIP_ID,SUM(NO_PLACES) as no_reserved_places
from RESERVATION
where STATUS != 'c'
group by TRIP_ID
union
select TRIP_ID,0 as no_reserved_places
from RESERVATION
where STATUS = 'c' AND TRIP_ID not in (select TRIP_ID from RESERVATION WHERE STATUS!='c')
group by TRIP_ID
union
select TRIP_ID,0 as no_reserved_places
from TRIP
where TRIP_ID not in (select unique TRIP_ID from RESERVATION);
```

	TRIP_ID	NO_RESERVED_PLACES
1	1	3
2	2	2
3	3	3
4	4	1
5	5	0

c. trips - widok zawierający informacje o wycieczkach wraz z liczbą dostępnych na nie miejsc.

```
create view trips as
select t.TRIP_ID,t.COUNTRY,t.TRIP_DATE,t.NAME,t.MAX_NO_PLACES,t.MAX_NO_PLACES - tr.NO_RESERVED_PLACES as
no_available_places
from TRIP t
join trips_no_reserved_places tr on t.TRIP_ID = tr.TRIP_ID;
```

	TRIP_ID	COUNTRY	TRIP_DATE	NAME	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	1	francja	2021-09-03	wycieczka do paryza	3	0
2	2	polska	2022-12-05	wycieczka do krakowa	2	0
3	3	polska	2022-05-05	wycieczka na AGH	3	0
4	4	hiszpania	2023-05-05	wycieczka do brצלony	1	0
5	5	niemcy	2022-10-15	wycieczka do berlina	3	3

d. available_trips - widok zawierający obecnie dostępne wycieczki wraz z liczbą dostępnych na nie miejsc.

```
create view available_trips as
select TRIP_ID, NAME, COUNTRY, TRIP_DATE, MAX_NO_PLACES, NO_AVAILABLE_PLACES
from TRIPS
WHERE NO_AVAILABLE_PLACES > 0 AND TRIP_DATE > SYSDATE;
```

	TRIP_ID	NAME	COUNTRY	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	5	wycieczka do berlina	niemcy	2022-10-15	3	3

3. FUNKCJE

a. **f_reservation_exists** - funkcja zwraca true jeśli rezerwacja o podanym ID istnieje, false w przeciwnym razie.

```
create or replace function f_reservation_exists(r_id in RESERVATION.RESERVATION_ID%type)
return boolean
as
    exist number;
begin
    select
        case
            when exists(select * from RESERVATION where RESERVATION_ID = r_id) then 1
            else 0
        end
    into exist
    from DUAL;

    if exist = 1 then
        return true;
    else
        return false;
    end if;
end;
```

```
BD_MSKOWRON> begin
    if F_RESERVATION_EXISTS(1) then
        DBMS_OUTPUT.PUT_LINE(1);
    else
        DBMS_OUTPUT.PUT_LINE(0);
    end if;

    if F_RESERVATION_EXISTS(500) then
        DBMS_OUTPUT.PUT_LINE(1);
    else
        DBMS_OUTPUT.PUT_LINE(0);
    end if;
end;
[2022-03-16 21:29:47] completed in 147 ms
[2022-03-16 21:29:47] 1
[2022-03-16 21:29:47] 0
```

b. f_trip_exists – funkcja zwraca true jeśli wycieczka o podanym ID istnieje, false w przeciwnym razie.

```
create or replace function f_trip_exists(t_id in TRIP.trip_id%type)
return boolean
as
    exist number;
begin
    select
        case
            when exists(select * from TRIP where TRIP_ID = t_id) then 1
            else 0
        end
    into exist
    from DUAL;

    if exist = 1 then
        return true;
    else
        return false;
    end if;
end;
```

```
SQL> begin
        if F_TRIP_EXISTS(1) then
            DBMS_OUTPUT.PUT_LINE(1);
        else
            DBMS_OUTPUT.PUT_LINE(0);
        end if;

        if F_TRIP_EXISTS(500) then
            DBMS_OUTPUT.PUT_LINE(1);
        else
            DBMS_OUTPUT.PUT_LINE(0);
        end if;
    end;
[2022-03-16 21:32:33] completed in 125 ms
[2022-03-16 21:32:33] 1
[2022-03-16 21:32:33] 0
```

c. f_country_trip_exists - funkcja zwraca true jeśli istnieje chociaż jedna wycieczka do danego państwa, false w przeciwnym razie.

```
create or replace function f_country_trip_exists(c in trip.COUNTRY%type)
return boolean
as
exist number;
begin
select
    case
        when exists(select * from TRIP where country = c) then 1
        else 0
    end
into exist
from DUAL;

if exist = 1 then
    return true;
else
    return false;
end if;
end;
```

```
BD_MSKOWRON> begin
    if F_COUNTRY_TRIP_EXISTS('polska') then
        DBMS_OUTPUT.PUT_LINE(1);
    else
        DBMS_OUTPUT.PUT_LINE(0);
    end if;

    if F_COUNTRY_TRIP_EXISTS('portugalia') then
        DBMS_OUTPUT.PUT_LINE(1);
    else
        DBMS_OUTPUT.PUT_LINE(0);
    end if;

end;
[2022-03-16 21:34:59] completed in 162 ms
[2022-03-16 21:34:59] 1
[2022-03-16 21:34:59] 0
```


d. f_person_exists - funkcja zwraca true jeśli osoba o podanym ID istnieje, false w przeciwnym razie.

```
create or replace function f_person_exists(p_id in PERSON.person_id%type)
return boolean
as
    exist number;
begin
    select
        case
            when exists(select * from PERSON where PERSON_ID = p_id) then 1
            else 0
        end
    into exist
    from DUAL;

    if exist = 1 then
        return true;
    else
        return false;
    end if;
end;
```

```
BD_MSKOWRON> begin
    if F_PERSON_EXISTS(1) then
        DBMS_OUTPUT.PUT_LINE(1);
    else
        DBMS_OUTPUT.PUT_LINE(0);
    end if;

    if F_PERSON_EXISTS(500) then
        DBMS_OUTPUT.PUT_LINE(1);
    else
        DBMS_OUTPUT.PUT_LINE(0);
    end if;
end;
[2022-03-16 21:35:47] completed in 116 ms
[2022-03-16 21:35:47] 1
[2022-03-16 21:35:47] 0
```

e. **f_trip_participants** - funkcja zwraca uczestników wycieczki o danym ID w postaci kolekcji będącej typem tablicowym *trip_participant_table*.

```
create or replace function f_trip_participants(trip_id in TRIP.trip_id%type)
  return trip_participant_table
as
  result trip_participant_table;
begin
  if not F_TRIP_EXISTS(trip_id) then
    raise_application_error(-20001,'trip not found');
  end if;

  select trip_participant(t.COUNTRY,t.TRIP_DATE,t.NAME,r.RESERVATION_ID,p.FIRSTNAME,p.LASTNAME)
  bulk collect
  into result
  from RESERVATION r join PERSON p
  on r.PERSON_ID = p.PERSON_ID
  join TRIP t
  on r.TRIP_ID = t.TRIP_ID
  where r.TRIP_ID = f_trip_participants.trip_id and r.STATUS != 'c';

  return result;

end;
```

```
BD_MSKOWRON> declare
    kolekcja TRIP_PARTICIPANT_TABLE := TRIP_PARTICIPANT_TABLE();
begin
    kolekcja := F_TRIP_PARTICIPANTS(1);

    FOR i IN kolekcja.FIRST..kolekcja.LAST
    LOOP
        DBMS_OUTPUT.PUT_LINE(kolekcja(i).firstname||' '||kolekcja(i).lastname);
    END LOOP;

end;
[2022-03-16 21:38:15] completed in 99 ms
[2022-03-16 21:38:16] adam kowalski
[2022-03-16 21:38:16] jan nowak
```

f. **f_person_reservations** – funkcja zwraca rezerwacje osoby o zadanym ID w postaci kolekcji będącej typem tablicowym *person_reservation_table*.

```
create or replace function f_person_reservations(person_id in PERSON.person_id%type)
  return person_reservation_table
as
  result person_reservation_table;
begin
  if not F_PERSON_EXISTS(person_id) then
    raise_application_error(-20001,'person not found');
  end if;

  select person_reservation(t.COUNTRY,t.TRIP_DATE,t.NAME,r.RESERVATION_ID)
  bulk collect
  into result
  from RESERVATION r join PERSON p
  on r.PERSON_ID = p.PERSON_ID
  join TRIP t
  on r.TRIP_ID = t.TRIP_ID
  where r.PERSON_ID = f_person_reservations.person_id;

  return result;

end;
```

```
BD_MSKOWRON> declare
  kolekcja PERSON_RESERVATION_TABLE := PERSON_RESERVATION_TABLE();
begin
  kolekcja := F_PERSON_RESERVATIONS(1);

  FOR i IN kolekcja.FIRST..kolekcja.LAST
  LOOP
    DBMS_OUTPUT.PUT_LINE(kolekcja(i).reservation_id||'. '||kolekcja(i).trip_name || ' '|| kolekcja(i).trip_date );
  END LOOP;
end;
[2022-03-16 21:39:55] completed in 137 ms
[2022-03-16 21:39:55] 1. wycieczka do paryza 21/09/03
```

g. **f_get_free_places_trip** - funkcja zwraca liczbę dostępnych miejsc na daną wycieczkę.

```
create or replace function f_get_free_places_trip(t_id in TRIP.trip_id%type)
return TRIPS.NO_AVAILABLE_PLACES%type
as
result TRIPS.NO_AVAILABLE_PLACES%type;
begin
select NO_AVAILABLE_PLACES into result from trips where TRIP_ID = t_id;
return result;
end;
```

```
BD_MSKOWRON> declare
    res int;
begin
    res := F_GET_FREE_PLACES_TRIP(5);
    DBMS_OUTPUT.PUT_LINE(res);
end;
[2022-03-16 21:41:18] completed in 130 ms
[2022-03-16 21:41:18] 3
```

h. f_available_trips_to - funkcja zwraca dostępne wycieczki do danego państwa w postaci kolekcji będącej typem tablicowym *available_trip_table*.

```
create or replace function f_available_trips_to(country trip.COUNTRY%type,date_from date,date_to date)
return available_trip_table
as
    result available_trip_table;
begin
    if not F_COUNTRY_TRIP_EXISTS(country) then
        raise_application_error(-20001,'Trips to this country not found.');
```

end if;

```
    select available_trip(TRIP_DATE,NAME,NO_AVAILABLE_PLACES)
    bulk collect
    into result
    from AVAILABLE_TRIPS r
    where r.COUNTRY = f_available_trips_to.country
    AND TRIP_DATE BETWEEN date_from AND date_to;
    return result;
end;
```

```
BD_MSKOWRON> declare
    kolekcja AVAILABLE_TRIP_TABLE := AVAILABLE_TRIP_TABLE();
begin
    kolekcja := F_AVAILABLE_TRIPS_TO('niemcy',to_date('2022-01-01','yyyy-mm-dd'),to_date('2022-12-31','yyyy-mm-dd'));

    FOR i IN kolekcja.FIRST..kolekcja.LAST
    LOOP
        DBMS_OUTPUT.PUT_LINE(kolekcja(i).trip_name||' '||kolekcja(i).trip_date || ' '|| kolekcja(i).no_available_places );
    END LOOP;
end;
```

[2022-03-16 21:44:27] completed in 136 ms
[2022-03-16 21:44:27] wycieczka do berlina 22/10/15 3

4. PROCEDURE

a. **add_reservation** - procedura dodająca rezerwację.

```
create or replace procedure add_reservation(t_id int,p_id int,n_places int)
as
    t_date date;
begin
    if not TRIP_EXISTS(t_id) then
        raise_application_error(-20001,'trip not found');
    end if;
    if not person_exists(p_id) then
        raise_application_error(-20001,'person not found');
    end if;
    select TRIP_DATE into t_date from TRIP where TRIP_ID = t_id;
    if t_date <= SYSDATE then
        raise_application_error(-20001,'The tour has already taken place. ');
    end if;

    insert into reservation(trip_id, person_id, no_places, status)
    values (t_id,p_id,n_places,'n');
end;
```

Na wycieczkę były dostępne jeszcze dwa miejsca.

```
BD_MSKOWRON> begin
                ADD_RESERVATION(5,1,1);
            end;
[2022-03-16 21:47:15] completed in 127 ms
BD_MSKOWRON> select *
                from RESERVATIONS
```

	RESERVATION_ID	NO_PLACES	STATUS	PERSON_ID	FIRSTNAME	LASTNAME	TRIP_ID	NAME	COUNTRY	TRIP_DATE
1	1	1	n	1	adam	kowalski	1	wycieczka do paryza	francja	2021-09-03
2	13	1	n	1	adam	kowalski	5	wycieczka do berlina	niemcy	2022-10-15
3	2	2	p	2	jan	nowak	1	wycieczka do paryza	francja	2021-09-03

Na wycieczkę nie ma dostępnych miejsc.

```
BD_MSKOWRON> begin
                ADD_RESERVATION(5,1,10);
            end;
[2022-03-16 22:01:53] [72000][20001]
[2022-03-16 22:01:53]   ORA-20001: Not enough place.
[2022-03-16 22:01:53]   ORA-06512: przy "BD_MSKOWRON.BI_RESERVATION", linia 6
[2022-03-16 22:01:53]   ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_MSKOWRON.BI_RESERVATION'
[2022-03-16 22:01:53]   ORA-06512: przy "BD_MSKOWRON.ADD_RESERVATION", linia 16
[2022-03-16 22:01:53]   ORA-06512: przy linia 2
[2022-03-16 22:01:53] Position: 0
```

b. modify_reservation_status - procedura służąca do modyfikacji statusu rezerwacji.

```
create or replace procedure modify_reservation_status(r_id int,st RESERVATION.STATUS%type)
as
  current_status RESERVATION.STATUS%type;
  d trip.Trip_DATE%type;
begin
  if not F_RESERVATION_EXISTS(r_id) then
    raise_application_error(-20001,'trip not found');
  end if;
  if not st in ('c','n','p') then
    raise_application_error(-20001,'invalid status');
  end if;

  select Trip_DATE into d from RESERVATIONS where RESERVATION_ID = r_id;
  if d <= SYSDATE then
    raise_application_error(-20001,'It is too late to change the status.');
```

```
end if;

  select STATUS into current_status from RESERVATION where RESERVATION_ID = r_id;
  if current_status = st then
    raise_application_error(-20001,'current status is the same as given');
```

```
end if;

  update RESERVATION
  set STATUS = st
  where RESERVATION_ID = r_id;
end;
```


Wywołanie dla wycieczki, na którą nie ma dostępnych miejsc.
Status przed wywołaniem to 'c'.

```
BD_MSKOWRON> begin
    MODIFY_RESERVATION_STATUS(5, 'n');
end;
[2022-03-16 21:51:45] [72000][20001]
[2022-03-16 21:51:45] ORA-20001: Not enough places
[2022-03-16 21:51:45] ORA-06512: przy "BD_MSKOWRON.BU_RESERVATION", linia 15
[2022-03-16 21:51:45] ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_MSKOWRON.BU_RESERVATION'
[2022-03-16 21:51:45] ORA-06512: przy "BD_MSKOWRON.MODIFY_RESERVATION_STATUS", linia 23
[2022-03-16 21:51:45] ORA-06512: przy linia 2
[2022-03-16 21:51:45] Position: 0
```

Wcześniejszy status był 'n'.

```
BD_MSKOWRON> begin
    MODIFY_RESERVATION_STATUS(7, 'c');
end;
[2022-03-16 21:54:17] completed in 1 s 472 ms
```

c. **modify_reservation** -procedura służąca do modyfikacji liczbę miejsc dla rezerwacji.

```
create or replace procedure modify_reservation(r_id int,n_places int)
as
  d trip.TRIP_DATE%type;
begin
  if not F_RESERVATION_EXISTS(r_id) then
    raise_application_error(-20001,'reservation not found');
  end if;
  if n_places < 0 then
    raise_application_error(-20001,'number of places cannot be < 0 ');
  end if;

  select TRIP_DATE into d from RESERVATIONS where RESERVATION_ID = r_id;
  if d <= SYSDATE then
    raise_application_error(-20001,'It is too late to modify.');
```

```
    end if;

    update RESERVATION
      set NO_PLACES = n_places
      where RESERVATION_ID = r_id;
end;
```

Poprzednia rezerwacja była na 1 osobę, na wycieczkę były dostępne jeszcze 2 miejsca.

```
BD_MSKOWRON> begin
                MODIFY_RESERVATION(13,3);
            end;
[2022-03-16 21:57:09] completed in 133 ms
```

Poprzednia rezerwacja była na 1 osobę, na wycieczkę były dostępne jeszcze 2 miejsca.

```
BD_MSKOWRON> begin
                MODIFY_RESERVATION(13,5);
            end;
[2022-03-16 21:58:17] [72000][20001]
[2022-03-16 21:58:17]   ORA-20001: Not enough slots
[2022-03-16 21:58:17]   ORA-06512: przy "BD_MSKOWRON.BU_RESERVATION", linia 25
[2022-03-16 21:58:17]   ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_MSKOWRON.BU_RESERVATION'
[2022-03-16 21:58:17]   ORA-06512: przy "BD_MSKOWRON.MODIFY_RESERVATION", linia 17
[2022-03-16 21:58:17]   ORA-06512: przy linia 2
[2022-03-16 21:58:17] Position: 0
```

d. modify_max_no_places - procedura służąca do modyfikacji maksymalnej liczby miejsc dla danej wycieczki.

```
create or replace procedure modify_max_no_places(t_id int,n_places int)
as
    d trip.TRIP_DATE%type;
begin
    if not F_TRIP_EXISTS(t_id) then
        raise_application_error(-20001,'trip not found');
    end if;
    if n_places < 0 then
        raise_application_error(-20001,'number of places cannot be < 0 ');
    end if;
    select TRIP_DATE into d from TRIP where TRIP_ID = t_id;
    if d <= SYSDATE then
        raise_application_error(-20001,'It is too late to modify max number of places.');
```

```
    end if;

    update TRIP
    set MAX_NO_PLACES = n_places
    where TRIP_ID = t_id;
end;
```

Maksymalna liczba miejsc na wycieczkę to 3 i wszystkie są zarezerwowane.
Próbuję zmienić na 2.

```
BD_MSKOWRON> begin
                MODIFY_MAX_NO_PLACES(3,2);
            end;
[2022-03-16 22:04:44] [72000][20001]
[2022-03-16 22:04:44]   ORA-20001: Too many places reserved.
[2022-03-16 22:04:44]   ORA-06512: przy "BD_MSKOWRON.BU_TRIP", linia 13
[2022-03-16 22:04:44]   ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_MSKOWRON.BU_TRIP'
[2022-03-16 22:04:44]   ORA-06512: przy "BD_MSKOWRON.MODIFY_MAX_NO_PLACES", linia 16
[2022-03-16 22:04:44]   ORA-06512: przy linia 2
[2022-03-16 22:04:44] Position: 0
```

Maksymalna liczba miejsc na wycieczkę to 3, zarezerwowane jest tylko 1 miejsce.
Próbuję zmienić maksymalną liczbę miejsc na 1.

```
BD_MSKOWRON> begin
                MODIFY_MAX_NO_PLACES(5,1);
            end;
[2022-03-16 22:06:27] completed in 141 ms
```

5. TRIGGERY

- a. aiu_reservation** - wywoływany przed insertem na tabeli *RESERVATION*. Służy do dziennikowania zmian statusów rezerwacji, oraz liczby zarezerwowanych miejsc

```
create or replace trigger AIU_RESERVATION
after insert or update
on RESERVATION
for each row
declare
begin
    insert into log(reservation_id, log_date, status, no_places)
    values (:NEW.RESERVATION_ID, SYSDATE, :NEW.STATUS, :NEW.NO_PLACES);
end;
```

- b. bi_reservation** - wywoływany przed insertem na tabeli *RESERVATION*. Obsługuje kontrolę dostępności miejsc.

```
create or replace trigger BI_RESERVATION
before insert
on RESERVATION
FOR EACH ROW

declare
    no_places int;
begin
    select NO_AVAILABLE_PLACES into no_places from AVAILABLE_TRIPS where TRIP_ID = :NEW.TRIP_ID;
    if :NEW.NO_PLACES > no_places then
        raise_application_error(-20001, 'Not enough place.');
```

c. bu_trip - wywoływany przed update'em na tabeli *TRIP*. Obsługuje kontrolę dostępności miejsc.

```
create or replace trigger BU_TRIP
before update
on TRIP
for each row
declare
    PRAGMA AUTONOMOUS_TRANSACTION;
    current_n_places int;
    no_a_places int;
begin
    select MAX_NO_PLACES, NO_AVAILABLE_PLACES into current_n_places, no_a_places from TRIPS where TRIP_ID =
:NEW.TRIP_ID;
    if :NEW.MAX_NO_PLACES >= current_n_places then
        commit ;
    else
        if (current_n_places - :NEW.MAX_NO_PLACES) <= no_a_places then
            commit ;
        else
            raise_application_error(-20001, 'Too many places reserved. ');
        end if;
    end if;
end;
```

d. bu_reservation - wywoływany przed update'em na tabeli *RESERVATION*. Obsługuje kontrolę statusów oraz dostępności miejsc.

```
CREATE OR REPLACE TRIGGER BU_RESERVATION
BEFORE UPDATE
ON RESERVATION
FOR EACH ROW
DECLARE
    PRAGMA AUTONOMOUS_TRANSACTION;
    t_noa_places int;
BEGIN
    -- MODYFIKACJI ULEGL STATUS
    if :NEW.STATUS != :OLD.STATUS then
        if :NEW.STATUS = 'c' then
            COMMIT;
        else
            if :OLD.STATUS = 'c' then
                select NO_AVAILABLE_PLACES into t_noa_places from TRIPS where TRIP_ID = :NEW.TRIP_ID;
                if t_noa_places >= :OLD.NO_PLACES then
                    COMMIT;
                else
                    raise_application_error(-20001, 'Not enough places');
                end if;
            else
                COMMIT;
            end if;
        end if;
    end if;
    -- MODYFIKACJI ULEGLA LICZBA MIEJSC
    if :NEW.NO_PLACES != :OLD.NO_PLACES then
        if (:new.no_places - :old.no_places) > F_GET_FREE_PLACES_TRIP(:old.TRIP_ID) then
            raise_application_error(-20001, 'Not enough slots');
        end if;
        COMMIT;
    end if;
    COMMIT;
end;
```


6. TYPY

```
create or replace type trip_participant as OBJECT
(
    country varchar2(50),
    trip_date date,
    trip_name varchar2(100),
    reservation_id int,
    firstname varchar2(50),
    lastname varchar2(50)
);
create or replace type trip_participant_table is table of trip_participant;
```

```
create or replace type person_reservation as OBJECT
(
    country varchar2(50),
    trip_date date,
    trip_name varchar2(100),
    reservation_id int
);
create or replace type person_reservation_table is table of person_reservation;
```

```
create or replace type available_trip as OBJECT
(
    trip_date date,
    trip_name varchar2(100),
    no_available_places int
);
create or replace type available_trip_table is table of available_trip;
```

