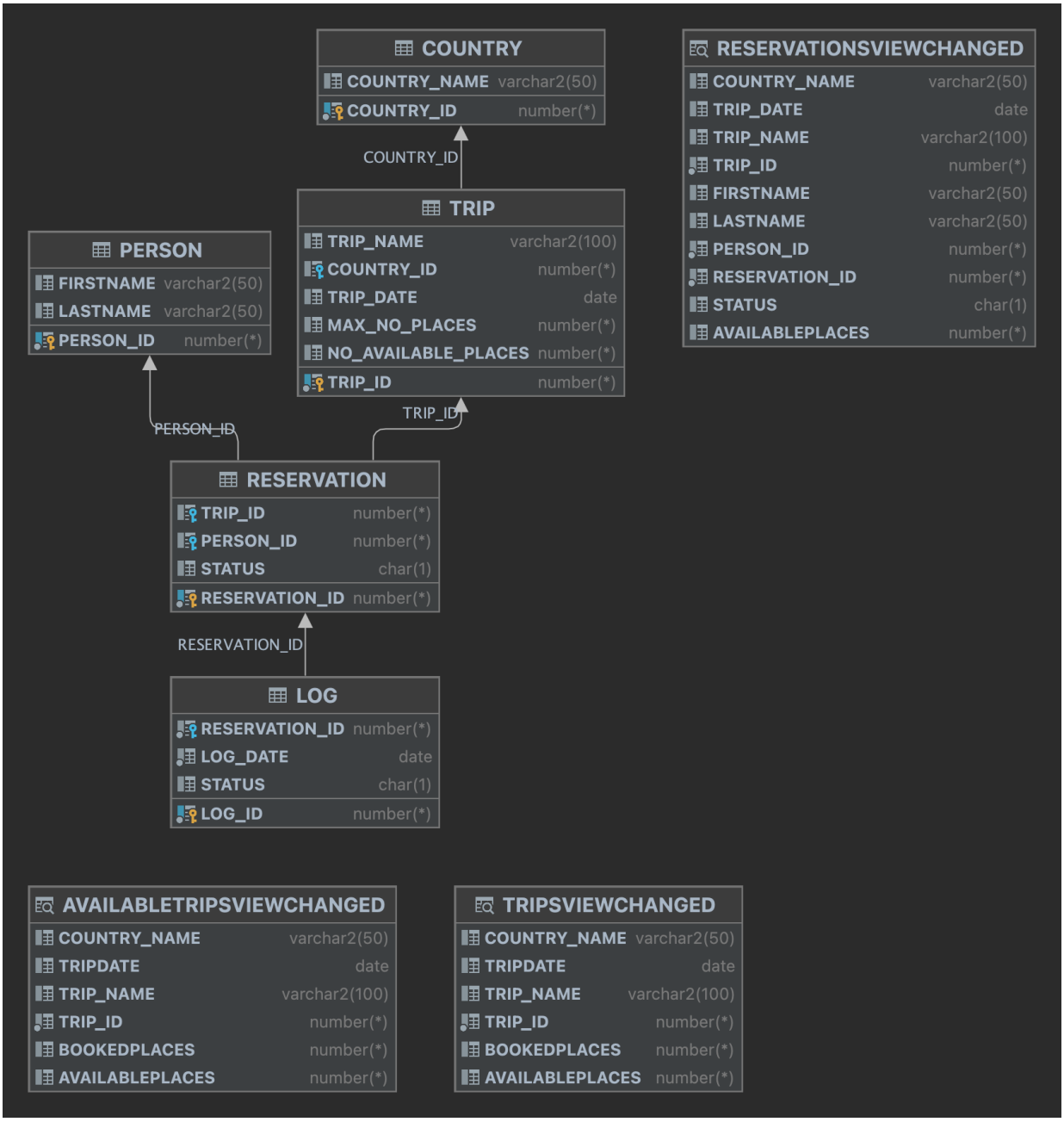
Julia Smerdel

Raport 1

Spis treści

Stworzona baza danych dostępna jest pod nazwą **BD_409614**

Schemat bazy danych


COUNTRY
COUNTRY_NAME varchar2(50)
COUNTRY_ID number(*)
COUNTRY_ID
RESERVATIONSVIEWCHANGED
COUNTRY_NAME varchar2(50)
TRIP_DATE date
TRIP_NAME varchar2(100)
TRIP_ID number(*)
FIRSTNAME varchar2(50)
LASTNAME varchar2(50)
PERSON_ID number(*)
RESERVATION_ID number(*)
STATUS char(1)
AVAILABLEPLACES number(*)
TRIP
TRIP_NAME varchar2(100)
COUNTRY_ID number(*)
TRIP_DATE date
MAX_NO_PLACES number(*)
NO_AVAILABLE_PLACES number(*)
TRIP_ID number(*)
PERSON
FIRSTNAME varchar2(50)
LASTNAME varchar2(50)
PERSON_ID number(*)
PERSON_ID
TRIP_ID
RESERVATION
TRIP_ID number(*)
PERSON_ID number(*)
STATUS char(1)
RESERVATION_ID number(*)
RESERVATION_ID
LOG
RESERVATION_ID number(*)
LOG_DATE date
STATUS char(1)
LOG_ID number(*)
AVAILABLETRIPSVIEWCHANGED
COUNTRY_NAME varchar2(50)
TRIPDATE date
TRIP_NAME varchar2(100)
TRIP_ID number(*)
BOOKEDPLACES number(*)
AVAILABLEPLACES number(*)
TRIPSVIEWCHANGED
COUNTRY_NAME varchar2(50)
TRIPDATE date
TRIP_NAME varchar2(100)
TRIP_ID number(*)
BOOKEDPLACES number(*)
AVAILABLEPLACES number(*)

Tabele

Tabela Person

Reprezentacja osób

Klucz główny: person_id

Imię: firstname

Nazwisko: lastname

```
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
);
```

Tabela Trip

Reprezentacja wycieczek

Klucz główny: trip_id

Nazwa wycieczki: trip_name

ID kraju wycieczki: country_id

Data wycieczki: trip_date

Maksymalna liczba miejsc: max_no_places

```
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
);
```

Później dodano również redundantne pole **no_available_places** reprezentujące liczbę dostępnych miejsc.

```
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,
  no_available_places int,
  constraint trip_pk primary key ( trip_id ) enable
);
```

Tabela Reservation

Reprezentacja rezerwacji

Klucz główny: reservation_id

ID wycieczki: trip_id

ID osoby: person_id

Status: status

```
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
);
```

Tabela Country

Reprezentacja krajów

Klucz główny: country_id

Nazwa kraju: country_name

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

Tabela Log

Reprezentacja zmian

Klucz główny: log_id

ID rezerwacji: reservation_id

Data zmiany: log_date

Status: status

```
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
);
```

Warunki integralnościowe

Trips

```
alter table trip
add constraint trip_fk foreign key
(country_id) references country(country_id) enable;
```

Log

```
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;
```

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;
```

Wstawianie danych do bazy

```
--country
insert into country(country_name)
values ('Polska');

insert into country(country_name)
values ('Francja');

insert into country(country_name)
values ('Czechy');
```

```
-- trip

insert into trip(trip_name, country_id, trip_date, max_no_places,
no_available_places)
values ('Ahoj przygodo', 1, to_date('2022-06-10','YYYY-MM-DD'), 10, 8);

insert into trip(trip_name, country_id, trip_date,  max_no_places,
no_available_places)
values (N'Łoscypek', 1, to_date('2023-07-03','YYYY-MM-DD'), 4, 4);

insert into trip(trip_name, country_id, trip_date,  max_no_places,
no_available_places)
values (N'Żelipapą', 2, to_date('2023-05-01','YYYY-MM-DD'), 5, 4);

insert into trip(trip_name, country_id, trip_date,  max_no_places,
no_available_places)
values ('Krecik puka w taborecik', 3, to_date('2023-05-01','YYYY-MM-DD'), 2,
2);
```

```
-- person

insert into person(firstname, lastname)
values (N'Paweł', N'Gaweł');

insert into person(firstname, lastname)
values ('Joanna', N'Bułeczka');

insert into person(firstname, lastname)
values ('Katarzyna', 'Pierzyna');

insert into person(firstname, lastname)
values ('Adam', 'Padam');

insert into person(firstname, lastname)
values  ('Jakub', N'Burak');

insert into person(firstname, lastname)
values ('Piotr', N'Łotr');

insert into person(firstname, lastname)
values ('Anna', 'Wanna');
```

```
insert into person(firstname, lastname)
values ('Helena', 'Polena');

insert into person(firstname, lastname)
values ('Dominik', N'Chałka');

insert into person(firstname, lastname)
values ('Szymon', 'Dywan');
```

```
-- reservation
-- trip 1
insert into reservation(trip_id, person_id, status)
values (1, 1, 'P');

insert into reservation(trip_id, person_id, status)
values (1, 2, 'N');

-- trip 2
insert into reservation(trip_id, person_id, status)
values (2, 1, 'P');

insert into reservation(trip_id, person_id, status)
values (2, 4, 'C');

-- trip 3
insert into reservation(trip_id, person_id, status)
values (3, 9, 'P');
```

Widoki

Widok Reservations

Wyświetla wszystkie rezerwacje

```
create view Reservations
as
    select c.country_name, t.trip_date, t.trip_name, t.trip_id,
           p.firstname, p.lastname, p.person_id, r.reservation_id,
           r.status, GETLEFTPLACES(t.trip_id) as AvailablePlaces

    from reservation r inner join trip t on r.trip_id = t.trip_id
        inner join country c on t.country_id = c.country_id
        inner join person p on r.person_id = p.person_id;
commit;
```

Zmiana po dodaniu redundantnego pola w tabeli trips

```
create view ReservationsViewChanged
as
    select c.country_name, t.trip_date, t.trip_name, t.trip_id,
           p.firstname, p.lastname, p.person_id, r.reservation_id,
           r.status, t.no_available_places as AvailablePlaces

    from reservation r inner join trip t on r.trip_id = t.trip_id
        inner join country c on t.country_id = c.country_id
        inner join person p on r.person_id = p.person_id;
commit;
```

Widok Trips

Wyświetla wycieczki

```
create view Trips
as
    select c.country_name, t.trip_date as tripDate, t.trip_name, t.trip_id,
           getBookedPlaces(t.trip_id) as bookedPlaces,
           getLeftPlaces(t.trip_id) as availablePlaces
    from country c inner join trip t on c.country_id = t.country_id;
commit;
```

Zmiana po dodaniu redundantnego pola w tabeli trips

```
create view TripsViewChanged
as
    select c.country_name, t.trip_date as tripDate, t.trip_name, t.trip_id,
           t.max_no_places - t.no_available_places as bookedPlaces,
           t.no_available_places as availablePlaces
    from country c inner join trip t on c.country_id = t.country_id;
commit;
```

Widok AvailableTrip

Wyświetla dostępne wycieczki

```
create view AvailableTrips
as
    select country_name, tripDate, trip_name, trip_id, bookedPlaces,
availablePlaces from Trips
    where availablePlaces > 0 and tripDate > SYSDATE;
commit;
```

Zmiana po dodaniu redundantnego pola w tabeli trips

```
create view AvailableTripsViewChanged
as
    select country_name, tripDate, trip_name, trip_id, bookedPlaces,
availablePlaces from TripsViewChanged
    where availablePlaces > 0 and tripDate > SYSDATE;
commit;
```

Procedury

Procedura AddReservation

Dodaje rezerwacje

```
create or replace procedure AddReservation(tripID int, personID int)
as
    availablePlaces int;
begin
    if not personExistence(personID) then
        raise_application_error(-20001, 'Person does not exist');
    end if;
    if not tripExistence(tripID) then
        raise_application_error(-20001, 'Trip does not exist');
    end if;
    if not isTheDateCorrect(tripID) then
        raise_application_error(-20001, 'The trip has already taken place');
    end if;


    availablePlaces := getLeftPlaces(tripID);

    if availablePlaces = 0 then
        raise_application_error(-20001, 'There are no spots for this trip');
    end if;



    insert into Reservation(trip_id, person_id, status)
    values (tripID, personID, 'N');
end;
commit;
```

Modyfikacja – dodanie tabeli Log

```
create or replace procedure AddReservationMod6(tripID int, personID int)
as
    availablePlaces int;
    reservationID int;
begin
    if not personExistence(personID) then
        raise_application_error(-20001, 'Person does not exist');
    end if;
    if not tripExistence(tripID) then
        raise_application_error(-20001, 'Trip does not exist');
    end if;
    if not isTheDateCorrect(tripID) then
        raise_application_error(-20001, 'The trip has already taken place');
    end if;


    availablePlaces := getLeftPlaces(tripID);

    if availablePlaces = 0 then
        raise_application_error(-20001, 'There are no spots for this trip');
    end if;



    insert into Reservation(trip_id, person_id, status)
    values (tripID, personID, 'N');

    select reservation_id into reservationID from Reservation
    where trip_id = tripID and personID = person_id and status = 'N';

    insert into Log(reservation_id, log_date, status) values
    (reservationID, sysdate, 'N');
end;
commit;
```

Modyfikacja – użycie triggerów

```
create or replace procedure AddReservationAfterTriggerMod(tripID int,
personID int)
as
begin
    insert into Reservation(trip_id, person_id, status)
    values (tripID, personID, 'N');
end;
commit;
```

Modyfikacja – dodanie pola no_available_places

```
create or replace procedure AddReservationChanged(tripID int, personID int)
as
begin

    insert into Reservation(trip_id, person_id, status)
    values (tripID, personID, 'N');

--      update Trip set no_available_places = getAvailablePlaces(tripID)- 1
--      where trip_id = tripID;
end;
commit;
```

Procedura ModifyReservationStatus

Modyfikuje status rezerwacji

```
create or replace procedure ModifyReservationStatus(reservationID int,
newStatus char)
as
    availablePlaces int;
    tripID int;
    currStatus char;
begin

    select trip_id, status into tripID, currStatus from Reservation
    where reservationID=reservation_id;

    availablePlaces := getLeftPlaces(tripID);

    if not reservationExistence(reservationID) then
        raise_application_error(-20001, 'Reservation does not exist');
    end if;
    if not tripExistence(tripID) then
        raise_application_error(-20001, 'Trip does not exist');
    end if;


    case newStatus
        when currStatus then
        raise_application_error(-20001,'Reservation already has this
status');

        when 'N' then
        null;

        when 'C' then
        null;

        when 'P' then
            if currStatus = 'C' then
                if availablePlaces = 0 then
                    raise_application_error(-20001, 'Not enough places
available');
                end if;
            end if;

        else
        raise_application_error(-20001, 'Wrong status given');
    end case;


    update Reservation set status=newStatus where
reservation_id=reservationID;

end;
commit;
```

Modyfikacja – dodanie tabeli log

```
create or replace procedure ModifyReservationStatusMod6(reservationID int,
newStatus char)
as
    availablePlaces int;
    tripID int;
    currStatus char;
begin

    select trip_id, status into tripID, currStatus from Reservation
    where reservationID=reservation_id;

    availablePlaces := getLeftPlaces(tripID);

    if not reservationExistence(reservationID) then
        raise_application_error(-20001, 'Reservation does not exist');
    end if;
    if not tripExistence(tripID) then
        raise_application_error(-20001, 'Trip does not exist');
    end if;


    case newStatus
        when currStatus then
        raise_application_error(-20001,'Reservation already has this
status');

        when 'N' then
        null;

        when 'C' then
        null;

        when 'P' then
            if currStatus = 'C' then
                if availablePlaces <= 0 then
                    raise_application_error(-20001, 'Not enough places
available');
                end if;
            end if;

        else
        raise_application_error(-20001, 'Wrong status given');
    end case;


    update Reservation set status=newStatus where
reservation_id=reservationID;

    insert into Log(reservation_id, log_date, status) values
    (reservationID, sysdate, newStatus);

end;
commit;
```

Modyfikacja – użycie triggerów

```
create or replace procedure
ModifyReservationStatusAfterTriggerMod(reservationID int, newStatus char)
as
begin

    if not reservationExistence(reservationID) then
        raise_application_error(-20001, 'Reservation does not exist');
    end if;

    update Reservation set status=newStatus where
reservation_id=reservationID;

end;
commit;
```

Modyfikacja – dodanie pola no_available_places

```
create or replace procedure ModifyNoPlacesChanged(tripID int, noPlaces int)
as
begin
    update Trip set max_no_places = noPlaces where tripID = trip_id;
    begin
        countAvailablePlaces(tripID);
    end;
end;

commit;
```

Procedura ModifyNoPlaces

Modyfikuje maksymalną liczbę miejsc wycieczki

```
create or replace procedure ModifyNoPlaces(tripID int, noPlaces int)
as
    currPlaces int;
    availablePlaces int;
begin
    select max_no_places into currPlaces from Trip where tripID = trip_id;
    availablePlaces := getLeftPlaces(tripID);

    if not tripExistence(tripID) then
        raise_application_error(-20001, 'Trip with this id does not exist');
    end if;

    if currPlaces - availablePlaces > noPlaces then
        raise_application_error(-20001, 'The amount of booked places is
greater than new max_no_places value.');
    end if;


    update Trip set max_no_places = noPlaces where tripID = trip_id;
end;
commit;
```

Modyfikacja – dodanie pola no_available_places

```
create or replace procedure ModifyNoPlacesChanged(tripID int, noPlaces int)
as
begin
    update Trip set max_no_places = noPlaces where tripID = trip_id;
    begin
        countAvailablePlaces(tripID);
    end;
end;
```

Procedura countAvailablePlaces

Zmienia dostępną liczbę miejsc (użyto podczas dodania redundantnego pola no_available_places do tabeli Trip)

```
create or replace procedure countAvailablePlaces(tripID int)
as
    result int;
begin
    select max_no_places - getBookedPlaces(tripID) into result from Trip
where tripID = trip_id;

    update Trip set no_available_places = result where trip_id=tripID;

end;
commit;
```

Funkcje

Funkcje skalarne

Funkcja getBookedPlaces

Funkcja do obliczania liczby zabookowanych miejsc

```
create or replace function getBookedPlaces(tripID int)
    return int
as
    result int;
begin
    select count(r.status) into result
    from reservation r
    where r.trip_id = tripID and r.status != 'C';

    return result;
end;
commit;
```

Funkcja getLeftPlaces

Funkcja do obliczania liczby dostępnych miejsc

```
create or replace function getLeftPlaces(tripID int)
    return int
as
    result int;
begin
    select t.max_no_places - getBookedPlaces(tripID) into result
    from trip t where tripID = t.trip_id;

    return result;
end;
commit;
```

Modyfikacja – dodanie pola no_available_places

```
create or replace function getAvailablePlaces(tripID int)
    return int
as
    result int;
begin
    select no_available_places into result from Trip where tripID=trip_id;

    return result;
end;
commit;
```

Funkcja tripExistence

Funkcja sprawdza, czy wycieczka istnieje

```
create or replace function tripExistence(tripID int)
    return boolean
as
    exist number;
begin
    select case
        when exists(select * from trip where trip.trip_id = tripID) then 1
        else 0
    end
    into exist from dual;

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

Funkcja personExistence

Funkcja sprawdza, czy osoba istnieje

```
create or replace function personExistence(personID int)
    return boolean
as
    exist number;
begin
    select case
        when exists(select * from person where person.person_id = personID)
then 1
        else 0
    end
    into exist from dual;

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

Funkcja reservationExistence

Funkcja sprawdza, czy rezerwacja istnieje

```
create or replace function reservationExistence(reservationID int)
    return boolean
as
    exist number;
begin
    select case
        when exists(select * from reservation where
reservation.reservation_id = reservationID) then 1
        else 0
    end
    into exist from dual;

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

Funkcja isDateCorrect

Funkcja sprawdza czy data wycieczki jest poprawna – czy już się odbyła

```
-- wyświetla klientów, którzy zamówili więcej niż x zamówień za minimalną kwotę y
create or replace function isTheDateCorrect(tripID int, givenDate date
default SYSDATE)
    return boolean
as
    result date;
begin
    if not tripExistence(tripID) then
        raise_application_error(-2000, 'The trip does not exist');
    end if;

    select trip_date into result from trip where trip_id = tripID;

    if givenDate >= result then --false when the trip has taken place
earlier
        return false;
    else
        return true;
    end if;
end;
commit;
```

Funkcje zwracające tabele

Funkcja TripParticipants

Funkcja pokazuje uczestników wycieczki

```
create or replace function TripParticipants(tripID int)
    return TripParticipantsTable
as
    result TripParticipantsTable;
begin
    if not tripExistence(tripID) then
        raise_application_error(-2000, 'There are no such trip in the
database');
    end if;

    select TripParticipant(firstname, lastname, person_id, trip_name,
trip_date)
        bulk collect into result
    from Reservations
    where tripID = trip_id;

    return result;
end;
commit;
```

Modyfikacja – dodanie pola no_available_places

```
create or replace function TripParticipantsChanged(tripID int)
    return TripParticipantsTable
as
    result TripParticipantsTable;
begin
    if not tripExistence(tripID) then
        raise_application_error(-2000, 'There are no such trip in the
database');
    end if;

    select TripParticipant(firstname, lastname, person_id, trip_name,
trip_date)
        bulk collect into result
    from ReservationsViewChanged
    where tripID = trip_id;

    return result;
end;
commit;
```

Funkcja PersonReservations

Funkcja wyświetla rezerwacje danej osoby

```
create or replace function PersonReservations(personID int)
    return PersonReservationsTable
as
    result PersonReservationsTable;
begin
    if not personExistence(personID) then
        raise_application_error(-2000, 'There are no such person in the
database');
    end if;

    select PersonReservationsO(trip_name, trip_date, trip_id, country_name,
reservation_id, status)
        bulk collect into result
    from Reservations
    where personID=person_id;

    return result;
end;
commit;
```

Modyfikacja – dodanie pola no_available_places

```
create or replace function PersonReservationsChanged(personID int)
    return PersonReservationsTable
as
    result PersonReservationsTable;
begin
    if not personExistence(personID) then
        raise_application_error(-2000, 'There are no such person in the
database');
    end if;

    select PersonReservationsO(trip_name, trip_date, trip_id, country_name,
reservation_id, status)
        bulk collect into result
    from ReservationsViewChanged
    where personID=person_id;

    return result;
end;
commit;
```

Funkcja AvailableTripsTo

Funkcja wyświetla dostępne wycieczki (w podanym terminie do podanego kraju)

```
create or replace function AvailableTripsTo(countryName varchar, dateFrom
date, dateTo date)
    return AvailableTripsTable
as
    result AvailableTripsTable;
begin
    if dateFrom > dateTo then raise_application_error(-2000, 'Wrong date
given');
    end if;

    if dateFrom < sysdate then raise_application_error(-2000, 'Wrong date
given');
    end if;

    select AvailableTripsO(trip_id, trip_name, country_name, tripDate,
availablePlaces)
    bulk collect into result
    from AvailableTrips
    where country_name = countryName and tripDate between dateFrom and
dateTo;

    return result;
end;
commit;
```

Modyfikacja – dodanie pola no_available_places

```
create or replace function AvailableTripsToChanged(countryName varchar,
dateFrom date, dateTo date)
    return AvailableTripsTable
as
    result AvailableTripsTable;
begin
    if dateFrom > dateTo then raise_application_error(-2000, 'Wrong date
given');
    end if;

    if dateFrom < sysdate then raise_application_error(-2000, 'Wrong date
given');
    end if;

    select AvailableTripsO(trip_id, trip_name, country_name, tripDate,
availablePlaces)
    bulk collect into result
    from AvailableTripsViewChanged
    where country_name = countryName and tripDate between dateFrom and
dateTo;

    return result;
end;
commit;
```

Triggery

Trigger AddReservationTrigger

Trigger aktualizuje dane w tabeli Log po dodaniu nowej rezerwacji

```
create or replace trigger AddReservationTrigger
    after insert on Reservation
    for each row
    begin
        insert into Log(reservation_id, log_date, status)
        values (:new.reservation_id, sysdate, :new.status);
    end;
commit;
```

Modyfikacja po dodaniu pola no_available_places

```
create or replace trigger AddReservationTrigger
    after insert on Reservation
    for each row
    begin
        insert into Log(reservation_id, log_date, status)
        values (:new.reservation_id, sysdate, :new.status);

        update Trip set no_available_places = no_available_places - 1
        where trip_id = :new.trip_id;

    end;
commit;
```

Trigger ModifyReservationStatusTrigger

Trigger aktualizuje dane w tabeli Log po modyfikacji statusu rezerwacji

```
create or replace trigger ModifyReservationStatusTrigger
    after update of status on Reservation
    for each row
    begin
        if :new.status != :old.status then
            insert into Log(reservation_id, log_date, status)
            values (:new.reservation_id, sysdate, :new.status);
        end if;
    end;
commit;
```

Modyfikacja po dodaniu pola no_available_places

```
create or replace trigger ModifyReservationStatusTrigger
    after update of status on Reservation
    for each row
    begin
        if :new.status != :old.status then
            insert into Log(reservation_id, log_date, status)
            values (:new.reservation_id, sysdate, :new.status);
        end if;

        if :new.status = 'C' then
            update Trip set no_available_places = no_available_places + 1
        where trip_id = :new.trip_id;
        end if;

    end;
commit;
```

Trigger BeforeAddingReservationTrigger

Trigger sprawdza, czy rezerwacja jest możliwa

```
create or replace trigger BeforeAddingReservationTrigger
    before insert on Reservation
    for each row
    declare availablePlaces int;
    begin
        if not personExistence(:NEW.person_id) then
            raise_application_error(-20001, 'Person does not exist');
        end if;
        if not tripExistence(:new.trip_id) then
            raise_application_error(-20001, 'Trip does not exist');
        end if;
        if not isTheDateCorrect(:new.trip_id) then
            raise_application_error(-20001, 'The trip has already taken
place');
        end if;

        availablePlaces := getLeftPlaces(:new.trip_id);

        if availablePlaces = 0 then
            raise_application_error(-20001, 'There are no spots for this
trip');
        end if;
    end;
commit;
```

Modyfikacja po dodaniu pola no_available_places

```
create or replace trigger BeforeAddingReservationTrigger
    before insert on Reservation
    for each row
    declare availablePlaces int;
    begin
        if not personExistence(:NEW.person_id) then
            raise_application_error(-20001, 'Person does not exist');
        end if;
        if not tripExistence(:new.trip_id) then
            raise_application_error(-20001, 'Trip does not exist');
        end if;
        if not isTheDateCorrect(:new.trip_id) then
            raise_application_error(-20001, 'The trip has already taken
place');
        end if;

        availablePlaces := getAvailablePlaces(:new.trip_id);

        if availablePlaces = 0 then
            raise_application_error(-20001, 'There are no spots for this
trip');
        end if;
    end;
commit;
```

Trigger BeforeChangingReservationStatus

Trigger sprawdza, czy zmiana statusu rezerwacji jest możliwa

```
create or replace trigger BeforeChangingReservationStatus
    before update of status on Reservation
    for each row
    declare
        pragma autonomous_transaction;
        availablePlaces int;
    begin
        availablePlaces := getLeftPlaces(:new.trip_id);

        case :new.status
        when :old.status then
        raise_application_error(-20001,'Reservation already has this
status');

        when 'N' then
        null;

        when 'C' then
        null;


        when 'P' then
            if :old.status = 'C' then
                if availablePlaces = 0 then
                    raise_application_error(-20001, 'Not enough places
available');
                end if;
            end if;

        else
        raise_application_error(-20001, 'Wrong status given');
    end case;
    end;
commit;
```

Modyfikacja po dodaniu pola no_available_places

```
create or replace trigger BeforeChangingReservationStatusTrigger
    before update of status on Reservation
    for each row
    declare
        pragma autonomous_transaction;
        availablePlaces int;
    begin
        availablePlaces := getAvailablePlaces(:new.trip_id);

        case :new.status
        when :old.status then
        raise_application_error(-20001,'Reservation already has this
status');

        when 'N' then
        null;

        when 'C' then
        null;

        when 'P' then
            if :old.status = 'C' then
                if availablePlaces = 0 then
                    raise_application_error(-20001, 'Not enough places
available');
                end if;
            end if;

        else
        raise_application_error(-20001, 'Wrong status given');
    end case;
    end;
commit;
```

Trigger BeforeChangingMaxPlacesTrigger

Trigger sprawdza, czy zmiana maksymalnej liczby miejsc jest możliwa

```
create or replace trigger BeforeChangingMaxPlacesTrigger
    before update of max_no_places on Trip
    for each row
    declare
        pragma autonomous_transaction;
        availablePlaces int;
    begin
        availablePlaces := getAvailablePlaces(:new.trip_id);

        if not tripExistence(:new.trip_id) then
            raise_application_error(-20001, 'Trip with this id does not
exist');
        end if;

        if :new.max_no_places <= 0 then
            raise_application_error(-20001, 'Wrong number of mx places
given');
        end if;

        if :old.max_no_places - availablePlaces > :new.max_no_places then
            raise_application_error(-20001, 'The amount of booked places is
greater than new max_no_places value.');
        end if;


    end;
commit;
```

Obiekty

TripParticipant & TripParticipantsTable

```
create or replace type TripParticipant as object(
    firstname varchar(50),
    lastname varchar(50),
    person_id int,
    trip_name varchar(100),
    trip_date date
);

create or replace type TripParticipantsTable is table of TripParticipant;
```

PersonReservations0 & PersonReservationsTable

```
create or replace type PersonReservationsO as object(
    trip_name varchar(100),
    trip_date date,
    trip_id int,
    country_name varchar(50),
    reservation_id int,
    status char(1)
);

create or replace type PersonReservationsTable is table of
PersonReservationsO;
```

AvailableTrips0 & AvailableTripsTable

```
create or replace type AvailableTripsO as object(
    trip_id int,
    trip_name varchar(100),
    country_name varchar(50),
    tripDate date,
    availablePlaces int
);

create or replace type AvailableTripsTable is table of AvailableTripsO;
```

Wyniki

Widok Trips

	COUNTRY_NAME	TRIPDATE	TRIP_NAME	TRIP_ID	BOOKEDPLACES	AVAILABLEPLACES
1	Polska	2022-06-10	Ahoj przygodo	1	2	10
2	Polska	2023-07-03	Łoscypek	2	2	5
3	Francja	2023-05-01	Żelipapą	3	1	4
4	Czechy	2023-05-01	Krecik puka w taborecik	4	1	0

Widok Available Trips

	COUNTRY_NAME	TRIPDATE	TRIP_NAME	TRIP_ID	BOOKEDPLACES	AVAILABLEPLACES
1	Polska	2023-07-03	Łoscypek	2	2	5
2	Francja	2023-05-01	Żelipapą	3	1	4

Próba dodania rezerwacji na wycieczkę z brakiem dostępnych miejsc

```
begin
    AddReservationAfterTriggerMod( tripID: 4,  personID: 7);
end;
commit;
```

```
[72000][20001]
ORA-20001: There are no spots for this trip
ORA-06512: at "BD_409614.BEFOREADDINGRESERVATIONTRIGGER", line 16
ORA-04088: error during execution of trigger 'BD_409614.BEFOREADDINGRESERVATIONTRIGGER'
ORA-06512: at "BD_409614.ADDRESERVATIONAFTERTRIGGERMOD", lin ...
```

Próba dodania rezerwacji na wycieczkę z dostępnymi miejscami

```
begin
    AddReservationAfterTriggerMod( tripID: 2,  personID: 7);
end;
commit;
```

	COUNTRY_NAME	TRIPDATE	TRIP_NAME	TRIP_ID	BOOKEDPLACES	AVAILABLEPLACES
1	Polska	2022-06-10	Ahoj przygodo	1	2	10
2	Polska	2023-07-03	Łoscypek	2	3	4
3	Francja	2023-05-01	Żelipapą	3	1	4
4	Czechy	2023-05-01	Krecik puka w taborecik	4	1	0

(widok Trips)

	LOG_ID	RESERVATION_ID	LOG_DATE	STATUS
1	21	142	2023-04-02 12:25:01	N

(tabela Log)

	TRIP_ID	FIRSTNAME	LASTNAME	PERSON_ID	RESERVATION_ID	STATUS	AVAILABLEPLACES
	1	Paweł	Gaweł	1	83	P	10
	2	Paweł	Gaweł	1	85	P	4
	1	Joanna	Bułeczka	2	84	N	10
	2	Adam	Padam	4	86	C	4
	2	Anna	Wanna	7	142	N	4

(widok ReservationsViewChanged)

Próba zmiany liczby miejsc wycieczki na mniejszą niż liczba zarezerwowanych

```
begin
    ModifyNoPlacesChanged( tripID: 2,  noPlaces: 1);
end;
commit;
```

```
[72000][20001]
ORA-20001: The amount of booked places is greater than new max_no_places value.
ORA-06512: at "BD_409614.BEFORECHANGINGMAXPLACESTRIGGER", line 16
ORA-04088: error during execution of trigger 'BD_409614.BEFORECHANGINGMAXPLACESTRIGGER'
ORA-06512: at "BD_409614.MODIFYNOPLACESCHANGED", lin ...
```

Zmiana statusu rezerwacji

```
begin
    ModifyReservationStatusChanged( reservationID: 142,  newStatus: 'P');
end;
commit;
```

	LOG_ID	RESERVATION_ID	LOG_DATE	STATUS
1	21	142	2023-04-02 12:25:01	N
2	22	142	2023-04-02 12:31:13	P