Laboratorium 1: Oracle PL/SQL

Autor: Krzysztof Solecki

1.Tabele

1.1 Person:

```
create table person
(
  person_id int generated always as identity not null,
  firstname varchar(50),
  lastname varchar(50),
  constraint person_pk primary key ( person_id ) enable
);
```

1.2 Trip:

```
create table trip
(
  trip_id int generated always as identity not null,
  trip_name varchar(100),
  country_id int,
  trip_date date,
  max_no_places int,
  constraint trip_pk primary key ( trip_id ) enable
);
```

1.3 Reservation:

```
create table reservation
(
  reservation_id int generated always as identity not null,
  trip_id int,
  person_id int,
  status char(1),
  constraint reservation_pk primary key ( reservation_id ) enable
);
```

1.4 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
);
```

1.5 Country:

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

1.6 Constrainty:

```
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;

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;
```

```sql
alter table trip
add constraint trip_fk1 foreign key
( COUNTRY_ID ) references country ( country_id ) enable;
```

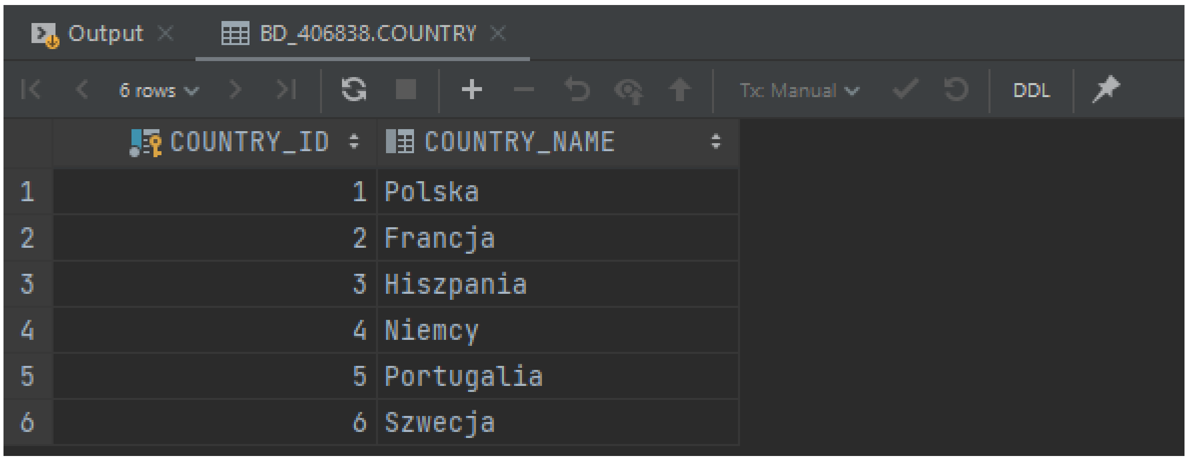
2. Wypełnianie tabele przykładowymi danymi:

```sql
-- trip

insert into trip(trip_name, COUNTRY_ID, trip_date, max_no_places)
values ('Wycieczka do Paryza', 2, to_date('2022-09-12','YYYY-MM-DD'), 3);

insert into trip(trip_name, COUNTRY_ID, trip_date,  max_no_places)
values ('Piękny Kraków', 1, to_date('2023-07-03','YYYY-MM-DD'), 2);

insert into trip(trip_name, country_id, trip_date,  max_no_places)
values ('Znów do Francji', 2, to_date('2023-05-01','YYYY-MM-DD'), 2);

insert into trip(trip_name, country_id, trip_date,  max_no_places)
values ('Hel', 1, to_date('2023-05-01','YYYY-MM-DD'), 2);

-- person

insert into person(firstname, lastname)
values ('Jan', 'Nowak');

insert into person(firstname, lastname)
values ('Jan', 'Kowalski');

insert into person(firstname, lastname)
values ('Jan', 'Nowakowski');

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

insert into person(firstname, lastname)
values  ('Novak', 'Nowak');

insert into person(firstname, lastname)
values ('Maciej', 'Wysoki');

insert into person(firstname, lastname)
values ('Wojciech', 'Niski');
```

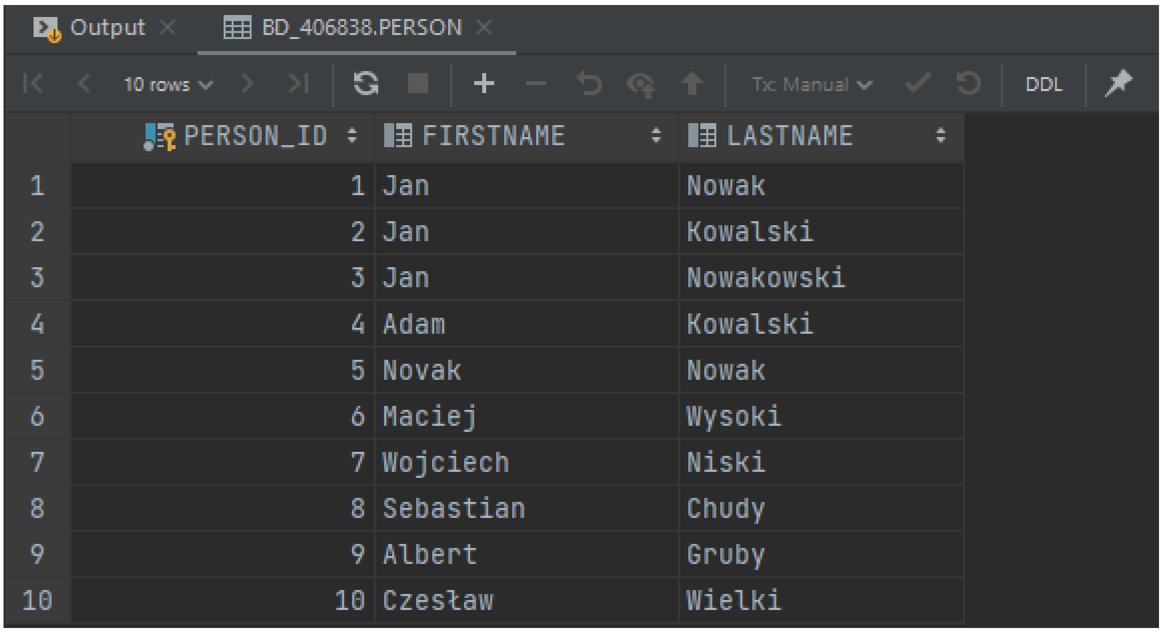
```sql
insert into person(firstname, lastname)
values ('Sebastian', 'Chudy');

insert into person(firstname, lastname)
values ('Albert', 'Gruby');

insert into person(firstname, lastname)
values ('Czesław', 'Wielki');

-- reservation
-- trip 1
insert into reservation(trip_id, person_id, status)
values (6, 1, 'P');

-- trip 2
insert into reservation(trip_id, person_id, status)
values (7, 1, 'P');

-- trip 3
insert into reservation(trip_id, person_id, status)
values (8, 5, 'P');

-- trip 4
insert into reservation(trip_id, person_id, status)
values (8, 1, 'N');

-- trip 5
insert into reservation(trip_id, person_id, status)
values (6, 7, 'N');

-- trip 6
insert into reservation(trip_id, person_id, status)
values (7, 8, 'P');

-- trip 7
insert into reservation(trip_id, person_id, status)
values (8, 9, 'P');

-- trip 8
insert into reservation(trip_id, person_id, status)
values (9, 4, 'P');

-- trip 9
insert into reservation(trip_id, person_id, status)
values (6, 2, 'N');
```

```
-- trip 10
insert into reservation(trip_id, person_id, status)
values (7, 4, 'C');

-- country
INSERT INTO COUNTRY(COUNTRY_NAME)
values ('Polska');

INSERT INTO COUNTRY(COUNTRY_NAME)
values ('Francja');

INSERT INTO COUNTRY(COUNTRY_NAME)
values ('Hiszpania');

INSERT INTO COUNTRY(COUNTRY_NAME)
values ('Niemcy');

INSERT INTO COUNTRY(COUNTRY_NAME)
values ('Portugalia');

INSERT INTO COUNTRY(COUNTRY_NAME)
values ('Szwecja');
```

Po wykonaniu powyższych zapytań tabele: TRIP,RESERVATION,PERSON,COUNTRY zostały wypełnione przykładowymi danymi.

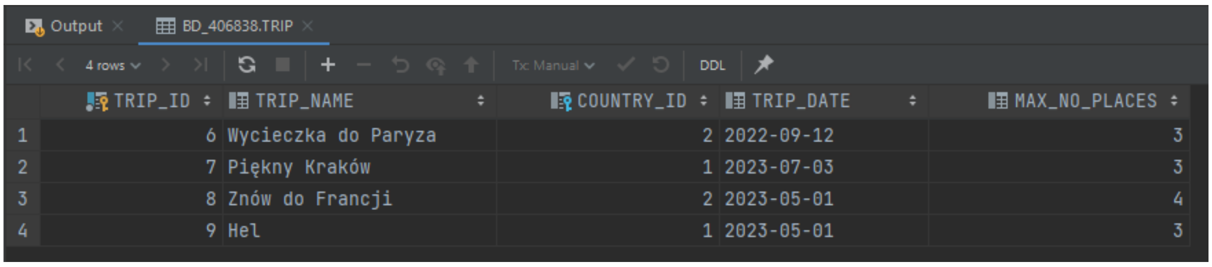


	COUNTRY_ID	COUNTRY_NAME
1	1	Polska
2	2	Francja
3	3	Hiszpania
4	4	Niemcy
5	5	Portugalia
6	6	Szwecja


Output × BD_406838.PERSON ×

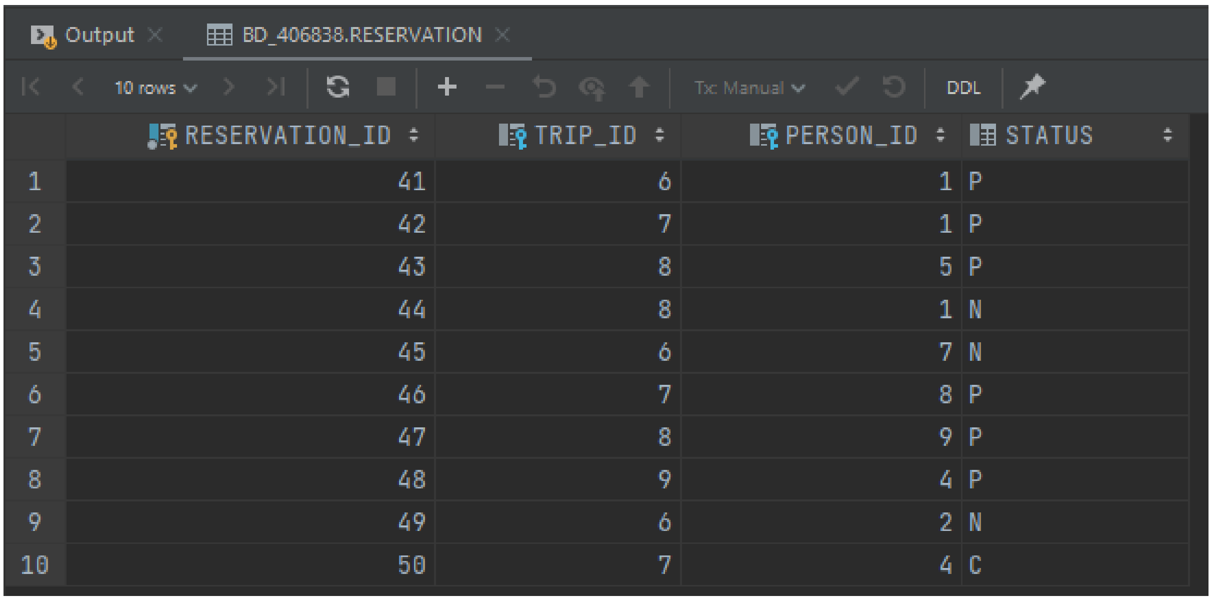
10 rows Tx: Manual DDL

	PERSON_ID	FIRSTNAME	LASTNAME
1	1	Jan	Nowak
2	2	Jan	Kowalski
3	3	Jan	Nowakowski
4	4	Adam	Kowalski
5	5	Novak	Nowak
6	6	Maciej	Wysoki
7	7	Wojciech	Niski
8	8	Sebastian	Chudy
9	9	Albert	Gruby
10	10	Czesław	Wielki


Output × BD_406838.TRIP ×

4 rows Tx: Manual DDL

	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES
1	6	Wycieczka do Paryza	2	2022-09-12	3
2	7	Piękny Kraków	1	2023-07-03	3
3	8	Znów do Francji	2	2023-05-01	4
4	9	Hel	1	2023-05-01	3


Output × BD_406838.RESERVATION ×

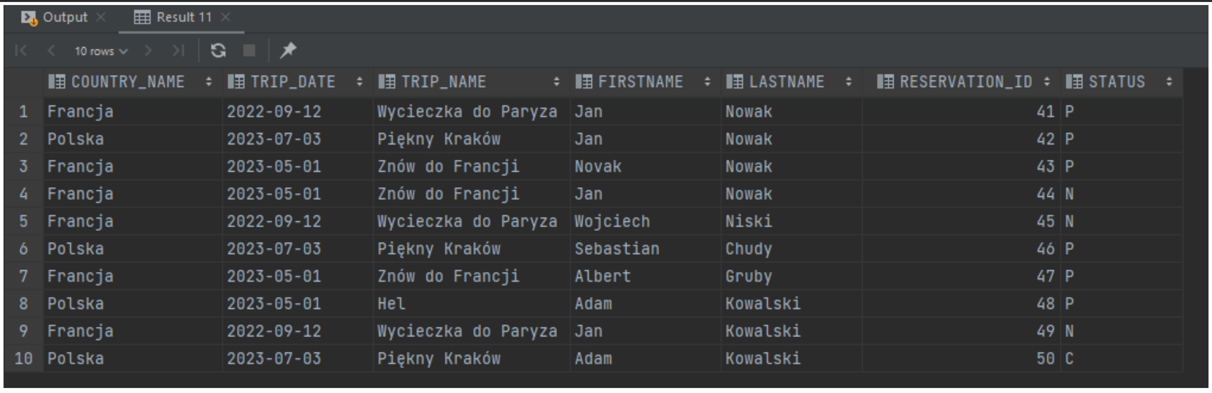
10 rows Tx: Manual DDL

	RESERVATION_ID	TRIP_ID	PERSON_ID	STATUS
1	41	6	1	P
2	42	7	1	P
3	43	8	5	P
4	44	8	1	N
5	45	6	7	N
6	46	7	8	P
7	47	8	9	P
8	48	9	4	P
9	49	6	2	N
10	50	7	4	C

3. Widoki:

3.1 Reservations:

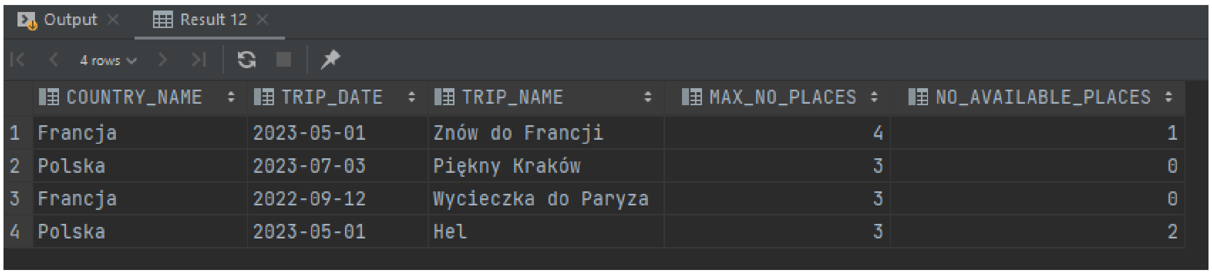
```
CREATE OR REPLACE VIEW RESERVATIONS_VIEW
AS
   SELECT COUNTRY_NAME,
          TRIP_DATE,
          TRIP_NAME,
          FIRSTNAME,
          LASTNAME,
          RESERVATION_ID,
          STATUS
    FROM COUNTRY
    JOIN TRIP T on COUNTRY.COUNTRY_ID = T.COUNTRY_ID
    JOIN RESERVATION R on T.TRIP_ID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



	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	FIRSTNAME	LASTNAME	RESERVATION_ID	STATUS
1	Francja	2022-09-12	Wycieczka do Paryza	Jan	Nowak	41	P
2	Polska	2023-07-03	Piękny Kraków	Jan	Nowak	42	P
3	Francja	2023-05-01	Znów do Francji	Novak	Nowak	43	P
4	Francja	2023-05-01	Znów do Francji	Jan	Nowak	44	N
5	Francja	2022-09-12	Wycieczka do Paryza	Wojciech	Niski	45	N
6	Polska	2023-07-03	Piękny Kraków	Sebastian	Chudy	46	P
7	Francja	2023-05-01	Znów do Francji	Albert	Gruby	47	P
8	Polska	2023-05-01	Hel	Adam	Kowalski	48	P
9	Francja	2022-09-12	Wycieczka do Paryza	Jan	Kowalski	49	N
10	Polska	2023-07-03	Piękny Kraków	Adam	Kowalski	50	C

3.2 Trips

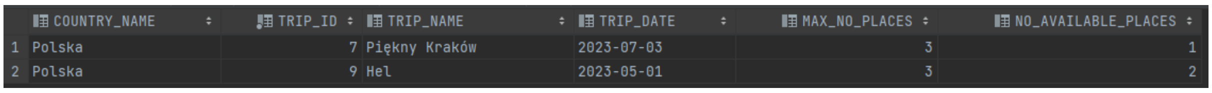
```
CREATE OR REPLACE VIEW TRIPS_VIEW
AS
    SELECT COUNTRY_NAME,
           TRIP_DATE,
           TRIP_NAME,
           MAX_NO_PLACES,
           (MAX_NO_PLACES- COUNT(DISTINCT PERSON_ID)) AS NO_AVAILABLE_PLACES
    FROM COUNTRY
    JOIN TRIP ON COUNTRY.COUNTRY_ID = TRIP.COUNTRY_ID
    LEFT JOIN (SELECT * FROM RESERVATION WHERE STATUS <> 'C') RESERVATION
    ON RESERVATION.TRIP_ID = TRIP.TRIP_ID
    GROUP BY COUNTRY.COUNTRY_NAME,TRIP.TRIP_ID, TRIP.TRIP_NAME, TRIP.TRIP_DATE,
TRIP.MAX_NO_PLACES
```



	COUNTRY_NAME	TRIP_DATE	TRIP_NAME	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	Francja	2023-05-01	Znów do Francji	4	1
2	Polska	2023-07-03	Piękny Kraków	3	0
3	Francja	2022-09-12	Wycieczka do Paryza	3	0
4	Polska	2023-05-01	Hel	3	2

3.3 Available_trips

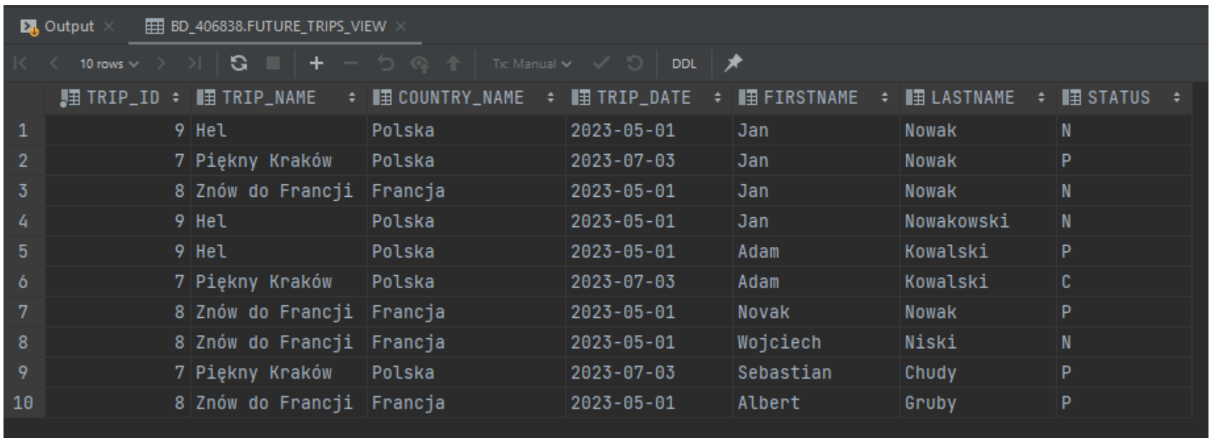
```
CREATE OR REPLACE VIEW AVAILABLE_TRIPS_VIEW
AS
    SELECT COUNTRY_NAME,
           TRIP_NAME,
           TRIP_DATE,
           MAX_NO_PLACES,
           (MAX_NO_PLACES- COUNT(DISTINCT PERSON_ID)) NO_AVAILABLE_PLACES
    FROM COUNTRY
    JOIN TRIP on COUNTRY.COUNTRY_ID = trip.COUNTRY_ID
    LEFT JOIN (SELECT * FROM RESERVATION WHERE STATUS <> 'C') RESERVATION
    ON RESERVATION.TRIP_ID = TRIP.TRIP_ID
    GROUP BY COUNTRY.COUNTRY_NAME,TRIP.TRIP_ID, TRIP.TRIP_NAME, TRIP.TRIP_DATE,
TRIP.MAX_NO_PLACES
    HAVING TRIP_DATE > current_date AND MAX_NO_PLACES- COUNT(DISTINCT PERSON_ID) > 0
```



	COUNTRY_NAME	TRIP_ID	TRIP_NAME	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	Polska	7	Piękny Kraków	2023-07-03	3	1
2	Polska	9	Hel	2023-05-01	3	2

3.4 Future_Trips:

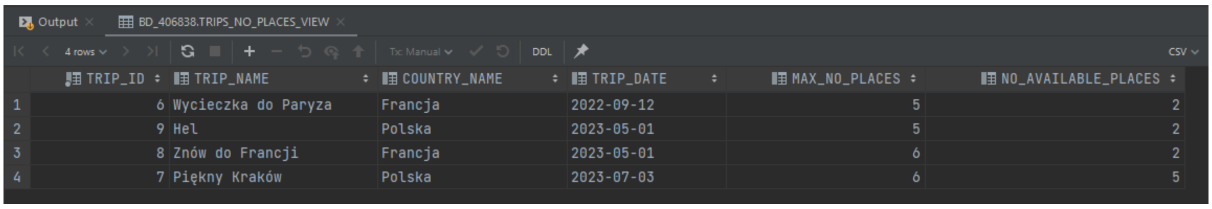
```
CREATE OR REPLACE VIEW FUTURE_TRIPS_VIEW AS
SELECT TRIP.TRIP_ID,
      TRIP_NAME,
      COUNTRY_NAME,
      TRIP_DATE,
      FIRSTNAME,
      LASTNAME,
      STATUS
FROM TRIP
JOIN RESERVATION ON TRIP.TRIP_ID = RESERVATION.TRIP_ID
JOIN COUNTRY ON TRIP.COUNTRY_ID = COUNTRY.COUNTRY_ID
JOIN PERSON ON RESERVATION.PERSON_ID = PERSON.PERSON_ID
WHERE TRIP_DATE > CURRENT_DATE;
```



	TRIP_ID	TRIP_NAME	COUNTRY_NAME	TRIP_DATE	FIRSTNAME	LASTNAME	STATUS
1	9	Hel	Polska	2023-05-01	Jan	Nowak	N
2	7	Piękny Kraków	Polska	2023-07-03	Jan	Nowak	P
3	8	Znów do Francji	Francja	2023-05-01	Jan	Nowak	N
4	9	Hel	Polska	2023-05-01	Jan	Nowakowski	N
5	9	Hel	Polska	2023-05-01	Adam	Kowalski	P
6	7	Piękny Kraków	Polska	2023-07-03	Adam	Kowalski	C
7	8	Znów do Francji	Francja	2023-05-01	Novak	Nowak	P
8	8	Znów do Francji	Francja	2023-05-01	Wojciech	Niski	N
9	7	Piękny Kraków	Polska	2023-07-03	Sebastian	Chudy	P
10	8	Znów do Francji	Francja	2023-05-01	Albert	Gruby	P

3.4 Trips_no_places:

```
CREATE OR REPLACE VIEW TRIPS_NO_PLACES_VIEW
AS
   SELECT TRIP.TRIP_ID,
          TRIP_NAME,
          COUNTRY_NAME,
          TRIP_DATE,
          MAX_NO_PLACES,
          (MAX_NO_PLACES - COUNT(DISTINCT r.PERSON_ID)) NO_AVAILABLE_PLACES
FROM TRIP
   JOIN COUNTRY C on TRIP.COUNTRY_ID = C.COUNTRY_ID
   LEFT JOIN (SELECT * FROM RESERVATION r WHERE r.STATUS <> 'C') r
   ON r.TRIP_ID = TRIP.TRIP_ID
GROUP BY TRIP.TRIP_ID, TRIP_NAME, COUNTRY_NAME, TRIP_DATE, MAX_NO_PLACES;
```



BD_406838.TRIPS_NO_PLACES_VIEW

	TRIP_ID	TRIP_NAME	COUNTRY_NAME	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	6	Wycieczka do Paryza	Francja	2022-09-12	5	2
2	9	Hel	Polska	2023-05-01	5	2
3	8	Znów do Francji	Francja	2023-05-01	6	2
4	7	Piękny Kraków	Polska	2023-07-03	6	5

4. Procedury/funkcje:

4.1 Utworzenie obiektów pomocniczych:

Obiekty pomocnicze zostały utworzone dla procedur pobierających dane.

4.1.1 TRIPS_OBJ:

```
CREATE OR REPLACE TYPE TRIP_OBJ AS OBJECT
(
    TRIP_ID         NUMBER,
    TRIP_NAME       VARCHAR(100),
    COUNTRY_ID      NUMBER,
    "TRIP_DATE"          DATE,
    MAX_NO_PLACES   NUMBER
);

CREATE OR REPLACE TYPE TRIPS_TABLE IS TABLE OF TRIP_OBJ;
```

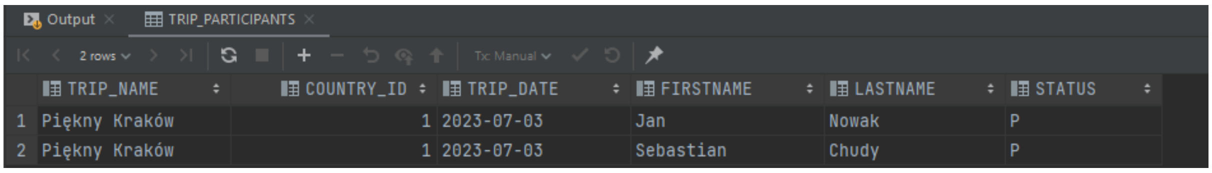
4.1.2 PERSON_TRIP_OBJ:

```
CREATE OR REPLACE TYPE PERSON_TRIP_OBJ AS OBJECT
(
    TRIP_NAME       NVARCHAR2(100),
    COUNTRY_ID      NUMBER,
    "TRIP_DATE"          DATE,
    FIRSTNAME       NVARCHAR2(50),
    LASTNAME        NVARCHAR2(50),
    STATUS          CHAR(1)
);

CREATE OR REPLACE TYPE PERSON_TRIP_TABLE IS TABLE OF PERSON_TRIP_OBJ;
```

4.2 Funkcje pobierające dane:

4.2.1 TripParticipants:

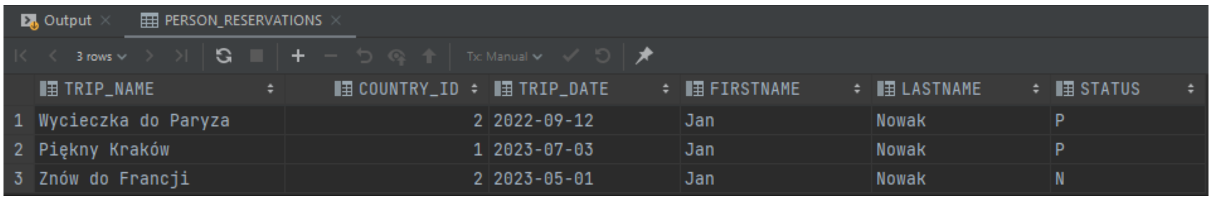
```
CREATE OR REPLACE FUNCTION TRIP_PARTICIPANTS(id TRIP.TRIP_ID%TYPE)
    RETURN PERSON_TRIP_TABLE
AS
    result PERSON_TRIP_TABLE;
    -- 1 - exists, 0 - not exists
    trip_exists INT;
BEGIN
    SELECT COUNT(*) INTO trip_exists
    FROM TRIP WHERE TRIP.TRIP_ID = TRIP_PARTICIPANTS.id;
    IF trip_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20000,'Trip with chosen ID not found');
    END IF;

    SELECT PERSON_TRIP_OBJ
        (TRIP_NAME,COUNTRY_ID,TRIP_DATE,FIRSTNAME,LASTNAME,STATUS)
        BULK COLLECT
    INTO result
    FROM TRIP
        JOIN RESERVATION ON TRIP.TRIP_ID = RESERVATION.TRIP_ID
        JOIN PERSON ON RESERVATION.PERSON_ID = PERSON.PERSON_ID
    WHERE TRIP.TRIP_ID = id
    AND RESERVATION.STATUS <> 'C';

    RETURN result;
END;
```



	TRIP_NAME	COUNTRY_ID	TRIP_DATE	FIRSTNAME	LASTNAME	STATUS
1	Piękny Kraków	1	2023-07-03	Jan	Nowak	P
2	Piękny Kraków	1	2023-07-03	Sebastian	Chudy	P

4.2.2 PersonReservations:

```sql
CREATE OR REPLACE FUNCTION PERSON_RESERVATIONS(id PERSON.PERSON_ID%TYPE)
    RETURN PERSON_TRIP_TABLE
AS
    result PERSON_TRIP_TABLE;
    -- 1-exists, 0-not exists
    trip_exists INT;
BEGIN
    SELECT COUNT(*) INTO trip_exists FROM PERSON WHERE PERSON.PERSON_ID =
PERSON_RESERVATIONS.id;
    IF trip_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20000,'Person with chosen ID not found');
    END IF;

    SELECT PERSON_TRIP_OBJ(TRIP_NAME,COUNTRY_ID,TRIP_DATE,FIRSTNAME,LASTNAME,STATUS)
        BULK COLLECT
    INTO result
    FROM TRIP
        JOIN RESERVATION ON TRIP.TRIP_ID = RESERVATION.TRIP_ID
        JOIN PERSON ON RESERVATION.PERSON_ID = PERSON.PERSON_ID
    WHERE PERSON.PERSON_ID = id
    AND RESERVATION.STATUS <> 'C';

    RETURN result;
END;
```

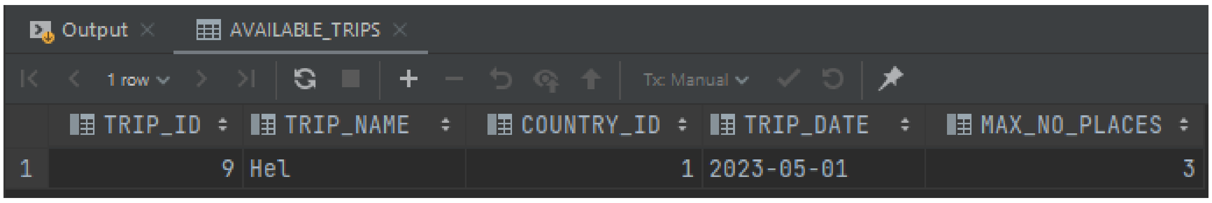


	TRIP_NAME	COUNTRY_ID	TRIP_DATE	FIRSTNAME	LASTNAME	STATUS
1	Wycieczka do Paryza	2	2022-09-12	Jan	Nowak	P
2	Piękny Kraków	1	2023-07-03	Jan	Nowak	P
3	Znów do Francji	2	2023-05-01	Jan	Nowak	N

4.2.3 AvailableTrips:

```sql
create FUNCTION AVAILABLE_TRIPS(country TRIP.COUNTRY_ID%TYPE,
    date_from DATE, date_to DATE)
    RETURN TRIPS_TABLE
AS
    result TRIPS_TABLE;
BEGIN
    IF date_from > date_to THEN
```

```
        RAISE_APPLICATION_ERROR(-20001,'Date_from must by <= date_to');
    end if;

    SELECT TRIP_OBJ(TRIP_ID,TRIP_NAME,COUNTRY_ID,TRIP_DATE,MAX_NO_PLACES)
    BULK COLLECT INTO result
    FROM TRIP
    WHERE TRIP.COUNTRY_ID = AVAILABLE_TRIPS.country
        AND TRIP.TRIP_DATE >= AVAILABLE_TRIPS.date_from
        AND TRIP.TRIP_DATE <= AVAILABLE_TRIPS.date_to
        AND TRIP.MAX_NO_PLACES >
            (SELECT COUNT(*)
             FROM RESERVATION r
             WHERE r.STATUS <> 'C'
                AND r.TRIP_ID = TRIP.TRIP_ID);
    RETURN result;
END;
```



```
   TRIP_ID  TRIP_NAME  COUNTRY_ID  TRIP_DATE   MAX_NO_PLACES
1        9  Hel                 1  2023-05-01              3
```

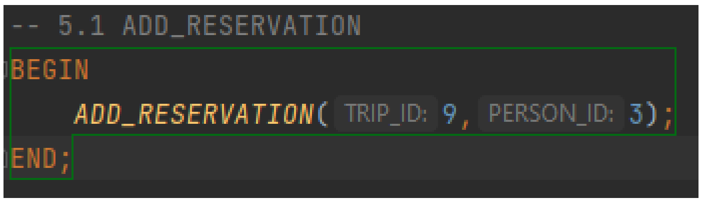
Wszystkie powyższe funkcje zwracają spodziewane wyniki.

5. Procedury modyfikujące dane:

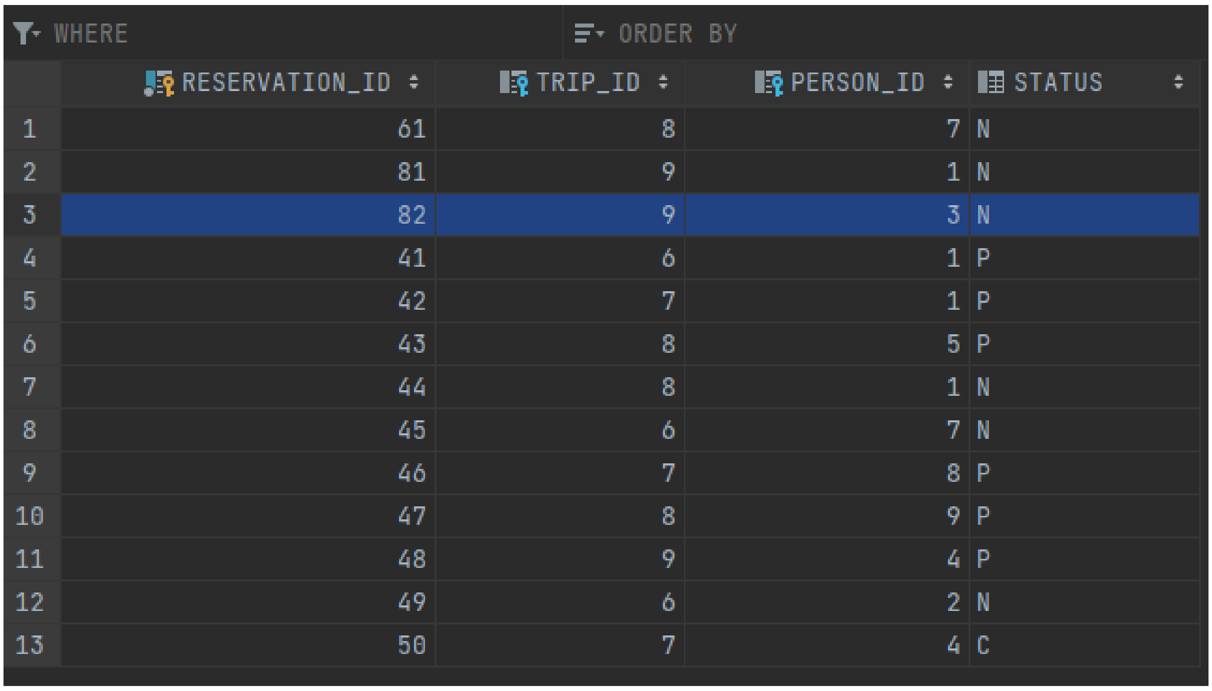
We wszystkich poniższych procedurach używam transakcji. W przypadku wystąpienia błędu w trakcie wykonywania zmian zostaje wywołany ROLLBACK (następuje on w trakcie wykonania RAISE_APPLICATION_ERROR). Jeśli cała procedura przebiegnie pomyślnie to transakcja zostaje zatwierdzona poprzez COMMIT. W poniższych procedurach uwzględniona jest już tabela LOG z pp.6, więc wywołania procedur dopisują nowe dane do tabeli LOG.

5.1 AddReservation:

```
create PROCEDURE
    ADD_RESERVATION(trip_id TRIP.TRIP_ID%TYPE,person_id PERSON.PERSON_ID%TYPE)
AS
    person_exists      INT;
    trip_available     INT;
    reservation_exists INT;
    new_reservation_id INT;
BEGIN
    -- check if person exists; if it does, should return count > 0
    SELECT COUNT(*) INTO person_exists FROM PERSON WHERE PERSON.PERSON_ID =
ADD_RESERVATION.person_id;

    IF person_exists = 0 THEN
        RAISE_APPLICATION_ERROR(-20002, 'Person with chosen ID not found.');
    END IF;

    -- check if trip exists and is available (is in future and has free places),
should return count > 0
    SELECT COUNT(*)
    INTO trip_available
    FROM AVAILABLE_TRIPS_VIEW
    WHERE AVAILABLE_TRIPS_VIEW.TRIP_ID = ADD_RESERVATION.trip_id;

    IF trip_available = 0 THEN
        RAISE_APPLICATION_ERROR(-20003, 'Trip with chosen ID not available.');
    END IF;

    -- checks if reservation exists; should return count = 0
    SELECT COUNT(*)
    INTO reservation_exists
    FROM RESERVATION
    WHERE RESERVATION.TRIP_ID = ADD_RESERVATION.trip_id
      AND RESERVATION.PERSON_ID = ADD_RESERVATION.person_id;
    IF reservation_exists > 0 THEN
        RAISE_APPLICATION_ERROR(-20004, 'Reservation for chosen trip ID and person ID
already exists.');
    END IF;

    -- save new automatically generated NR_REZERWACJI (it's generated as identity) in
    -- new_reservation_ID for logging
    INSERT INTO RESERVATION (TRIP_ID, PERSON_ID, STATUS)
    VALUES (ADD_RESERVATION.trip_id, ADD_RESERVATION.person_id, 'N')
    RETURNING RESERVATION_ID INTO new_reservation_ID;

    INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
    VALUES (new_reservation_id, CURRENT_DATE, 'N');


    -- INSERT is DML and does not automatically commit, and procedures don't
automatically commit either
    COMMIT;
END;
```



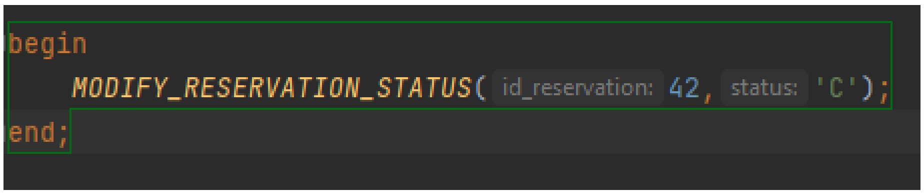
```
-- 5.1 ADD_RESERVATION
BEGIN
    ADD_RESERVATION( TRIP_ID: 9, PERSON_ID: 3);
END;
```



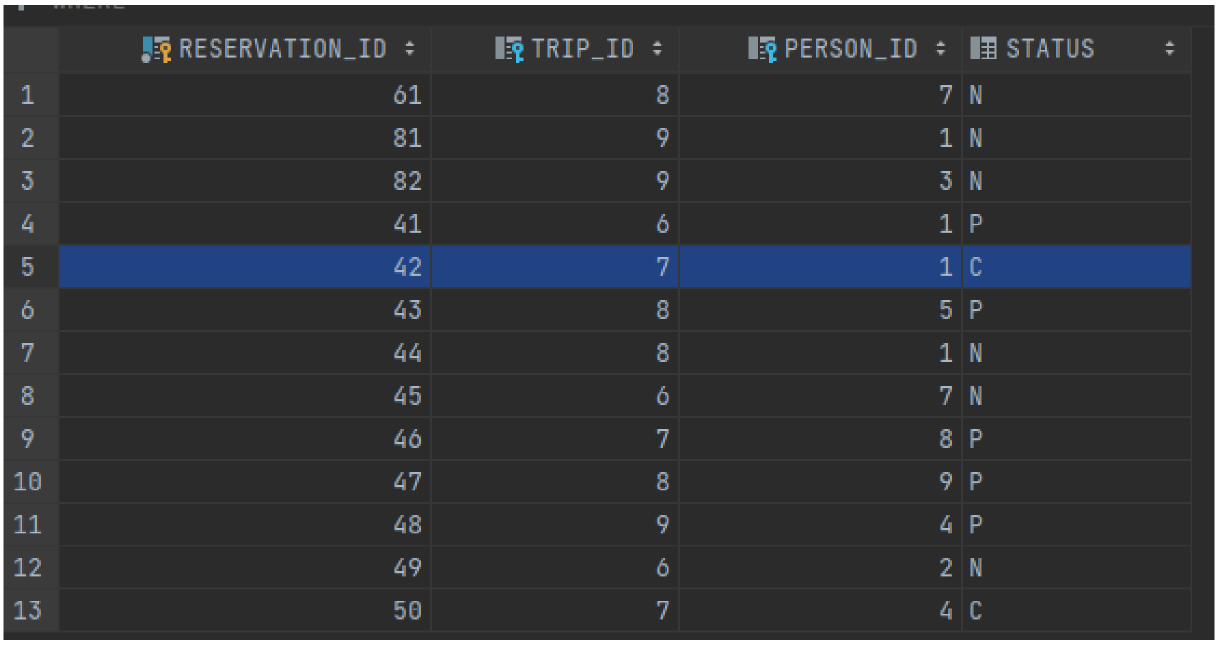
	RESERVATION_ID	TRIP_ID	PERSON_ID	STATUS
1	61	8	7	N
2	81	9	1	N
3	82	9	3	N
4	41	6	1	P
5	42	7	1	P
6	43	8	5	P
7	44	8	1	N
8	45	6	7	N
9	46	7	8	P
10	47	8	9	P
11	48	9	4	P
12	49	6	2	N
13	50	7	4	C

5.2 ModifyReservationStatus:

```
CREATE OR REPLACE PROCEDURE
   MODIFY_RESERVATION_STATUS(id_reservation RESERVATION.RESERVATION_ID%TYPE,
       status RESERVATION.STATUS%TYPE)
AS
   old_status RESERVATION.STATUS%TYPE;
   trip_exists INT;
BEGIN
   SELECT COUNT (*)
   INTO trip_exists
   FROM FUTURE_TRIPS_VIEW ft
   JOIN RESERVATION r ON ft.TRIP_ID = r.TRIP_ID
   WHERE r.RESERVATION_ID = MODIFY_RESERVATION_STATUS.id_reservation;

   IF trip_exists =  0 THEN
       RAISE_APPLICATION_ERROR(-20005,'Trip with chosen  ID not found');
   end if;

   SELECT STATUS INTO old_status FROM RESERVATION r
   WHERE r.RESERVATION_ID = MODIFY_RESERVATION_STATUS.id_reservation;

   CASE
       WHEN old_status IS NULL
           THEN RAISE_APPLICATION_ERROR(-20005,'Reservation with chosen ID not
found');
       WHEN old_status = 'C'
           THEN RAISE_APPLICATION_ERROR(-20006,'Status of canceled reservation cannot
be changed');
       WHEN old_status = 'P'
           THEN IF (MODIFY_RESERVATION_STATUS.status <> 'C')
               THEN RAISE_APPLICATION_ERROR(-20006,'Status of confirmed reservation
can be changed only to canceled');
               END IF;
       ELSE
           RAISE_APPLICATION_ERROR(-20999,'Internal application error');
   END CASE;

   UPDATE RESERVATION
   SET STATUS = MODIFY_RESERVATION_STATUS.status
   WHERE RESERVATION_ID = MODIFY_RESERVATION_STATUS.id_reservation;

   INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
   VALUES (MODIFY_RESERVATION_STATUS.id_reservation,
       CURRENT_DATE,
       MODIFY_RESERVATION_STATUS.status);


   COMMIT;
end;
```



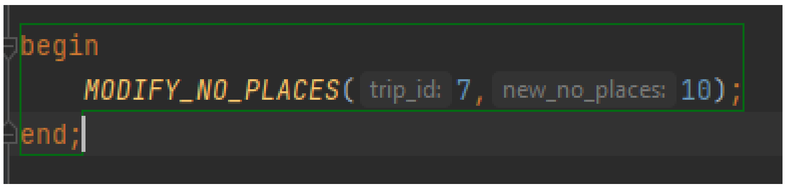
```
begin
    MODIFY_RESERVATION_STATUS( id_reservation: 42, status: 'C');
end;
```



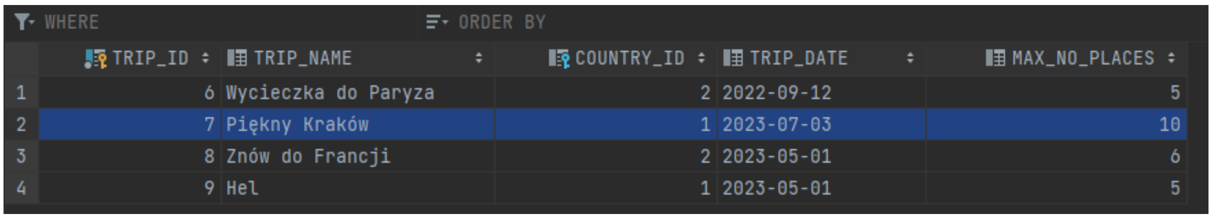
	RESERVATION_ID	TRIP_ID	PERSON_ID	STATUS
1	61	8	7	N
2	81	9	1	N
3	82	9	3	N
4	41	6	1	P
5	42	7	1	C
6	43	8	5	P
7	44	8	1	N
8	45	6	7	N
9	46	7	8	P
10	47	8	9	P
11	48	9	4	P
12	49	6	2	N
13	50	7	4	C

5.3 ModifyNoPlaces:

```
CREATE OR REPLACE PROCEDURE
    MODIFY_NO_PLACES(trip_id TRIP.TRIP_ID%TYPE,
                     new_no_places TRIP.MAX_NO_PLACES%TYPE)
AS
    trip_exists INT;
    reserved_places INT;
BEGIN
    SELECT COUNT (*)
    INTO trip_exists
    FROM TRIP t
    WHERE t.TRIP_ID = MODIFY_NO_PLACES.trip_id;

    IF trip_exists =  0 THEN
        RAISE_APPLICATION_ERROR(-20005,'Trip with chosen  ID not found');
    end if;

    SELECT tnp.MAX_NO_PLACES - tnp.NO_AVAILABLE_PLACES
    INTO reserved_places
    FROM TRIPS_NO_PLACES_VIEW tnp
    WHERE tnp.TRIP_ID = MODIFY_NO_PLACES.trip_id;

    IF new_no_places < 0 OR reserved_places > new_no_places
    THEN
        RAISE_APPLICATION_ERROR(-20007,'Too low number of free places');
    end if;

    UPDATE TRIP
    SET MAX_NO_PLACES = MODIFY_NO_PLACES.new_no_places
    WHERE TRIP_ID = MODIFY_NO_PLACES.trip_id;

    COMMIT;
end;
```



```
begin
    MODIFY_NO_PLACES( trip_id: 7, new_no_places: 10);
end;
```



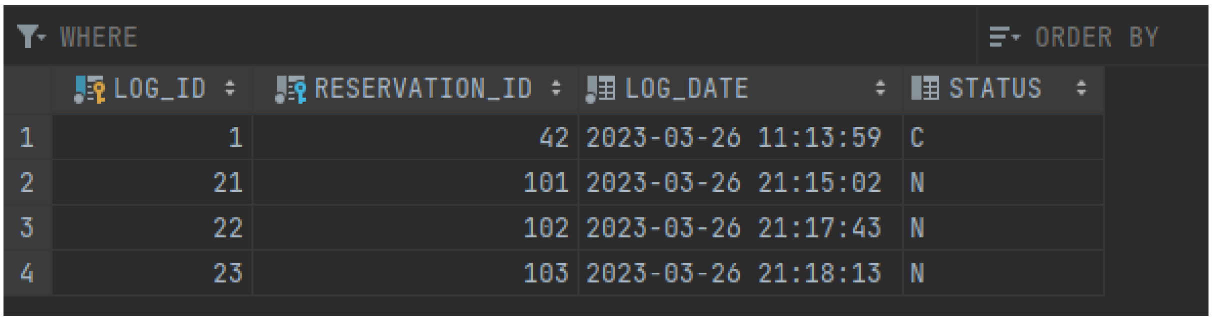
	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES
1	6	Wycieczka do Paryza	2	2022-09-12	5
2	7	Piękny Kraków	1	2023-07-03	10
3	8	Znów do Francji	2	2023-05-01	6
4	9	Hel	1	2023-05-01	5

Jak widać wszystkie powyższe procedury działają prawidłowo.

6. Dodanie tabeli dziennikującej zmiany statusu rezerwacji:

```
create table LOG
```

```
(
  LOG_ID NUMBER generated as identity
     constraint LOG_PK
        primary key,
  RESERVATION_ID NUMBER not null
     constraint LOG_FK1
        references RESERVATION,
  LOG_DATE DATE not null,
  STATUS CHAR
     constraint LOG_CHK1
        check (status in ('N','P','C'))
)
```



	LOG_ID	RESERVATION_ID	LOG_DATE	STATUS
1	1	42	2023-03-26 11:13:59	C
2	21	101	2023-03-26 21:15:02	N
3	22	102	2023-03-26 21:17:43	N
4	23	103	2023-03-26 21:18:13	N

Tabela LOG została częściowo wypełniona testowymi wywołaniami procedur.

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

7.1 Trigger obsługujący dodanie rezerwacji:

```
CREATE OR REPLACE TRIGGER ADD_RESERVATION_TRIGGER
   AFTER INSERT
   ON RESERVATION
   FOR EACH ROW
```

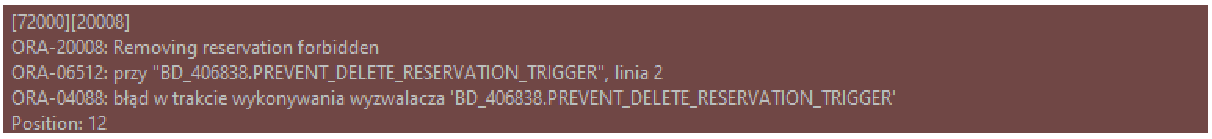
```
BEGIN
   INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
   VALUES (:NEW.RESERVATION_ID, CURRENT_DATE, :NEW.STATUS);
end;
```

7.2 Trigger obsługujący zmianę statusu:

```
CREATE OR REPLACE TRIGGER MODIFY_RESERVATION_STATUS_TRIGGER
   AFTER UPDATE
   ON RESERVATION
   FOR EACH ROW
BEGIN
   INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
   VALUES (:NEW.RESERVATION_ID, CURRENT_DATE, :NEW.STATUS);
end MODIFY_RESERVATION_STATUS_TRIGGER;
```

7.3 Trigger zabraniający usunięcia rezerwacji:

```
CREATE OR REPLACE TRIGGER PREVENT_DELETE_RESERVATION_TRIGGER
   BEFORE DELETE
   ON RESERVATION
   FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20008, 'Removing reservation forbidden');
end;
```



```
[72000][20008]
ORA-20008: Removing reservation forbidden
ORA-06512: przy "BD_406838.PREVENT_DELETE_RESERVATION_TRIGGER", linia 2
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'BD_406838.PREVENT_DELETE_RESERVATION_TRIGGER'
Position: 12
```

7.4 Zmodyfikowane procedury modyfikujące dane:

7.4.1 Add reservation ver.2

```
create PROCEDURE
   ADD_RESERVATION_2(trip_id TRIP.TRIP_ID%TYPE,person_id PERSON.PERSON_ID%TYPE)
AS
   person_exists       INT;
```

```sql
   trip_available     INT;
   reservation_exists INT;
   new_reservation_id INT;
BEGIN
   -- check if person exists; if it does, should return count > 0
   SELECT COUNT(*) INTO person_exists FROM PERSON WHERE PERSON.PERSON_ID =
ADD_RESERVATION_2.person_id;

   IF person_exists = 0 THEN
       RAISE_APPLICATION_ERROR(-20002, 'Person with chosen ID not found.');
   END IF;

   -- check if trip exists and is available (is in future and has free places),
should return count > 0
   SELECT COUNT(*)
   INTO trip_available
   FROM AVAILABLE_TRIPS_VIEW
   WHERE AVAILABLE_TRIPS_VIEW.TRIP_ID = ADD_RESERVATION_2.trip_id;

   IF trip_available = 0 THEN
       RAISE_APPLICATION_ERROR(-20003, 'Trip with chosen ID not available.');
   END IF;

   -- checks if reservation exists; should return count = 0
   SELECT COUNT(*)
   INTO reservation_exists
   FROM RESERVATION
   WHERE RESERVATION.TRIP_ID = ADD_RESERVATION_2.trip_id
     AND RESERVATION.PERSON_ID = ADD_RESERVATION_2.person_id;

   IF reservation_exists > 0 THEN
       RAISE_APPLICATION_ERROR(-20004, 'Reservation for chosen trip ID and person ID
already exists.');
   END IF;

   -- save new automatically generated NR_REZERWACJI (it's generated as identity) in
   -- new_reservation_ID for logging
   INSERT INTO RESERVATION (TRIP_ID, PERSON_ID, STATUS)
   VALUES (ADD_RESERVATION_2.trip_id, ADD_RESERVATION_2.person_id, 'N')
   RETURNING RESERVATION_ID INTO new_reservation_ID;

   -- INSERT is DML and does not automatically commit, and procedures don't
automatically commit either
   COMMIT;
END;
```

7.4.2 Modify reservation status ver.2:

```
create PROCEDURE
   MODIFY_RESERVATION_STATUS_2(id_reservation RESERVATION.RESERVATION_ID%TYPE,
       status RESERVATION.STATUS%TYPE)
AS
   old_status RESERVATION.STATUS%TYPE;
   trip_exists INT;
   t_noa_places INT;
    trip_id INT;

BEGIN
   SELECT COUNT (*)
   INTO trip_exists
   FROM FUTURE_TRIPS_VIEW ft
   JOIN RESERVATION r ON ft.TRIP_ID = r.TRIP_ID
   WHERE r.RESERVATION_ID = MODIFY_RESERVATION_STATUS_2.id_reservation;

   IF trip_exists =  0 THEN
       RAISE_APPLICATION_ERROR(-20005,'Trip with chosen  ID not found');
   end if;

   SELECT STATUS INTO old_status FROM RESERVATION r
   WHERE r.RESERVATION_ID = MODIFY_RESERVATION_STATUS_2.id_reservation;

   CASE
       WHEN old_status IS NULL
           THEN RAISE_APPLICATION_ERROR(-20005,'Reservation with chosen ID not
found');
       WHEN old_status = 'C'
           THEN RAISE_APPLICATION_ERROR(-20006,'Status of canceled reservation cannot
be changed');
       WHEN old_status = 'P'
           THEN IF (MODIFY_RESERVATION_STATUS_2.status <> 'C')
               THEN RAISE_APPLICATION_ERROR(-20006,'Status of confirmed reservation
can be changed only to canceled');
               END IF;
       ELSE
           RAISE_APPLICATION_ERROR(-20999,'Internal application error');
   END CASE;

   SELECT TRIP_ID INTO trip_id
    FROM RESERVATION
    WHERE RESERVATION_ID = MODIFY_RESERVATION_STATUS_2.id_reservation;

    SELECT NO_AVAILABLE_PLACES
    INTO t_noa_places
    FROM TRIP
    WHERE TRIP.TRIP_ID = trip_id;
    IF t_noa_places < 1 THEN
       RAISE_APPLICATION_ERROR(-20001,'Not enough places');
    end if;


   UPDATE RESERVATION
   SET STATUS = MODIFY_RESERVATION_STATUS_2.status
   WHERE RESERVATION_ID = MODIFY_RESERVATION_STATUS_2.id_reservation;

   COMMIT;
end;
```

Ciężko było znaleźć trigger dla którego można by wykonać zrzut ekranu, więc jedynym screen'em jest z działania triggera zabraniającego usunięcia rezerwacji.

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

8.1 Trigger obsługujący dodanie rezerwacji:

```
create trigger ADD_RESERVATION_TRIGGER_2
   before insert
   on RESERVATION
   for each row
DECLARE
   trip_available INT;
BEGIN

   -- check if trip exists and is available (is in future and has free places),
should return count > 0
   SELECT COUNT(*)
   INTO trip_available
   FROM AVAILABLE_TRIPS_VIEW
   WHERE AVAILABLE_TRIPS_VIEW.TRIP_ID = :NEW.trip_id;

   IF trip_available = 0 THEN
       RAISE_APPLICATION_ERROR(-20003, 'Trip with chosen ID not available.');
   END IF;
end;
```

8.2 Trigger obsługujący zmianę statusu:

```
CREATE OR REPLACE TRIGGER MODIFY_RESERVATION_STATUS_TRIGGER_2
   BEFORE UPDATE
   ON RESERVATION
   FOR EACH ROW
DECLARE
   PRAGMA AUTONOMOUS_TRANSACTION;
   t_noa_places INT;
BEGIN
   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 TRIP
                   WHERE TRIP.TRIP_ID = :NEW.TRIP_ID;
               IF t_noa_places >= 1 THEN
                   COMMIT;
               ELSE
                   RAISE_APPLICATION_ERROR(-20001,'Not enough places');
               end if;
           end if;
       end if;
   end if;
end;
```

8.3 Aktualizacja procedur:

8.3.1 ADD_RESERVATION_3:

```
create PROCEDURE
   ADD_RESERVATION_3(trip_id TRIP.TRIP_ID%TYPE,person_id PERSON.PERSON_ID%TYPE)
AS
   person_exists       INT;
   reservation_exists INT;
   new_reservation_id INT;
BEGIN
   -- check if person exists; if it does, should return count > 0
   SELECT COUNT(*) INTO person_exists FROM PERSON WHERE PERSON.PERSON_ID =
ADD_RESERVATION_3.person_id;

   IF person_exists = 0 THEN
       RAISE_APPLICATION_ERROR(-20002, 'Person with chosen ID not found.');
   END IF;

   -- checks if reservation exists; should return count = 0
   SELECT COUNT(*)
   INTO reservation_exists
   FROM RESERVATION
   WHERE RESERVATION.TRIP_ID = ADD_RESERVATION_3.trip_id
     AND RESERVATION.PERSON_ID = ADD_RESERVATION_3.person_id;

   IF reservation_exists > 0 THEN
       RAISE_APPLICATION_ERROR(-20004, 'Reservation for chosen trip ID and person ID
already exists.');
   END IF;

   -- save new automatically generated NR_REZERWACJI (it's generated as indentity) in
   -- new_reservation_ID for logging
   INSERT INTO RESERVATION (TRIP_ID, PERSON_ID, STATUS)
   VALUES (ADD_RESERVATION_3.trip_id, ADD_RESERVATION_3.person_id, 'N')
   RETURNING RESERVATION_ID INTO new_reservation_ID;

   INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
   VALUES (new_reservation_id, CURRENT_DATE, 'N');

   -- INSERT is DML and does not automatically commit, and procedures don't
automatically commit either
   COMMIT;
END;
```

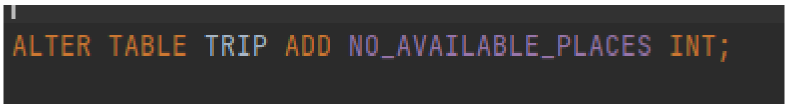
```
/
```

8.3.2 MODIFY_RESERVATION_STATUS_3:

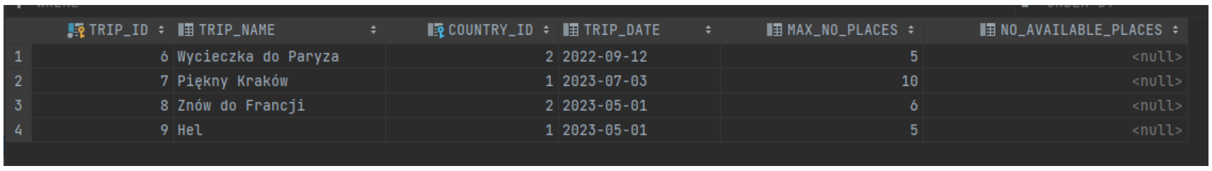
```
create PROCEDURE
   MODIFY_RESERVATION_STATUS_3(id_reservation RESERVATION.RESERVATION_ID%TYPE,
       status RESERVATION.STATUS%TYPE)
AS
   old_status RESERVATION.STATUS%TYPE;
   trip_exists INT;
BEGIN
   SELECT COUNT (*)
   INTO trip_exists
   FROM FUTURE_TRIPS_VIEW ft
   JOIN RESERVATION r ON ft.TRIP_ID = r.TRIP_ID
   WHERE r.RESERVATION_ID = MODIFY_RESERVATION_STATUS_3.id_reservation;

   IF trip_exists =  0 THEN
       RAISE_APPLICATION_ERROR(-20005,'Trip with chosen  ID not found');
   end if;

   SELECT STATUS INTO old_status FROM RESERVATION r
   WHERE r.RESERVATION_ID = MODIFY_RESERVATION_STATUS_3.id_reservation;

   CASE
       WHEN old_status IS NULL
           THEN RAISE_APPLICATION_ERROR(-20005,'Reservation with chosen ID not
found');
       WHEN old_status = 'C'
           THEN RAISE_APPLICATION_ERROR(-20006,'Status of canceled reservation cannot
be changed');
       WHEN old_status = 'P'
           THEN IF (MODIFY_RESERVATION_STATUS_3.status <> 'C')
               THEN RAISE_APPLICATION_ERROR(-20006,'Status of confirmed reservation
can be changed only to canceled');
               END IF;
       ELSE
           RAISE_APPLICATION_ERROR(-20999,'Internal application error');
   END CASE;

   UPDATE RESERVATION
   SET STATUS = MODIFY_RESERVATION_STATUS_3.status
   WHERE RESERVATION_ID = MODIFY_RESERVATION_STATUS_3.id_reservation;
```

```
    COMMIT;
end;
/
```

9. Dodanie pola no_available_places:



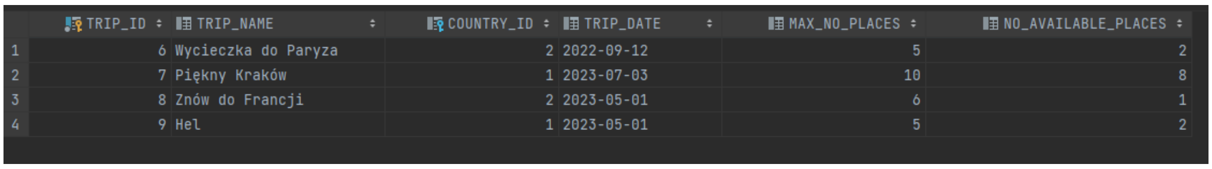
```
ALTER TABLE TRIP ADD NO_AVAILABLE_PLACES INT;
```



	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	6	Wycieczka do Paryza	2	2022-09-12	5	<null>
2	7	Piękny Kraków	1	2023-07-03	10	<null>
3	8	Znów do Francji	2	2023-05-01	6	<null>
4	9	Hel	1	2023-05-01	5	<null>

9.1 Dodanie procedury aktualizującej dane dla nowej wersji tabeli TRIP:

```
CREATE OR REPLACE PROCEDURE RECALCULATE
AS
BEGIN
   UPDATE TRIP t
   SET NO_AVAILABLE_PLACES =
       MAX_NO_PLACES - (SELECT COUNT(*) FROM RESERVATION r
                        WHERE r.TRIP_ID = t.TRIP_ID
                        AND r.STATUS <> 'C');
end;
```



	TRIP_ID	TRIP_NAME	COUNTRY_ID	TRIP_DATE	MAX_NO_PLACES	NO_AVAILABLE_PLACES
1	6	Wycieczka do Paryza	2	2022-09-12	5	2
2	7	Piękny Kraków	1	2023-07-03	10	8
3	8	Znów do Francji	2	2023-05-01	6	1
4	9	Hel	1	2023-05-01	5	2

9.2 Aktualizacja widoków:

9.2.1 TRIPS_NO_PLACES_VIEW_4:

```
create view TRIPS_NO_PLACES_VIEW_4 as
SELECT TRIP.TRIP_ID,
          TRIP_NAME,
          COUNTRY_NAME,
          TRIP_DATE,
          MAX_NO_PLACES,
          NO_AVAILABLE_PLACES
FROM TRIP
   JOIN COUNTRY C on TRIP.COUNTRY_ID = C.COUNTRY_ID
   LEFT JOIN (SELECT * FROM RESERVATION r WHERE r.STATUS <> 'C') r
   ON r.TRIP_ID = TRIP.TRIP_ID
GROUP BY TRIP.TRIP_ID, TRIP_NAME, COUNTRY_NAME, TRIP_DATE, MAX_NO_PLACES,
NO_AVAILABLE_PLACES
```

9.2.2 AVAILABLE_TRIPS_VIEW_4:

```
create view AVAILABLE_TRIPS_VIEW_4 as
SELECT COUNTRY_NAME,
          trip.TRIP_ID,
          TRIP_NAME,
          TRIP_DATE,
          MAX_NO_PLACES,
          NO_AVAILABLE_PLACES
   FROM COUNTRY
   JOIN TRIP on COUNTRY.COUNTRY_ID = trip.COUNTRY_ID
   LEFT JOIN (SELECT * FROM RESERVATION WHERE STATUS <> 'C') RESERVATION
   ON RESERVATION.TRIP_ID = TRIP.TRIP_ID
   GROUP BY COUNTRY.COUNTRY_NAME,TRIP.TRIP_ID, TRIP.TRIP_NAME, TRIP.TRIP_DATE,
TRIP.MAX_NO_PLACES, TRIP.NO_AVAILABLE_PLACES
   HAVING TRIP_DATE > current_date AND MAX_NO_PLACES- COUNT(DISTINCT PERSON_ID) > 0
```

9.3 Aktualizacja procedur:

9.3.1 ADD_RESERVATION_4:

```
CREATE OR REPLACE TRIGGER ADD_RESERVATION_TRIGGER
   AFTER INSERT
   ON RESERVATION
   FOR EACH ROW
BEGIN
   INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
   VALUES (:NEW .RESERVATION_ID, CURRENT_DATE, :NEW .STATUS);
end;

CREATE OR REPLACE TRIGGER MODIFY_RESERVATION_STATUS_TRIGGER
   AFTER UPDATE
   ON RESERVATION
   FOR EACH ROW
BEGIN
   INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
   VALUES (:NEW .RESERVATION_ID, CURRENT_DATE, :NEW .STATUS);
end MODIFY_RESERVATION_STATUS_TRIGGER;

CREATE OR REPLACE TRIGGER PREVENT_DELETE_RESERVATION_TRIGGER
   BEFORE DELETE
   ON RESERVATION
   FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20008, 'Removing reservation forbidden');
end;

create PROCEDURE
   ADD_RESERVATION_4(trip_id TRIP.TRIP_ID%TYPE,person_id PERSON.PERSON_ID%TYPE)
AS
   person_exists      INT;
   trip_available     INT;
   reservation_exists INT;
   new_reservation_id INT;
BEGIN
   -- check if person exists; if it does, should return count > 0
   SELECT COUNT(*) INTO person_exists FROM PERSON WHERE PERSON.PERSON_ID =
ADD_RESERVATION_4.person_id;

   IF person_exists = 0 THEN
       RAISE_APPLICATION_ERROR(-20002, 'Person with chosen ID not found.');
   END IF;
   -- check if trip exists and is available (is in future and has free places),
should return count > 0
   SELECT COUNT(*)
   INTO trip_available
   FROM AVAILABLE_TRIPS_VIEW
   WHERE AVAILABLE_TRIPS_VIEW.TRIP_ID = ADD_RESERVATION_4.trip_id;

   IF trip_available = 0 THEN
       RAISE_APPLICATION_ERROR(-20003, 'Trip with chosen ID not available.');
   END IF;

   -- checks if reservation exists; should return count = 0
   SELECT COUNT(*)
   INTO reservation_exists
   FROM RESERVATION
   WHERE RESERVATION.TRIP_ID = ADD_RESERVATION_4.trip_id
     AND RESERVATION.PERSON_ID = ADD_RESERVATION_4.person_id;

   IF reservation_exists > 0 THEN
       RAISE_APPLICATION_ERROR(-20004, 'Reservation for chosen trip ID and person ID
already exists.');
   END IF;

   -- save new automatically generated NR_REZERWACJI (it's generated as identity) in
   -- new_reservation_ID for logging
   INSERT INTO RESERVATION (TRIP_ID, PERSON_ID, STATUS)
   VALUES (ADD_RESERVATION_4.trip_id, ADD_RESERVATION_4.person_id, 'N')
   RETURNING RESERVATION_ID INTO new_reservation_ID;

   UPDATE TRIP
   SET NO_AVAILABLE_PLACES = NO_AVAILABLE_PLACES-1
   WHERE TRIP_ID = ADD_RESERVATION_4.trip_id;

   -- INSERT is DML and does not automatically commit, and procedures don't
automatically commit either
   COMMIT;
END;
```

9.3.2 AVAILABLE_TRIPS_4:

```
create FUNCTION AVAILABLE_TRIPS_4(country TRIP.COUNTRY_ID%TYPE,
   date_from DATE, date_to DATE)
   RETURN TRIPS_TABLE
```

```
AS
   result TRIPS_TABLE;
BEGIN
   IF date_from > date_to THEN
       RAISE_APPLICATION_ERROR(-20001,'Date_from must by <= date_to');
   end if;

   SELECT TRIP_OBJ(TRIP_ID,TRIP_NAME,COUNTRY_ID,TRIP_DATE,MAX_NO_PLACES)
   BULK COLLECT INTO result
   FROM TRIP
   WHERE TRIP.COUNTRY_ID = AVAILABLE_TRIPS_4.country
       AND TRIP.TRIP_DATE >= AVAILABLE_TRIPS_4.date_from
       AND TRIP.TRIP_DATE <= AVAILABLE_TRIPS_4.date_to
       AND TRIP.NO_AVAILABLE_PLACES > 0;
   RETURN result;
END;
```

9.3.3 MODIFY_RESERVATION_STATUS_4:

```
create PROCEDURE
   MODIFY_RESERVATION_STATUS_4(id_reservation RESERVATION.RESERVATION_ID%TYPE,
       status RESERVATION.STATUS%TYPE)
AS
   old_status RESERVATION.STATUS%TYPE;
   trip_exists INT;
   new_places INT;
BEGIN
   SELECT COUNT (*)
   INTO trip_exists
   FROM FUTURE_TRIPS_VIEW ft
   JOIN RESERVATION r ON ft.TRIP_ID = r.TRIP_ID
   WHERE r.RESERVATION_ID = MODIFY_RESERVATION_STATUS_4.id_reservation;

   IF trip_exists =  0 THEN
       RAISE_APPLICATION_ERROR(-20005,'Trip with chosen  ID not found');
   end if;

   SELECT STATUS INTO old_status FROM RESERVATION r
   WHERE r.RESERVATION_ID = MODIFY_RESERVATION_STATUS_4.id_reservation;
   CASE
       WHEN old_status IS NULL
           THEN RAISE_APPLICATION_ERROR(-20005,'Reservation with chosen ID not
found');
       WHEN old_status = 'C'
           THEN RAISE_APPLICATION_ERROR(-20006,'Status of canceled reservation cannot
be changed');
       WHEN old_status = 'P'
           THEN IF (MODIFY_RESERVATION_STATUS_4.status <> 'C')
               THEN RAISE_APPLICATION_ERROR(-20006,'Status of confirmed reservation
can be changed only to canceled');
               END IF;
       ELSE
           RAISE_APPLICATION_ERROR(-20999,'Internal application error');
   END CASE;

   UPDATE RESERVATION
   SET STATUS = MODIFY_RESERVATION_STATUS_4.status
   WHERE RESERVATION_ID = MODIFY_RESERVATION_STATUS_4.id_reservation;

   IF status = 'C' THEN
       new_places := 1;
   ELSE
       new_places := 0;
   end if;

   UPDATE TRIP t
   SET NO_AVAILABLE_PLACES = NO_AVAILABLE_PLACES + new_places
   WHERE t.TRIP_ID = (SELECT TRIP_ID FROM RESERVATION r
                       WHERE r.RESERVATION_ID =
MODIFY_RESERVATION_STATUS_4.id_reservation);

   COMMIT;
```

9.3.4 MODIFY_NO_PLACES_4:

```
create PROCEDURE
   MODIFY_NO_PLACES_4(trip_id TRIP.TRIP_ID%TYPE,
                   new_no_places TRIP.MAX_NO_PLACES%TYPE)
AS
   trip_exists INT;
   reserved_places INT;
BEGIN
   SELECT COUNT (*)
   INTO trip_exists
   FROM TRIP t
   WHERE t.TRIP_ID = MODIFY_NO_PLACES_4.trip_id;

   IF trip_exists =  0 THEN
       RAISE_APPLICATION_ERROR(-20005,'Trip with chosen  ID not found');
   end if;

   SELECT MAX_NO_PLACES-NO_AVAILABLE_PLACES
   INTO reserved_places
   FROM TRIP
   WHERE TRIP.TRIP_ID = MODIFY_NO_PLACES_4.trip_id;

   IF new_no_places < 0 OR reserved_places > new_no_places
   THEN
       RAISE_APPLICATION_ERROR(-20007,'Too low number of free places');
   end if;

   UPDATE TRIP
   SET MAX_NO_PLACES = MODIFY_NO_PLACES_4.new_no_places,
       NO_AVAILABLE_PLACES = NO_AVAILABLE_PLACES + (new_no_places-TRIP.MAX_NO_PLACES)
   WHERE TRIP_ID = MODIFY_NO_PLACES_4.trip_id;

   COMMIT;
end;
```

10. Zmiana strategii obsługi redundantnego pola no_available_places, realizacja przy pomocy triggerów:

10.1 ADD_RESERVATION_TRIGGER:

```
CREATE OR REPLACE TRIGGER ADD_RESERVATION_TRIGGER
   AFTER INSERT
   ON RESERVATION
   FOR EACH ROW
BEGIN
   INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
   VALUES (:NEW .RESERVATION_ID, CURRENT_DATE, :NEW .STATUS);

   UPDATE TRIP t
   SET NO_AVAILABLE_PLACES = NO_AVAILABLE_PLACES-1
   WHERE t.TRIP_ID = :NEW .TRIP_ID;
end;
```

10.2 MODIFY_RESERVATION_STATUS_TRIGGER:

```
CREATE OR REPLACE TRIGGER MODIFY_RESERVATION_STATUS_TRIGGER
   AFTER UPDATE
   ON RESERVATION
   FOR EACH ROW
DECLARE
   new_places INT;
BEGIN
   INSERT INTO LOG (RESERVATION_ID, LOG_DATE, STATUS)
   VALUES (:NEW .RESERVATION_ID, CURRENT_DATE, :NEW .STATUS);

   IF :NEW .STATUS = 'C' THEN
       new_places := 1;
   ELSE
       new_places := 0;
   end if;

   UPDATE TRIP t
   SET NO_AVAILABLE_PLACES = NO_AVAILABLE_PLACES+new_places
   WHERE t.TRIP_ID = :NEW .trip_id;

end MODIFY_RESERVATION_STATUS_TRIGGER;
```

10.3 Zmiana liczby miejsc na poziomie wycieczki (MODIFY_NO_PLACES_TRIGGER):

```
CREATE OR REPLACE TRIGGER MODIFY_NO_PLACES_TRIGGER
   AFTER UPDATE OF MAX_NO_PLACES
   ON TRIP
   FOR EACH ROW
BEGIN
```

```
   UPDATE TRIP
   SET NO_AVAILABLE_PLACES = NO_AVAILABLE_PLACES + (:NEW .MAX_NO_PLACES
 - TRIP.MAX_NO_PLACES);
end;
```

10.4 Modyfikacja procedur:

10.4.1 ADD_RESERVATION_5:

```
create PROCEDURE
   ADD_RESERVATION_5(trip_id TRIP.TRIP_ID%TYPE,person_id PERSON.PERSON_ID%TYPE)
AS
   person_exists      INT;
   trip_available     INT;
   reservation_exists INT;
   new_reservation_id INT;
BEGIN
   -- check if person exists; if it does, should return count > 0
   SELECT COUNT(*) INTO person_exists FROM PERSON WHERE PERSON.PERSON_ID =
ADD_RESERVATION_5.person_id;

   IF person_exists = 0 THEN
       RAISE_APPLICATION_ERROR(-20002, 'Person with chosen ID not found.');
   END IF;

   -- check if trip exists and is available (is in future and has free places),
should return count > 0
   SELECT COUNT(*)
   INTO trip_available
   FROM AVAILABLE_TRIPS_VIEW
   WHERE AVAILABLE_TRIPS_VIEW.TRIP_ID = ADD_RESERVATION_5.trip_id;

   IF trip_available = 0 THEN
       RAISE_APPLICATION_ERROR(-20003, 'Trip with chosen ID not available.');
   END IF;

   -- checks if reservation exists; should return count = 0
   SELECT COUNT(*)
   INTO reservation_exists
   FROM RESERVATION
   WHERE RESERVATION.TRIP_ID = ADD_RESERVATION_5.trip_id
     AND RESERVATION.PERSON_ID = ADD_RESERVATION_5.person_id;
   IF reservation_exists > 0 THEN
       RAISE_APPLICATION_ERROR(-20004, 'Reservation for chosen trip ID and person ID
already exists.');
   END IF;

   -- save new automatically generated NR_REZERWACJI (it's generated as identity) in
   -- new_reservation_ID for logging
   INSERT INTO RESERVATION (TRIP_ID, PERSON_ID, STATUS)
   VALUES (ADD_RESERVATION_5.trip_id, ADD_RESERVATION_5.person_id, 'N')
   RETURNING RESERVATION_ID INTO new_reservation_ID;

   -- INSERT is DML and does not automatically commit, and procedures don't
automatically commit either
   COMMIT;
END;
```

10.4.2 MODIFY_RESERVATION_STATUS_5:

```
create PROCEDURE
   MODIFY_RESERVATION_STATUS_5(id_reservation RESERVATION.RESERVATION_ID%TYPE,
       status RESERVATION.STATUS%TYPE)
AS
   old_status RESERVATION.STATUS%TYPE;
   trip_exists INT;
   new_places INT;
BEGIN
   SELECT COUNT (*)
   INTO trip_exists
   FROM FUTURE_TRIPS_VIEW ft
   JOIN RESERVATION r ON ft.TRIP_ID = r.TRIP_ID
   WHERE r.RESERVATION_ID = MODIFY_RESERVATION_STATUS_5.id_reservation;

   IF trip_exists =  0 THEN
       RAISE_APPLICATION_ERROR(-20005,'Trip with chosen  ID not found');
   end if;

   SELECT STATUS INTO old_status FROM RESERVATION r
   WHERE r.RESERVATION_ID = MODIFY_RESERVATION_STATUS_5.id_reservation;

   CASE
       WHEN old_status IS NULL
           THEN RAISE_APPLICATION_ERROR(-20005,'Reservation with chosen ID not
found');
       WHEN old_status = 'C'
           THEN RAISE_APPLICATION_ERROR(-20006,'Status of canceled reservation cannot
be changed');
       WHEN old_status = 'P'
           THEN IF (MODIFY_RESERVATION_STATUS_5.status <> 'C')
               THEN RAISE_APPLICATION_ERROR(-20006,'Status of confirmed reservation
can be changed only to canceled');
               END IF;
       ELSE
           RAISE_APPLICATION_ERROR(-20999,'Internal application error');
   END CASE;

   UPDATE RESERVATION
   SET STATUS = MODIFY_RESERVATION_STATUS_5.status
   WHERE RESERVATION_ID = MODIFY_RESERVATION_STATUS_5.id_reservation;

   COMMIT;
end;
```

10.4.3 MODIFY_NO_PLACES_5:

```
create PROCEDURE
   MODIFY_NO_PLACES_5(trip_id TRIP.TRIP_ID%TYPE,
                   new_no_places TRIP.MAX_NO_PLACES%TYPE)
AS
   trip_exists INT;
   reserved_places INT;
BEGIN
   SELECT COUNT (*)
   INTO trip_exists
   FROM TRIP t
   WHERE t.TRIP_ID = MODIFY_NO_PLACES_5.trip_id;

   IF trip_exists =  0 THEN
       RAISE_APPLICATION_ERROR(-20005,'Trip with chosen  ID not found');
   end if;

   SELECT MAX_NO_PLACES-NO_AVAILABLE_PLACES
   INTO reserved_places
   FROM TRIP
   WHERE TRIP.TRIP_ID = MODIFY_NO_PLACES_5.trip_id;

   IF new_no_places < 0 OR reserved_places > new_no_places
   THEN
       RAISE_APPLICATION_ERROR(-20007,'Too low number of free places');
   end if;
```

```sql

   UPDATE TRIP
   SET MAX_NO_PLACES = MODIFY_NO_PLACES_5.new_no_places
   WHERE TRIP_ID = MODIFY_NO_PLACES_5.trip_id;

   COMMIT;
end;
```