# Gracjan Filipek – raport, Oracle PL/SQL

# Programowanie proceduralne - widoki, procedury, triggery itp.

# I. Oracle PL/SQL

1. Tabele

Trip (trip\_id, name, country, date, no\_places)

Person(person\_id, firstname, lastname)

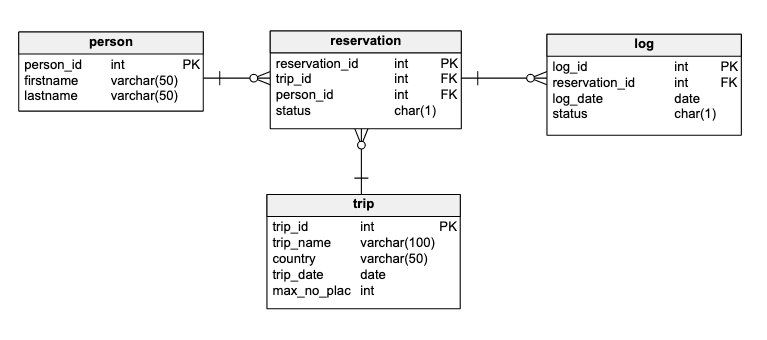
Reservation(reservation\_id, trip\_id, person\_id, status)

Pole status w tabeli Rezerwacje może przyjmować jedną z 4 wartości

N – New

P – Paid

C – Canceled



1a) Wypełnianie tabele przykładowymi danymi

Zmodyfikuj początkowy schemat bazy danych dodając tabelę "słownikową" z listą krajów

### Kod

create table country  
(  
 country\_name varchar(50),  
 constraint coutry\_pk primary key (country\_name) enable  
);

1. Wypełnianie tabele przykładowymi danymi

4 wycieczki

10 osób

10 rezerwacji

### Kod

-- 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('Grecja');  
  
-- trip  
insert into trip(trip\_name, country, trip\_date, max\_no\_places)  
values ('Wycieczka do Paryza', 'Francja', *to\_date*('2022-09-12','YYYY-MM-DD'), 3);  
  
insert into trip(trip\_name, country, trip\_date, max\_no\_places)  
values ('Piękny Kraków', 'Polska', *to\_date*('2023-07-03','YYYY-MM-DD'), 2);  
  
insert into trip(trip\_name, country, trip\_date, max\_no\_places)  
values ('Znów do Francji', 'Francja', *to\_date*('2023-05-01','YYYY-MM-DD'), 5);  
  
insert into trip(trip\_name, country, trip\_date, max\_no\_places)  
values ('Hel', 'Polska', *to\_date*('2023-05-01','YYYY-MM-DD'), 7);  
  
  
-- person  
insert into person(firstname, lastname)  
values ('Jan', 'Nowak');  
  
insert into person(firstname, lastname)  
values ('Jan', 'Kowalski');  
  
insert into person(firstname, lastname)  
values ('Jan', 'Nowakowski');  
  
insert into person(firstname, lastname)  
values ('Adam', 'Kowalski');  
  
insert into person(firstname, lastname)  
values ('Novak', 'Nowak');  
  
insert into person(firstname, lastname)  
values ('Piotr', 'Piotrowski');  
  
insert into PERSON(firstname, lastname)  
VALUES ('Marek', 'Adamowski');  
  
insert into PERSON(firstname, lastname)  
VALUES ('Adam', 'Markowski');  
  
insert into PERSON(firstname, lastname)  
VALUES ('Filip', 'Maciejewski');  
  
insert into PERSON(firstname, lastname)  
VALUES ('Maciej', 'Filipowski');  
  
-- 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, 'C');  
  
insert into reservation(trip\_id, person\_id, status)  
values (2, 4, 'C');  
  
insert into reservation(trip\_id, person\_id, status)  
values (2, 7, 'N');  
  
insert into reservation(trip\_id, person\_id, status)  
values (2, 9, 'P');  
  
-- trip 3  
insert into reservation(trip\_id, person\_id, status)  
values (3, 4, 'P');  
  
-- trip 4  
insert into reservation(trip\_id, person\_id, status)  
values (4, 6, 'P');  
  
insert into reservation(trip\_id, person\_id, status)  
values (4, 9, 'C');  
  
insert into reservation(trip\_id, person\_id, status)  
values (4, 3, 'N');

1. Tworzenie widoków. Należy przygotować kilka widoków ułatwiających dostęp do danych. Należy zwrócić uwagę na strukturę kodu (należy unikać powielania kodu)
2. Reservations(country,trip\_date,trip\_name, firstname, lastname,reservation\_id,status)
3. Trips(country,trip\_date, trip\_name,no\_places, no\_available\_places)
4. AvailableTrips(country,trip\_date, trip\_name,no\_places, no\_available\_places)

Proponowany zestaw widoków można rozbudować wedle uznania/potrzeb

- np. można dodać nowe/pomocnicze widoki

- np. można zmienić def. widoków, dodając nowe/potrzebne pola

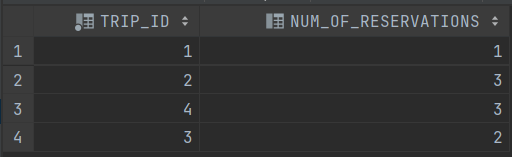
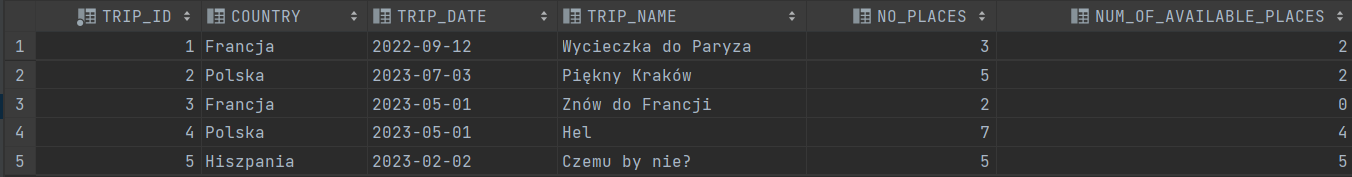
### Komentarz

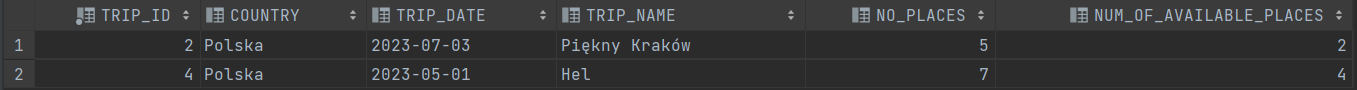
Widoki zostały przeze mnie lekko zmodyfikowane (Reservations zawiera dodatkowo trip\_id oraz person\_id), gdyż w takiej formie można je skutecznie zastosować w procedurach z zadania 4. Dodatkowo stworzyłem też widok pomocniczy, który zlicza liczbę rezerwacji dla każdej wycieczki.

### Kod

CREATE OR REPLACE VIEW Reservations (trip\_id, country, trip\_date, trip\_name, person\_id, firstname, lastname, reservation\_id, status) AS  
select T.TRIP\_ID, T.COUNTRY, T.TRIP\_DATE, T.TRIP\_NAME, P.PERSON\_ID, P.FIRSTNAME, P.LASTNAME, R.RESERVATION\_ID, R.STATUS  
from RESERVATION R  
 join TRIP T on R.TRIP\_ID = T.TRIP\_ID  
 join PERSON P on P.PERSON\_ID = R.PERSON\_ID;



/\* auxiliary view \*/  
CREATE OR REPLACE VIEW ActiveReservationsByTrips AS  
select R.TRIP\_ID, *count*(R.RESERVATION\_ID) NUM\_OF\_RESERVATIONS  
from RESERVATION R  
where R.STATUS != 'C'  
group by R.TRIP\_ID;  
  
  
CREATE OR REPLACE VIEW Trips (trip\_id, country, trip\_date, trip\_name, no\_places, num\_of\_available\_places) AS  
select T.TRIP\_ID, T.COUNTRY, T.TRIP\_DATE, T.TRIP\_NAME, T.MAX\_NO\_PLACES, (T.MAX\_NO\_PLACES - *NVL*(MR.NUM\_OF\_RESERVATIONS, 0))  
from TRIP T  
 left join ActiveReservationsByTrips MR on T.TRIP\_ID = MR.TRIP\_ID;  
  
  
CREATE OR REPLACE VIEW AvailableTrips (trip\_id, country, trip\_date, trip\_name, no\_places, num\_of\_available\_places) AS  
select T.trip\_id, T.COUNTRY, T.TRIP\_DATE, T.TRIP\_NAME, T.NO\_PLACES, T.NUM\_OF\_AVAILABLE\_PLACES  
from TRIPS T  
where num\_of\_available\_places > 0  
 and *current\_date* < T.TRIP\_DATE;



1. Tworzenie procedur/funkcji pobierających dane. Podobnie jak w poprzednim przykładzie należy przygotować kilka procedur ułatwiających dostęp do danych
2. TripParticipants (trip\_id), procedura ma zwracać podobny zestaw danych jak widok Reservations
3. PersonReservations(person\_id), procedura ma zwracać podobny zestaw danych jak widok Reservations
4. AvailableTrips(country, date\_from, date\_to)

Procedury/funkcje powinny zwracać tabelę/zbiór wynikowy

Należy zwrócić uwagę na kontrolę parametrów (np. jeśli parametrem jest trip\_id to należy sprawdzić czy taka wycieczka istnieje). Podobnie jak w przypadku widoków należy unikać powielania kodu.

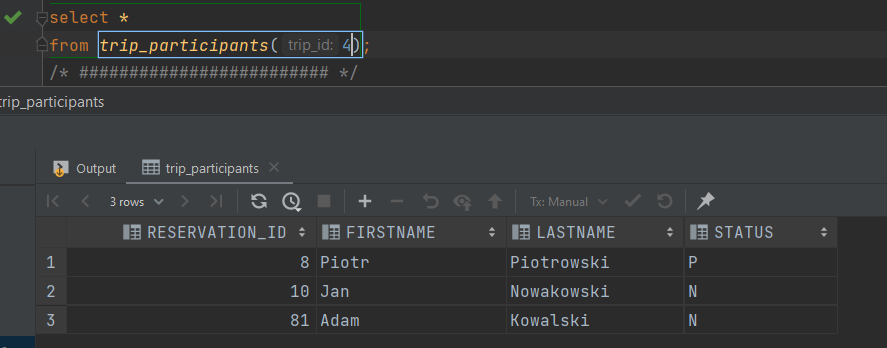
### Komentarz

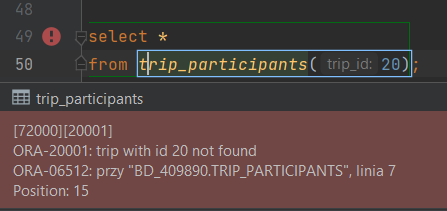
Pomijam wyniki działań funkcji pomocniczych, sprawdzających istnienie różnych elementów w tablicach, gdyż niewiele wniesie to do raportu oraz poprawność działania samych funkcji zwracających dane jest świadectwem poprawnego działania funkcji pomocniczych.

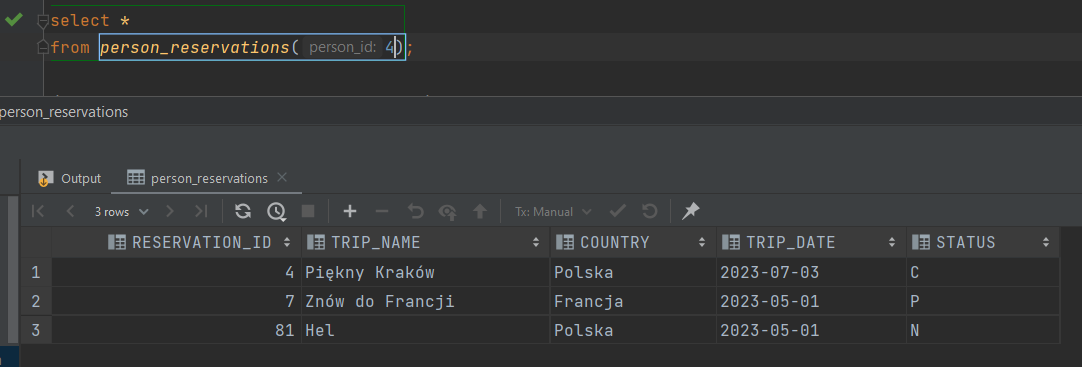
Zmieniłem też nazwę ostatniej funkcji na f\_available\_trips z powodu kolizji z widokiem o takiej samej nazwie

### Kod

create or replace type trip\_participant as OBJECT  
(  
 reservation\_id int,  
 firstname varchar2(50),  
 Lastname varchar2(50),  
 status char  
);  
  
  
create or replace type trip\_participant\_table is table of trip\_participant;  
  
  
CREATE OR REPLACE FUNCTION *trip\_exists*(trip\_id in TRIP.trip\_id%type)  
 RETURN boolean  
AS  
 exist number;  
BEGIN  
 select case  
 when *exists*(select \* from TRIP T where T.TRIP\_ID = *trip\_exists*.trip\_id) then 1  
 else 0  
 end  
 into exist  
 from dual;  
 if exist = 1 then  
 return true;  
 else  
 return false;  
 end if;  
END;  
  
  
CREATE OR REPLACE FUNCTION *trip\_participants*(trip\_id TRIP.TRIP\_ID%type)  
 RETURN trip\_participant\_table  
AS  
 result trip\_participant\_table;  
BEGIN  
 if not *trip\_exists*(trip\_id) then  
 *raise\_application\_error*(-20001, 'trip with id ' || trip\_id || ' not found');  
 end if;  
  
 select trip\_participant(RESERVATION\_ID, FIRSTNAME, LASTNAME, STATUS) bulk collect  
 into result  
 from RESERVATIONS R  
 where R.TRIP\_ID = *trip\_participants*.trip\_id  
 and R.STATUS != 'C';  
 return result;  
END;

  
Gdy wycieczka nie istnieje:



/\* ######################### \*/  
  
create or replace type person\_reservation as OBJECT  
(  
 reservation\_id int,  
 trip\_name varchar2(100),  
 country varchar2(50),  
 trip\_date date,  
 status char  
);  
  
  
create or replace type person\_reservation\_table is table of person\_reservation;  
  
drop function *person\_exists*;  
CREATE OR REPLACE FUNCTION *person\_exists*(person\_id in PERSON.PERSON\_ID%type)  
 RETURN boolean  
AS  
 exist number;  
BEGIN  
 select case  
 when *exists*(select \* from PERSON P where P.PERSON\_ID = *person\_exists*.person\_id) then 1  
 else 0  
 end  
 into exist  
 from dual;  
 if exist = 1 then  
 return true;  
 else  
 return false;  
 end if;  
END;  
  
  
CREATE OR REPLACE FUNCTION *person\_reservations*(person\_id number)  
 RETURN person\_reservation\_table  
AS  
 result person\_reservation\_table;  
BEGIN  
 if not *person\_exists*(*person\_reservations*.person\_id) then  
 *raise\_application\_error*(-20001, 'person with id ' || person\_id || ' not found');  
 end if;  
  
 select person\_reservation(R.RESERVATION\_ID, R.TRIP\_NAME, R.COUNTRY, R.TRIP\_DATE, R.STATUS) bulk collect  
 into result  
 from RESERVATIONS R  
 where R.PERSON\_ID = *person\_reservations*.person\_id;  
 return result;  
END;  
  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
  
create or replace type available\_trips as OBJECT  
(  
 country varchar2(50),  
 trip\_date date,  
 trip\_name varchar2(100),  
 no\_places number,  
 no\_available\_places number  
);  
  
  
create or replace type available\_trips\_table is table of available\_trips;  
  
/\* named with f\_... because there already exists a view with the same name \*/  
  
CREATE OR REPLACE FUNCTION *f\_available\_trips*(country COUNTRY.COUNTRY\_NAME%type, date\_from date, date\_to date)  
 RETURN available\_trips\_table  
AS  
 result available\_trips\_table;  
BEGIN  
 if not date\_from < date\_to then  
 *raise\_application\_error*(-20001, 'ending date cannot be before starting date');  
 end if;  
  
 select available\_trips(AvTr.country,  
 AvTr.trip\_date,  
 AvTr.trip\_name,  
 AvTr.no\_places,  
 AvTr.num\_of\_available\_places) bulk collect  
 into result  
 from AVAILABLETRIPS AvTr  
 where date\_from < TRIP\_DATE  
 and TRIP\_DATE < date\_to  
 and AvTr.COUNTRY = *f\_available\_trips*.country;  
 return result;  
END;

Istniejące wycieczki:

Obraz zawierający tekst

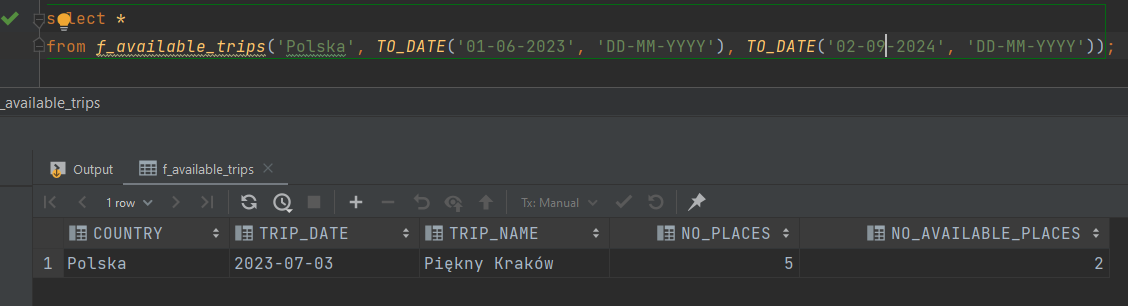
Opis wygenerowany automatycznie

Wynik pierwszego zapytania:

Obraz zawierający tekst, zrzut ekranu, monitor, blisko/zamknięte

Opis wygenerowany automatycznie

Komentarz to wyniku: Żadna wycieczka się nie wyświetliła,gdyż pierwsza nie mieśi się w przedziale czasowym, a na drugą nie ma już miejsc

Wynik drugiego zapytania:

Komentarz do wyniku: Pojawiła się tylko wycieczka o trip\_id 2, gdyż mieści się ona w przedziale czasowym

1. Tworzenie procedur modyfikujących dane. Należy przygotować zestaw procedur pozwalających na modyfikację danych oraz kontrolę poprawności ich wprowadzania
2. AddReservation(trip\_id, person\_id), procedura powinna kontrolować czy wycieczka jeszcze się nie odbyła, i czy sa wolne miejsca
3. ModifyReservationStatus(reservation\_id, status), procedura kontrolować czy możliwa jest zmiana statusu, np. zmiana statusu już anulowanej wycieczki (przywrócenie do stanu aktywnego nie zawsze jest możliwa – może już nie być miejsc)
4. ModifyNoPlaces(trip\_id, no\_places), nie wszystkie zmiany liczby miejsc są dozwolone, nie można zmniejszyć liczby miejsc na wartość poniżej liczby zarezerwowanych miejsc

Należy rozważyć użycie transakcji

Należy zwrócić uwagę na kontrolę parametrów (np. jeśli parametrem jest trip\_id to należy sprawdzić czy taka wycieczka istnieje, jeśli robimy rezerwację to należy sprawdzać czy są wolne miejsca itp..)

### Komentarz

Dodałem atrybuty trip\_id w widokach Trips oraz AvailableTrips, aby można było z nich skorzystać w procedurze add\_reservation. Stworzyłem też parę funkcji pomocniczych.

### Kod

CREATE OR REPLACE FUNCTION *trip\_is\_available*(trip\_id TRIP.trip\_id%type)  
 RETURN boolean  
AS  
 exist number;  
BEGIN  
 select case  
 when *exists*(select \* from AVAILABLETRIPS AvTr where AvTr.TRIP\_ID = *trip\_is\_available*.trip\_id)  
 then 1  
 else 0  
 end  
 into exist  
 from dual;  
 if exist = 1 then  
 return true;  
 else  
 return false;  
 end if;  
END;  
  
  
CREATE OR REPLACE FUNCTION *person\_has\_trip\_reservation*(person\_id PERSON.PERSON\_ID%type, trip\_id TRIP.trip\_id%type)  
 RETURN boolean  
AS  
 exist number;  
BEGIN  
 select case  
 when *exists*(select \*  
 from RESERVATION R  
 where R.TRIP\_ID = *person\_has\_trip\_reservation*.trip\_id  
 and R.PERSON\_ID = *person\_has\_trip\_reservation*.person\_id)  
 then 1  
 else 0  
 end  
 into exist  
 from dual;  
 if exist = 1 then  
 return true;  
 else  
 return false;  
 end if;  
END;  
  
  
create or replace procedure *add\_reservation*(trip\_id TRIP.TRIP\_ID%type, person\_id PERSON.PERSON\_ID%type)  
 is  
 reservation\_id RESERVATION.RESERVATION\_ID%type;  
begin  
 if not *trip\_exists*(trip\_id) then  
 *raise\_application\_error*(-20001, 'trip with id ' || trip\_id || ' doesn''t exist');  
 elsif not *PERSON\_EXISTS*(person\_id) then  
 *raise\_application\_error*(-20002, 'person wit id ' || person\_id || ' doesn''t exist');  
 elsif not *trip\_is\_available*(trip\_id) then  
 *raise\_application\_error*(-20003, 'trip with id ' || trip\_id || ' is not available');  
 elsif *person\_has\_trip\_reservation*(person\_id, trip\_id) then  
 *raise\_application\_error*(-20004,  
 'person with id ' || person\_id || ' already has a reservation for trip with id ' ||  
 trip\_id);  
 end if;

insert into RESERVATION (TRIP\_ID, PERSON\_ID, STATUS)  
 VALUES (TRIP\_ID, PERSON\_ID, 'N') returning RESERVATION\_ID into RESERVATION\_ID;  
  
 *add\_log*(reservation\_id, 'N');  
end;

Pokazuję wyniki:

Available trips:

Obraz zawierający tekst

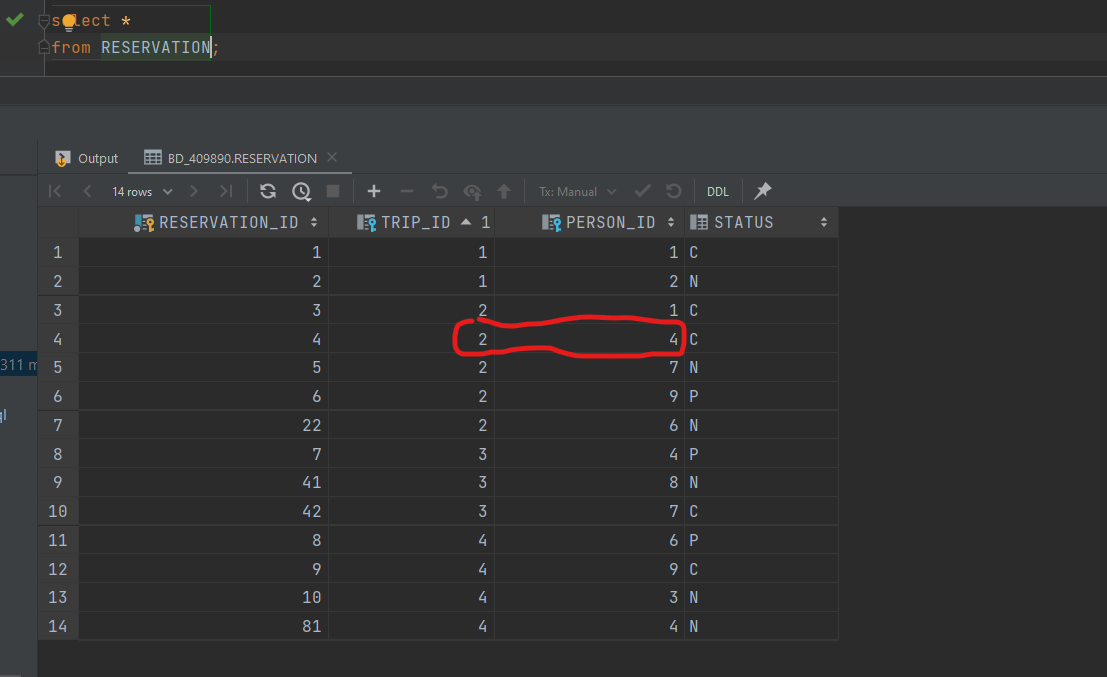
Opis wygenerowany automatycznie

Próba dodania rezerwacji na niedostępną wycieczkę:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Rezerwacje:



Próba dodania drugiej rezerwacji danej osoby na tę samą wycieczkę:

Obraz zawierający tekst

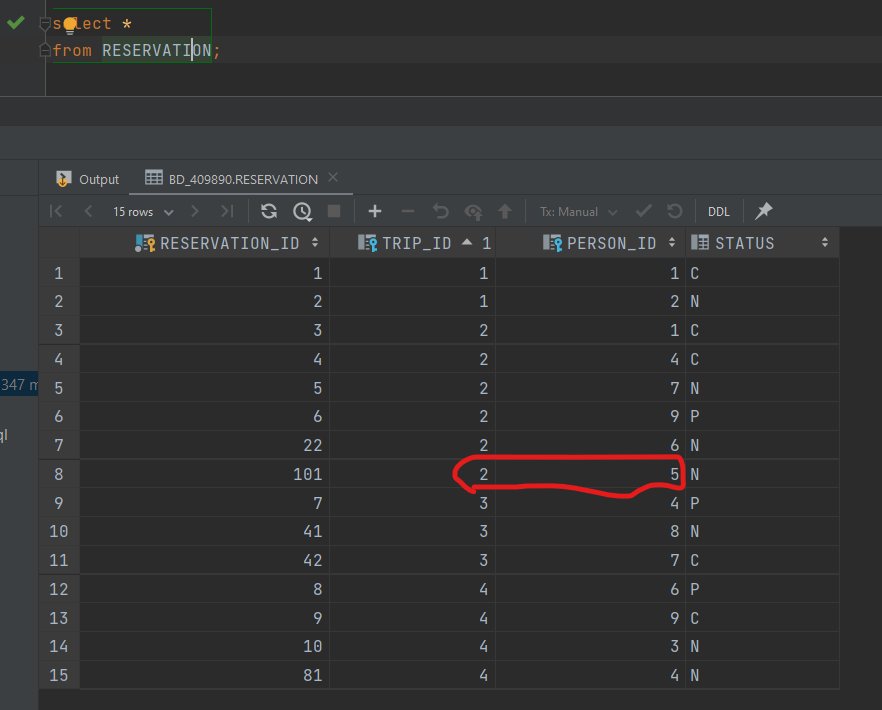
Opis wygenerowany automatycznie

Poprawne wstawienie:

Obraz zawierający tekst, zrzut ekranu, ekran

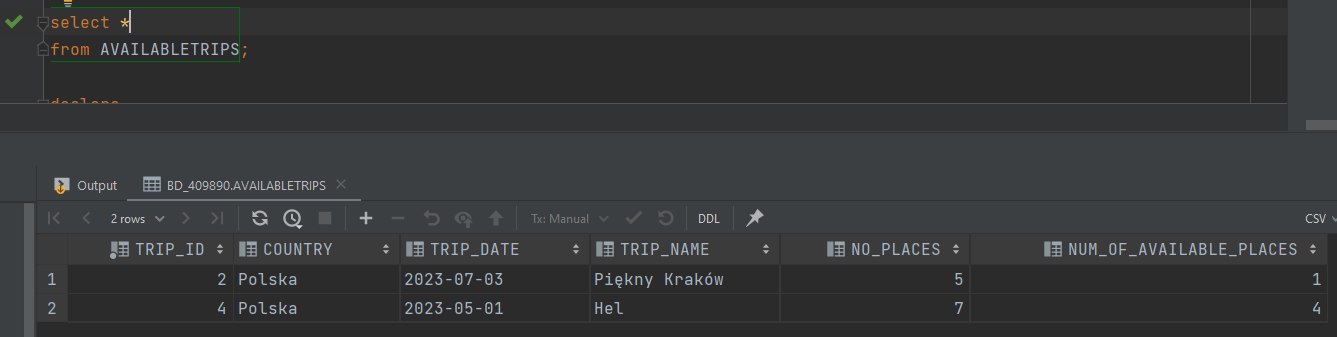
Opis wygenerowany automatycznie

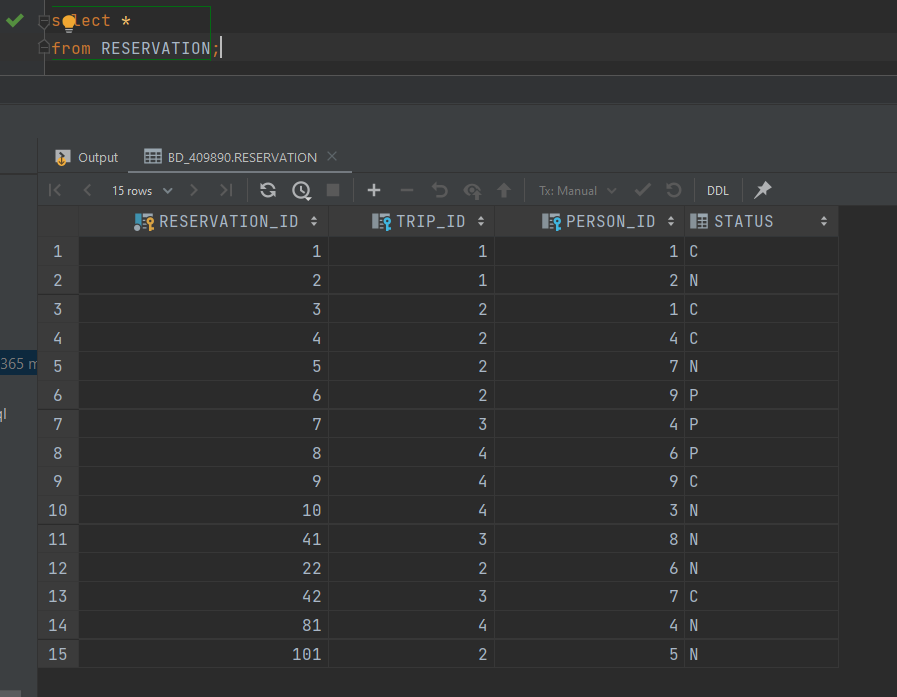
Efekt:



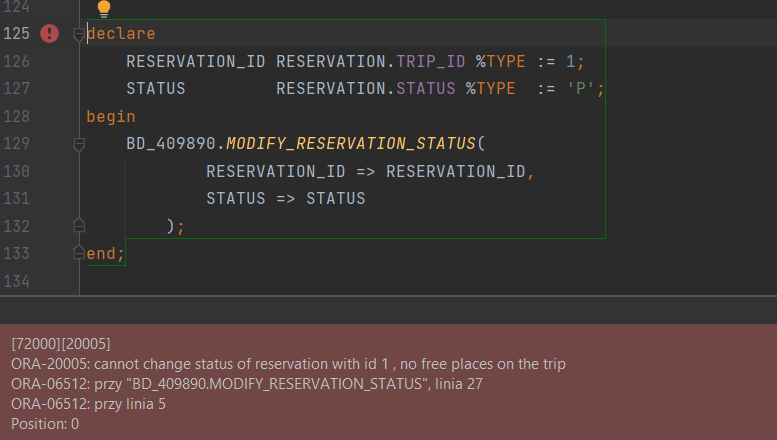
CREATE OR REPLACE FUNCTION *reservation\_exists*(reservation\_id RESERVATION.RESERVATION\_ID%type)  
 RETURN boolean  
AS  
 exist number;  
BEGIN  
 select case  
 when *exists*(select \*  
 from RESERVATION R  
 where R.RESERVATION\_ID = *reservation\_exists*.reservation\_id) then 1  
 else 0  
 end  
 into exist  
 from dual;  
 if exist = 1 then  
 return true;  
 else  
 return false;  
 end if;  
END;  
  
  
CREATE OR REPLACE PROCEDURE *modify\_reservation\_status*(reservation\_id RESERVATION.TRIP\_ID%type,  
 status RESERVATION.STATUS%type)  
 is  
 current\_status RESERVATION.STATUS%type;  
 trip\_id RESERVATION.TRIP\_ID%type;  
begin  
 if not *reservation\_exists*(reservation\_id) then  
 *raise\_application\_error*(-20001, 'reservation with id ' || reservation\_id || ' doesn''t exist');  
 end if;  
  
 select R.STATUS, R.TRIP\_ID  
 into current\_status, trip\_id  
 from RESERVATION R  
 where R.RESERVATION\_ID = *modify\_reservation\_status*.reservation\_id;  
  
 if current\_status = status then  
 return;  
 elsif current\_status = 'N' or  
 current\_status = 'P' or  
 (current\_status = 'C' and *trip\_is\_available*(trip\_id)) then  
 update RESERVATION R  
 set R.STATUS = *modify\_reservation\_status*.status  
 where R.RESERVATION\_ID = *modify\_reservation\_status*.RESERVATION\_ID;  
 else  
 *raise\_application\_error*(-20005, 'cannot change status of reservation with id ' || reservation\_id ||  
 ' , no free places on the trip');  
 end if;  
end;

Dostępne wycieczki przed:

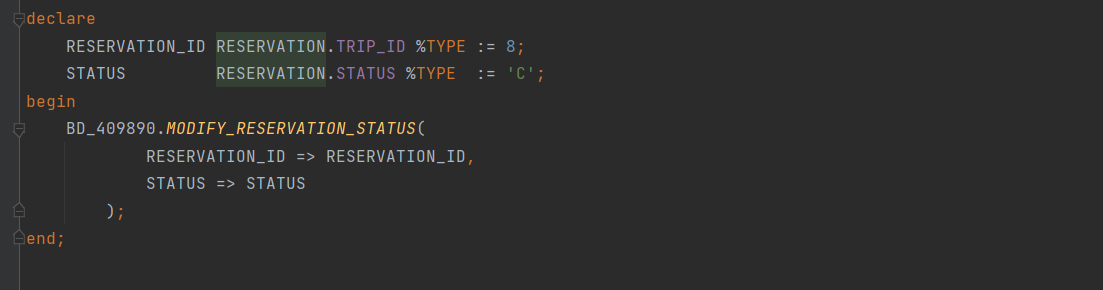


Wszystkie rezerwacje przed:

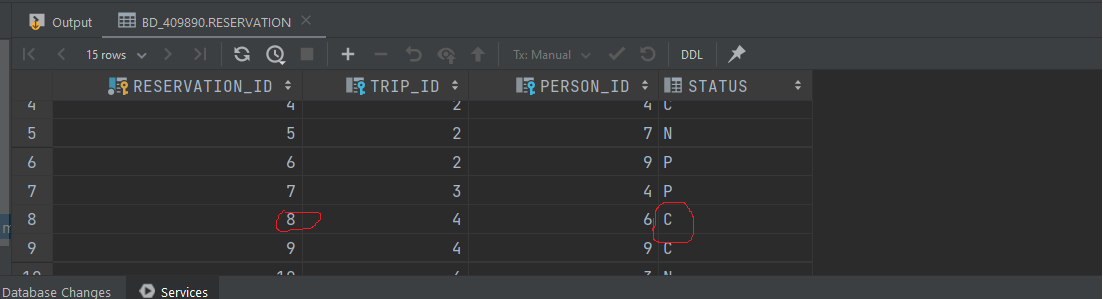
Próba zmiany statusu wycieczki z ‘C’ na ‘P’ w wycieczce bez wolnych miejsc lub wycieczce, która już się odbyła:



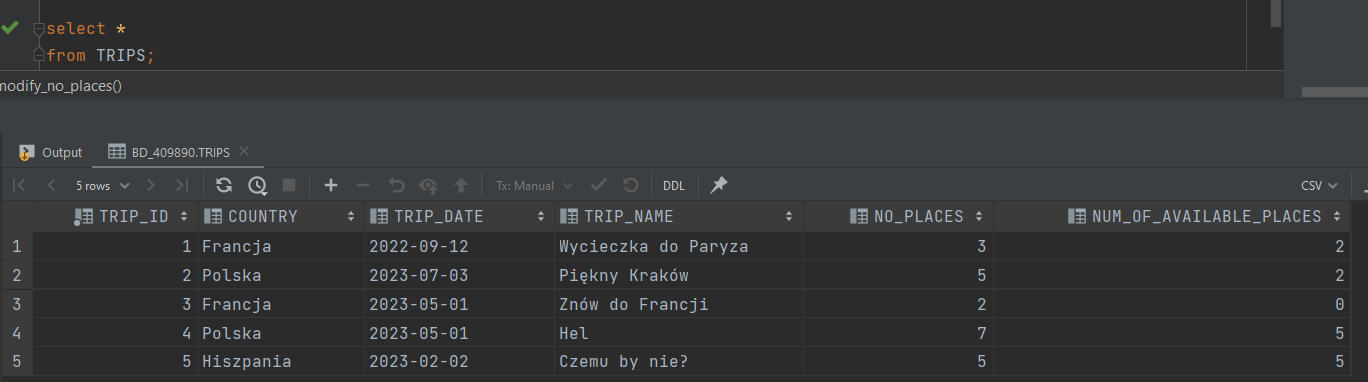
Poprawna zmiana:



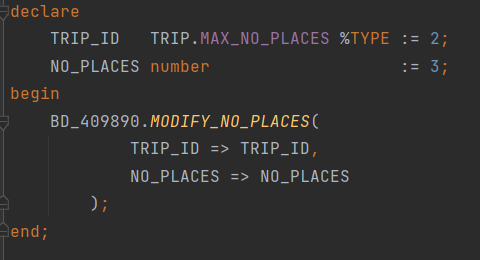
I jej efekt:



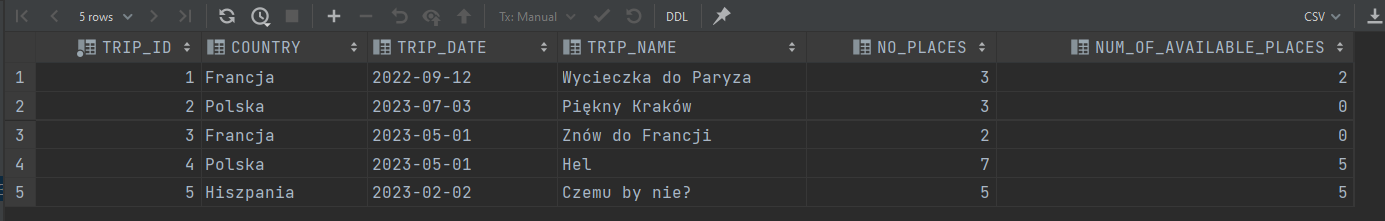
create or replace procedure *modify\_no\_places*(trip\_id TRIP.MAX\_NO\_PLACES%type, no\_places number)  
 is  
 curr\_no\_places number;  
 curr\_num\_of\_available\_places number;  
begin  
 if not *TRIP\_EXISTS*(trip\_id) then  
 *raise\_application\_error*(-20001, 'trip with id ' || trip\_id || ' doesn''t exist');  
 end if;  
  
 select T.NO\_PLACES, NUM\_OF\_AVAILABLE\_PLACES  
 into curr\_no\_places, curr\_num\_of\_available\_places  
 from TRIPS T  
 where T.TRIP\_ID = *modify\_no\_places*.trip\_id;  
  
 /\* we can always increase number of places on a trip \*/  
 /\* we can lower number of places only by up to the number of currently available places \*/  
 if no\_places > curr\_no\_places or curr\_no\_places - no\_places <= curr\_num\_of\_available\_places then  
 update TRIP  
 set MAX\_NO\_PLACES = no\_places  
 where TRIP\_ID = *modify\_no\_places*.TRIP\_ID;  
 else  
 *raise\_application\_error*(-20002,  
 'cannot set number of available places to ' || no\_places || ' for trip with id ' ||  
 trip\_id ||  
 '; otherwise there would be more reservations than places');  
 end if;  
end;

Wycieczki przed:

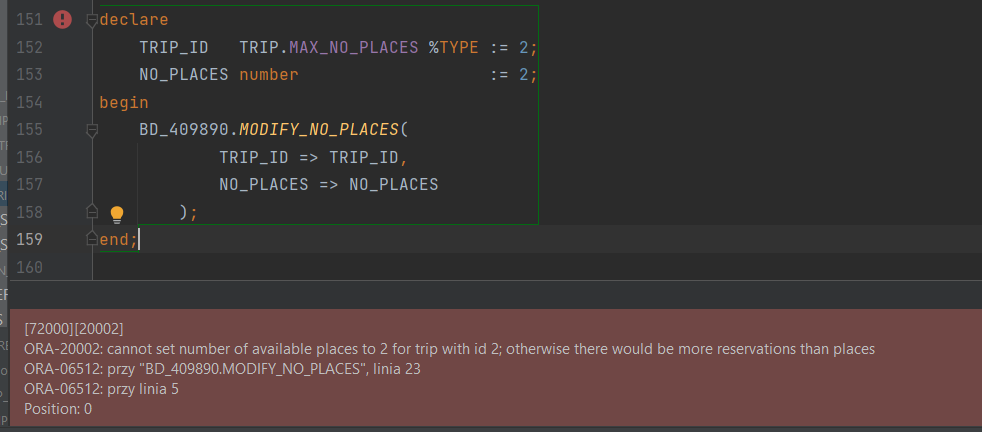
Zmiana liczby miejsc w wycieczce o trip\_id 2:



Efekt:



Wywołanie niepoprawne:



1. Dodajemy tabelę dziennikującą zmiany statusu rezerwacji

Log(id, reservation\_id, date, status)

Należy zmienić warstwę procedur modyfikujących dane tak aby dopisywały informację do dziennika

### Kod

Kod tworzący tablicę dziennikową:

create table log  
(  
 log\_id int generated always as identity not null,  
 reservation\_id int not null,  
 log\_date date not null,  
 status char(1),  
 constraint log\_pk primary key ( log\_id ) enable  
);  
  
alter table log  
add constraint log\_chk1 check  
(status in ('N','P','C')) enable;  
  
alter table log  
add constraint log\_fk1 foreign key  
( reservation\_id ) references reservation ( reservation\_id ) enable;

Funkcja dodająca wpis to tabeli Log

create or replace procedure *add\_log*(reservation\_id RESERVATION.RESERVATION\_ID%type, status RESERVATION.STATUS%type)  
 is  
begin  
 if not *reservation\_exists*(reservation\_id) then  
 *raise\_application\_error*(-20006, 'reservation with id ' || reservation\_id || ' doesn''t exist');  
 end if;  
  
 insert into LOG (RESERVATION\_ID, LOG\_DATE, STATUS)  
 VALUES (*add\_log*.reservation\_id, *current\_date*, *add\_log*.status);  
end;

Zmodyfikowany / dodany kod poprzednio napisanych procedur:

w add\_reservation:

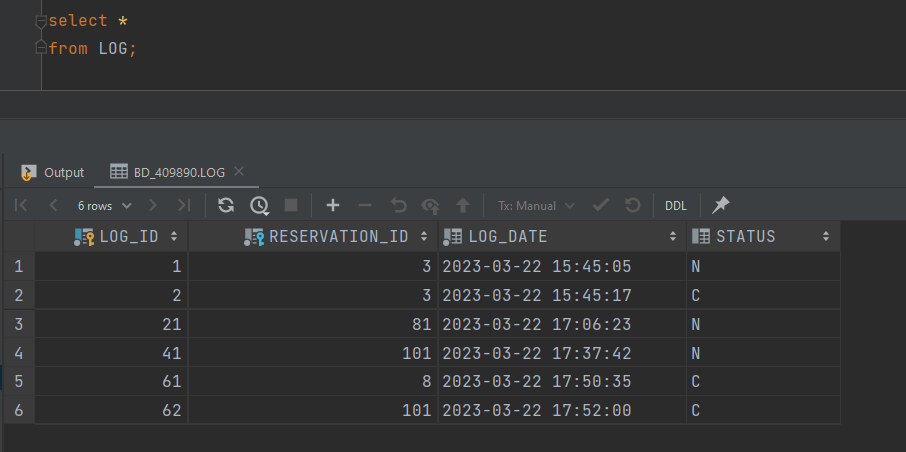
insert into RESERVATION (TRIP\_ID, PERSON\_ID, STATUS)  
VALUES (TRIP\_ID, PERSON\_ID, 'N') returning RESERVATION\_ID into RESERVATION\_ID;  
  
*add\_log*(reservation\_id, 'N');

w modify\_reservation\_status:

update RESERVATION R  
set R.STATUS = *modify\_reservation\_status*.status  
where R.RESERVATION\_ID = *modify\_reservation\_status*.RESERVATION\_ID;  
  
*add\_log*(*modify\_reservation\_status*.reservation\_id, *modify\_reservation\_status*.status);

Tabela Log po paru wywołanych procedurach:

Stan przed:



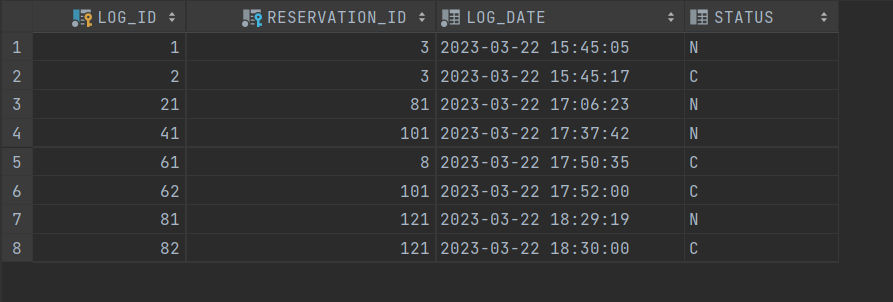
Wywołania:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

Stan po:

1. Zmiana strategii zapisywania do dziennika rezerwacji. Realizacja przy pomocy triggerów

Należy wprowadzić zmianę, która spowoduje, że zapis do dziennika rezerwacji będzie realizowany przy pomocy trierów

triger obsługujący dodanie rezerwacji

triger obsługujący zmianę statusu

triger zabraniający usunięcia rezerwacji

Oczywiście po wprowadzeniu tej zmiany należy uaktualnić procedury modyfikujące dane. Najlepiej to zrobić tworząc nowe wersje (np. z sufiksem 2)

### Komentarz

Procedury add\_reservation\_2 i modify\_reservation\_status\_2 różnią się od swoich poprzedników tylko tym, że nie wywołują w swoim ciele procedury add\_log. Zadanie to przejęły triggery.

### Kod

create or replace trigger reservation\_added  
 after insert or update  
 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 reservation\_removed  
 before delete  
 on RESERVATION  
begin  
 *raise\_application\_error*(-20001, 'you cannot completely remove any reservation from database;' ||  
 'if you want to cancel a reservation, change it''s status to "C" (cancelled)');  
end;

Wywołanie:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Efekt:

Obraz zawierający tekst, elektronika, blisko/zamknięte

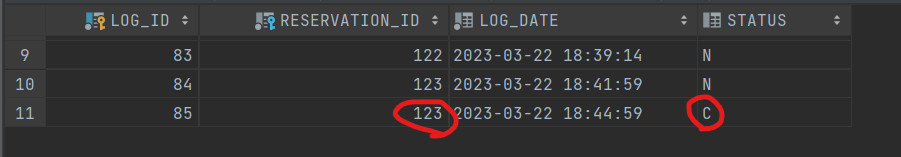
Opis wygenerowany automatycznie

Wywołanie:

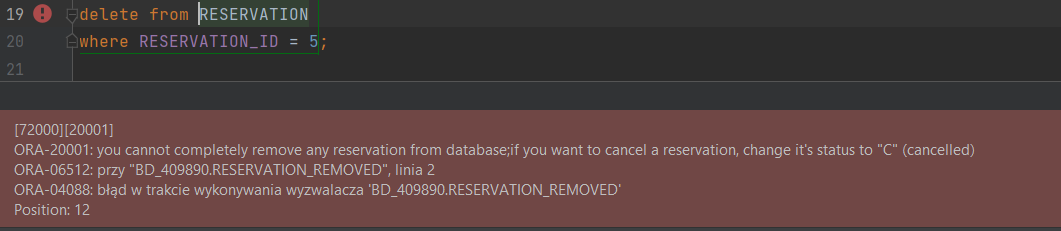
Obraz zawierający tekst

Opis wygenerowany automatycznie

Efekt:



Próba usunięcia rezerwacji z tabeli:



1. Zmiana strategii kontroli dostępności miejsc. Realizacja przy pomocy triggerów

Należy wprowadzić zmianę, która spowoduje, że zapis do dziennika rezerwacji będzie realizowany przy pomocy trierów

triger obsługujący dodanie rezerwacji

triger obsługujący zmianę statusu

Oczywiście po wprowadzeniu tej zmiany należy uaktualnić procedury modyfikujące dane. Najlepiej to zrobić tworząc nowe wersje (np. z sufiksem 3)

### Kod

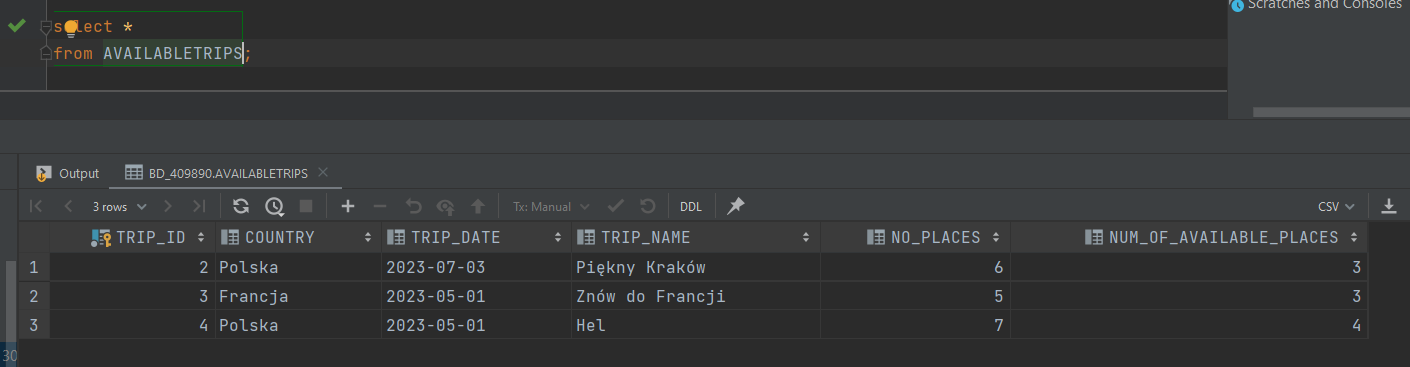
# Triggery

create or replace trigger about\_to\_add\_reservation  
 before insert  
 on RESERVATION  
 for each row  
begin  
 if not *trip\_is\_available*(:NEW.TRIP\_ID) then  
 *raise\_application\_error*(-20003, 'trip with id ' || :NEW.TRIP\_ID || ' is not available');  
 elsif *person\_has\_trip\_reservation*(:NEW.PERSON\_ID, :NEW.TRIP\_ID) then  
 *raise\_application\_error*(-20004,  
 'person with id ' || :NEW.PERSON\_ID || ' already has a reservation for trip with id ' ||  
 :NEW.TRIP\_ID);  
 end if;  
end;  
  
create or replace trigger about\_to\_modify\_reservation  
 before update  
 on RESERVATION  
 for each row  
begin  
 /\* cannot change reservation id, trip id nor person id \*/  
 if :OLD.RESERVATION\_ID != :NEW.RESERVATION\_ID or  
 :OLD.TRIP\_ID != :NEW.TRIP\_ID or  
 :OLD.PERSON\_ID != :NEW.PERSON\_ID then  
 *raise\_application\_error*(-20002, 'you cannot change reservation id, trip id nor person id of any reservation');  
 end if;  
  
 if :OLD.STATUS = 'C' and not *trip\_is\_available*(:OLD.TRIP\_ID) then  
 *raise\_application\_error*(-20005, 'cannot change status of reservation with id ' || :NEW.RESERVATION\_ID ||  
 ' , no free places on the trip');  
 end if;  
end;

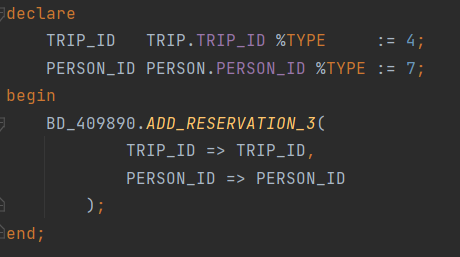
# Procedury w wersji nr. 3

/\* These procedures are modified versions of previous procedures\_2  
 They work together correctly with triggers from triggers\_3 \*/  
  
create or replace procedure *add\_reservation\_3*(trip\_id TRIP.TRIP\_ID%type, person\_id PERSON.PERSON\_ID%type)  
 is  
begin  
 if not *trip\_exists*(trip\_id) then  
 *raise\_application\_error*(-20001, 'trip with id ' || trip\_id || ' doesn''t exist');  
 elsif not *PERSON\_EXISTS*(person\_id) then  
 *raise\_application\_error*(-20002, 'person wit id ' || person\_id || ' doesn''t exist');  
 end if;  
  
 insert into RESERVATION (TRIP\_ID, PERSON\_ID, STATUS)  
 VALUES (TRIP\_ID, PERSON\_ID, 'N');  
end;

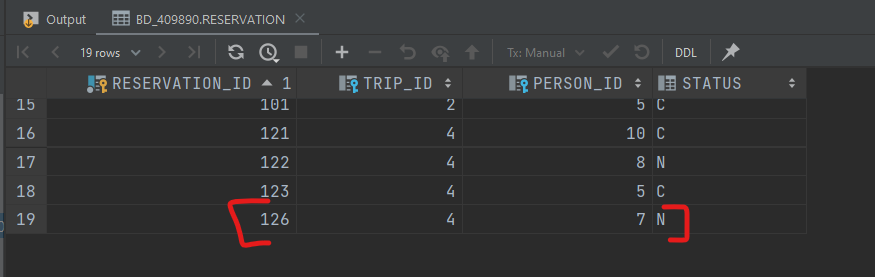
Przed:



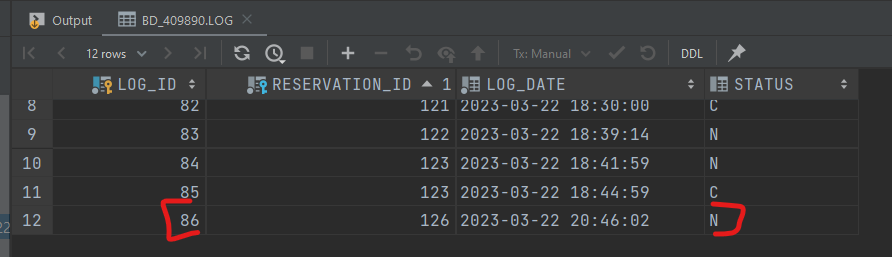
Wykonanie:



Efekt w Reservation:

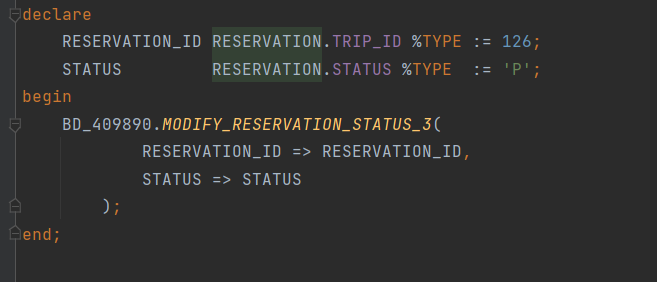


Efekt w Log:

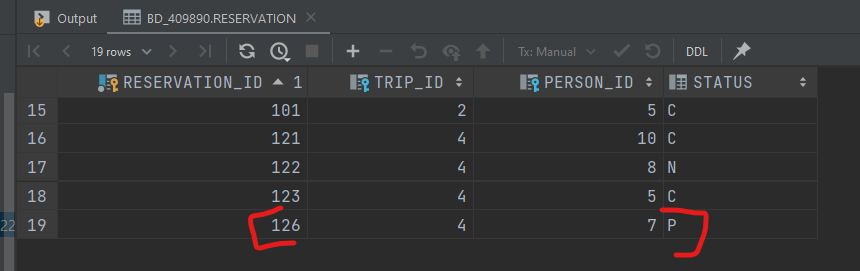


CREATE OR REPLACE PROCEDURE *modify\_reservation\_status\_3*(reservation\_id RESERVATION.TRIP\_ID%type,  
 status RESERVATION.STATUS%type)  
 is  
begin  
 if not *reservation\_exists*(reservation\_id) then  
 *raise\_application\_error*(-20001, 'reservation with id ' || reservation\_id || ' doesn''t exist');  
 end if;  
  
 update RESERVATION R  
 set R.STATUS = *modify\_reservation\_status\_3*.status  
 where R.RESERVATION\_ID = *modify\_reservation\_status\_3*.RESERVATION\_ID;  
end;

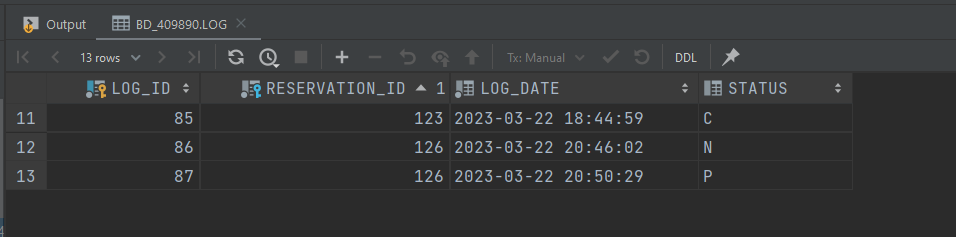
Wykonanie:



Efekt w Reservation:



Efekt w Log:



1. Zmiana struktury bazy danych, w tabeli wycieczki dodajemy redundantne pole no\_available\_places

Obsługa redundantnego pola (kontrola liczby dostępnych miejsc) w procedurach

Należy zmodyfikować zestaw widoków. Proponuję dodać kolejne widoki (np. z sufiksem 4), które pobierają informację o wolnych miejscach z nowo dodanego pola.

Należy napisać procedurę przelicz która zaktualizuje wartość liczby wolnych miejsc dla już istniejących danych

Należy zmodyfikować warstwę procedur/funkcji pobierających dane, podobnie jak w przypadku widoków.

Należy zmodyfikować procedury wprowadzające dane tak aby korzystały/aktualizowały pole no\_available\_places w tabeli wycieczki

Najlepiej to zrobić tworząc nowe wersje (np. z sufiksem 4)

### Komentarz

Niestety, przez moje niezrozumienie polecenia, w celu przebudowy struktury bazy danych, usunąłem wszystkie dotychczasowe dane. Gdy zdałem sobie sprawę, że powinienem był dodać kolumnę no\_available\_places a następnie wpisać do niej procedurą liczbę dostępnych obecnie miejsc, było już po moim błędzie. Z tego też powodu wypełniłem jeszcze raz bazę różnymi danymi, a następnie (tym razem poprawnie) dodałem żądaną kolumnę i wpisałem do niej dane.

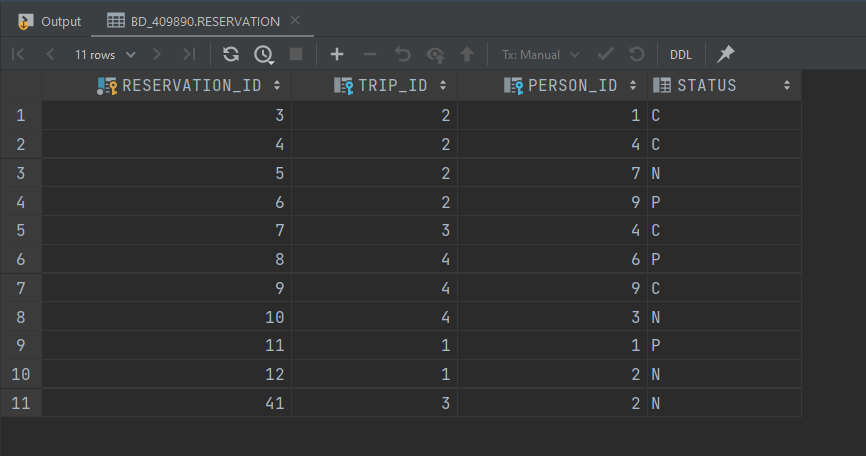
### Kod

# Modyfikacja tablicy:

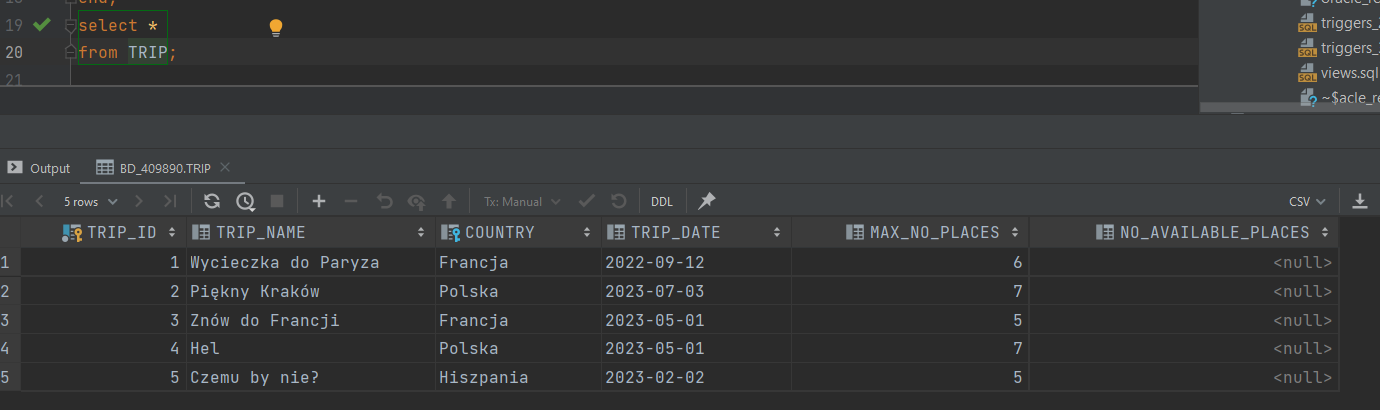
alter table trip  
 add no\_available\_places int;  
  
alter table trip  
 add constraint no\_available\_places\_check1 check  
 ( no\_available\_places >= 0 ) enable;

# Wpisanie wartości do no\_available\_places

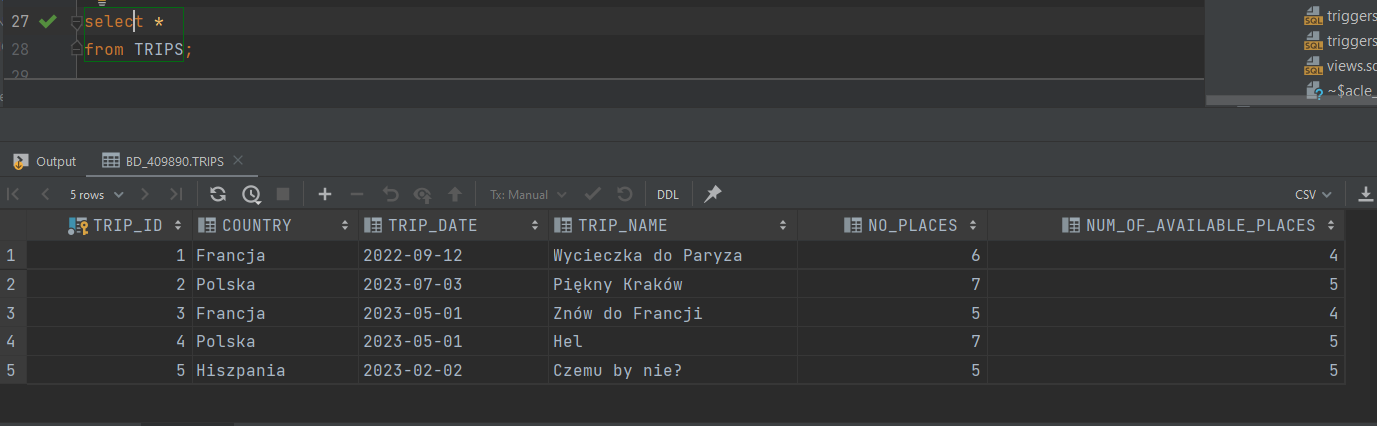
create procedure *calculate\_no\_available\_places* is  
begin  
 update TRIP T  
 set T.no\_available\_places = *NVL*((select NUM\_OF\_AVAILABLE\_PLACES  
 from AVAILABLETRIPS AvTr  
 where AvTr.TRIP\_ID = T.TRIP\_ID), T.MAX\_NO\_PLACES);  
end;

Rezerwacje:

Przed:



Po:



# Widoki

# Komentarz: Widok Trips\_4 uznałem za niepotrzebny, gdyż w obecnej sytuacji pokrywa się on z samą tabelą Trip

CREATE OR REPLACE VIEW Reservations\_4 (trip\_id, country, trip\_date, trip\_name, person\_id, firstname, lastname, reservation\_id, status) AS  
select T.TRIP\_ID, T.COUNTRY, T.TRIP\_DATE, T.TRIP\_NAME, P.PERSON\_ID, P.FIRSTNAME, P.LASTNAME, R.RESERVATION\_ID, R.STATUS  
from RESERVATION R  
 join TRIP T on R.TRIP\_ID = T.TRIP\_ID  
 join PERSON P on P.PERSON\_ID = R.PERSON\_ID;  
  
CREATE OR REPLACE VIEW AvailableTrips\_4 (trip\_id, country, trip\_date, trip\_name, no\_places, num\_of\_available\_places) AS  
select TRIP\_ID, COUNTRY, TRIP\_DATE, TRIP\_NAME, MAX\_NO\_PLACES, NO\_AVAILABLE\_PLACES  
from Trip  
where NO\_AVAILABLE\_PLACES > 0  
 and *current\_date* < TRIP\_DATE;

# Funkcje zwracające dane

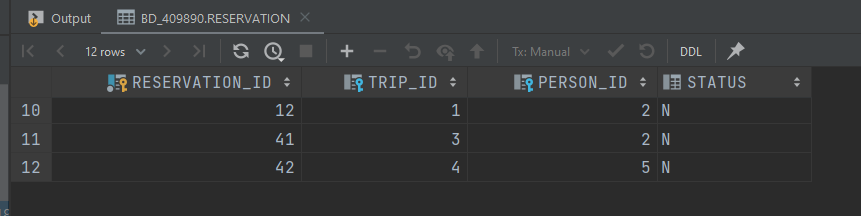
# Zmienił się kod jedynie poniższej funkcji (zmiana tablicy źródłowej)

CREATE OR REPLACE FUNCTION *f\_available\_trips\_4*(country COUNTRY.COUNTRY\_NAME%type, date\_from date, date\_to date)  
 RETURN available\_trips\_table  
AS  
 result available\_trips\_table;  
BEGIN  
 if not date\_from < date\_to then  
 *raise\_application\_error*(-20001, 'ending date cannot be before starting date');  
 end if;  
  
 select available\_trips(T.country,  
 T.trip\_date,  
 T.trip\_name,  
 T.max\_no\_places,  
 T.no\_available\_places) bulk collect  
 into result  
 from TRIP T  
 where date\_from < TRIP\_DATE  
 and TRIP\_DATE < date\_to  
 and T.COUNTRY = *f\_available\_trips\_4*.country;  
 return result;  
END;

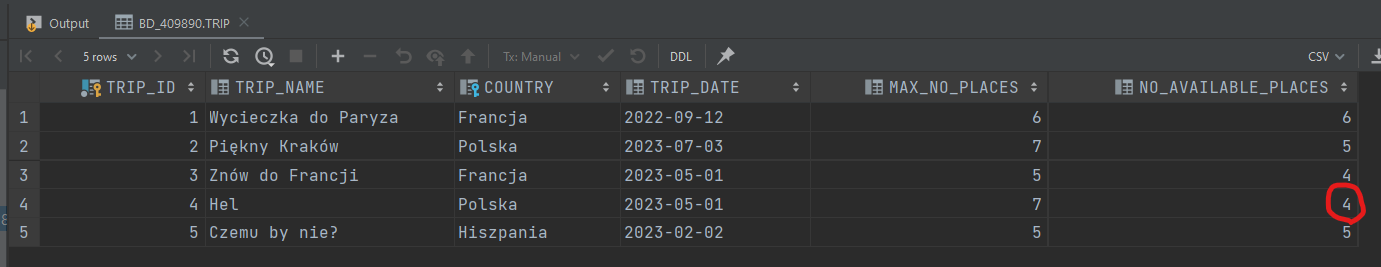
Procedury modyfikujące dane:

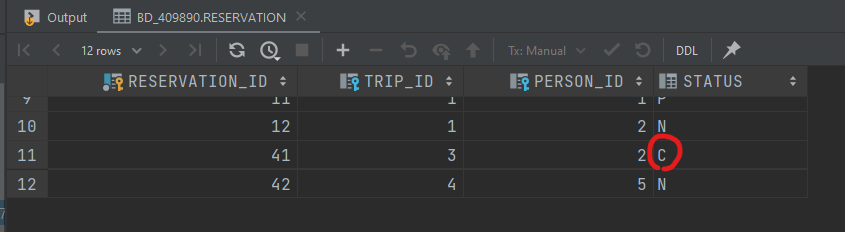
create or replace procedure *add\_reservation\_4*(trip\_id TRIP.TRIP\_ID%type, person\_id PERSON.PERSON\_ID%type)  
 is  
begin  
 if not *trip\_exists*(trip\_id) then  
 *raise\_application\_error*(-20001, 'trip with id ' || trip\_id || ' doesn''t exist');  
 elsif not *PERSON\_EXISTS*(person\_id) then  
 *raise\_application\_error*(-20002, 'person wit id ' || person\_id || ' doesn''t exist');  
 elsif not *trip\_is\_available*(trip\_id) then  
 *raise\_application\_error*(-20003, 'trip with id ' || trip\_id || ' is not available');  
 elsif *person\_has\_trip\_reservation*(person\_id, trip\_id) then  
 *raise\_application\_error*(-20004,  
 'person with id ' || person\_id || ' already has a reservation for trip with id ' ||  
 trip\_id);  
 end if;  
  
 insert into RESERVATION (TRIP\_ID, PERSON\_ID, STATUS)  
 VALUES (TRIP\_ID, PERSON\_ID, 'N');  
  
 update TRIP  
 set NO\_AVAILABLE\_PLACES = NO\_AVAILABLE\_PLACES - 1  
 where TRIP\_ID = *add\_reservation\_4*.TRIP\_ID;  
end;  
  
  
CREATE OR REPLACE PROCEDURE *modify\_reservation\_status\_4*(reservation\_id RESERVATION.TRIP\_ID%type,  
 status RESERVATION.STATUS%type)  
 is  
 current\_status RESERVATION.STATUS%type;  
 trip\_id RESERVATION.TRIP\_ID%type;  
begin  
 if not *reservation\_exists*(reservation\_id) then  
 *raise\_application\_error*(-20001, 'reservation with id ' || reservation\_id || ' doesn''t exist');  
 end if;  
  
 select R.STATUS, R.TRIP\_ID  
 into current\_status, trip\_id  
 from RESERVATION R  
 where R.RESERVATION\_ID = *modify\_reservation\_status\_4*.reservation\_id;  
  
 if current\_status = status then  
 return;  
  
 elsif (current\_status = 'N' or current\_status = 'P') and status = 'C' then  
 update RESERVATION R  
 set R.STATUS = *modify\_reservation\_status\_4*.status  
 where R.RESERVATION\_ID = *modify\_reservation\_status\_4*.RESERVATION\_ID;  
  
 update TRIP  
 set NO\_AVAILABLE\_PLACES = NO\_AVAILABLE\_PLACES + 1  
 where TRIP\_ID = *modify\_reservation\_status\_4*.trip\_id;  
  
 elsif (current\_status = 'N' or current\_status = 'P') and (status = 'N' or status = 'P') then  
 update RESERVATION R  
 set R.STATUS = *modify\_reservation\_status\_4*.status  
 where R.RESERVATION\_ID = *modify\_reservation\_status\_4*.RESERVATION\_ID;  
  
 elsif current\_status = 'C' and *TRIP\_IS\_AVAILABLE*(trip\_id) then  
 update RESERVATION R  
 set R.STATUS = *modify\_reservation\_status\_4*.status  
 where R.RESERVATION\_ID = *modify\_reservation\_status\_4*.RESERVATION\_ID;  
  
 update TRIP  
 set NO\_AVAILABLE\_PLACES = NO\_AVAILABLE\_PLACES - 1  
 where TRIP\_ID = *modify\_reservation\_status\_4*.trip\_id;  
  
 else  
 *raise\_application\_error*(-20005, 'cannot change status of reservation with id ' || reservation\_id ||  
 ' , no free places on the trip');  
 end if;  
end;

Będziemy modyfikować reservację nr. 41. Zmienimy jej status na ‘C’:

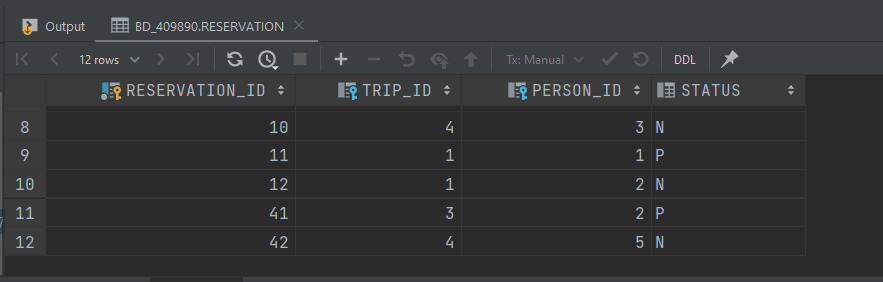


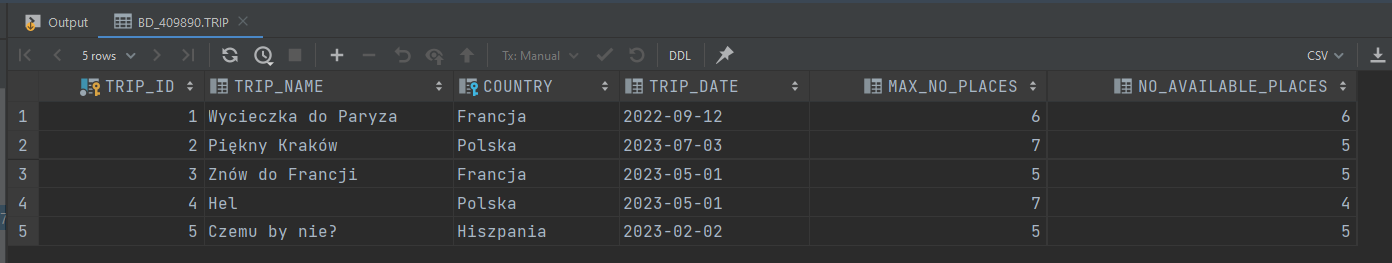
Przed:



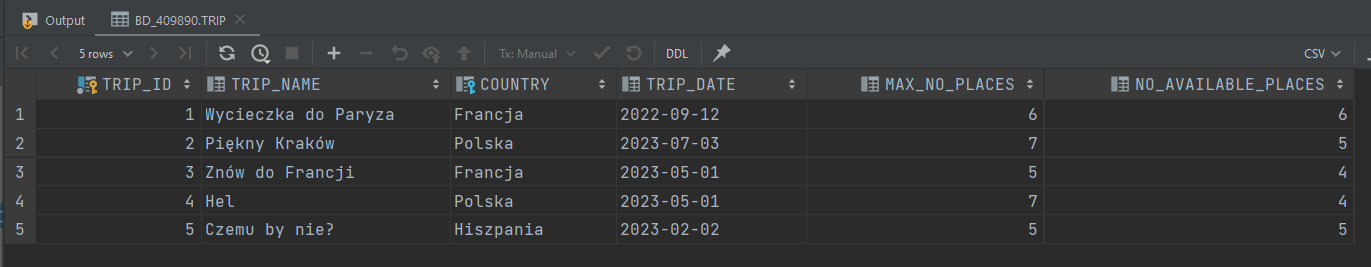


Po:

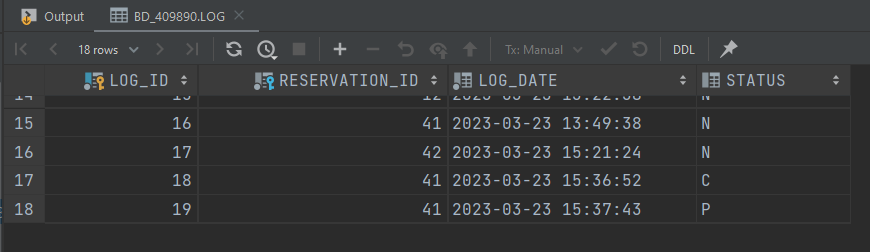




Z powrotem zmieniamy status na ‘N’:



Wszystko zarejestrowała tablica Log:



create or replace procedure *modify\_no\_places\_4*(trip\_id TRIP.MAX\_NO\_PLACES%type, no\_places number)  
 is  
 curr\_no\_places number;  
 curr\_num\_of\_available\_places number;  
begin  
 if not *TRIP\_EXISTS*(trip\_id) then  
 *raise\_application\_error*(-20001, 'trip with id ' || trip\_id || ' doesn''t exist');  
 end if;  
  
 select T.MAX\_NO\_PLACES, NO\_AVAILABLE\_PLACES  
 into curr\_no\_places, curr\_num\_of\_available\_places  
 from TRIP T  
 where T.TRIP\_ID = *modify\_no\_places\_4*.trip\_id;  
  
 /\* we can always increase number of places on a trip \*/  
 /\* we can lower number of places only by up to the number of currently available places \*/  
 if no\_places > curr\_no\_places or curr\_no\_places - no\_places <= curr\_num\_of\_available\_places then  
 update TRIP  
 set MAX\_NO\_PLACES = no\_places, NO\_AVAILABLE\_PLACES = NO\_AVAILABLE\_PLACES + no\_places - curr\_no\_places  
 where TRIP\_ID = *modify\_no\_places\_4*.TRIP\_ID;  
 else  
 *raise\_application\_error*(-20002,  
 'cannot set number of available places to ' || no\_places || ' for trip with id ' ||  
 trip\_id ||  
 '; otherwise there would be more reservations than places');  
 end if;  
end;

1. Zmiana strategii obsługi redundantnego pola no\_available\_places, realizacja przy pomocy trigerów

triger obsługujący dodanie rezerwacji

triger obsługujący zmianę statusu

triger obsługujący zmianę liczby miejsc na poziomie wycieczki

Oczywiście po wprowadzeniu tej zmiany należy uaktualnić procedury modyfikujące dane. Najlepiej to zrobić tworząc nowe wersje (np. z sufiksem 5)

### Kod

# Triggery

create or replace trigger about\_to\_add\_reservation\_5  
 before insert  
 on RESERVATION  
 for each row  
begin  
 if not *trip\_is\_available*(:NEW.TRIP\_ID) then  
 *raise\_application\_error*(-20003, 'trip with id ' || :NEW.TRIP\_ID || ' is not available');  
 elsif *person\_has\_trip\_reservation*(:NEW.PERSON\_ID, :NEW.TRIP\_ID) then  
 *raise\_application\_error*(-20004,  
 'person with id ' || :NEW.PERSON\_ID || ' already has a reservation for trip with id ' ||  
 :NEW.TRIP\_ID);  
 end if;  
  
 update TRIP  
 set NO\_AVAILABLE\_PLACES = NO\_AVAILABLE\_PLACES - 1  
 where TRIP\_ID = :NEW.TRIP\_ID;  
end;  
  
  
create or replace trigger about\_to\_modify\_reservation\_5  
 before update  
 on RESERVATION  
 for each row  
begin  
 /\* cannot change reservation id, trip id nor person id \*/  
 if :OLD.RESERVATION\_ID != :NEW.RESERVATION\_ID or  
 :OLD.TRIP\_ID != :NEW.TRIP\_ID or  
 :OLD.PERSON\_ID != :NEW.PERSON\_ID then  
 *raise\_application\_error*(-20002, 'you cannot change reservation id, trip id nor person id of any reservation');  
 end if;  
  
  
 if :OLD.STATUS = :NEW.STATUS then  
 return;  
  
 elsif (:OLD.STATUS = 'N' or :OLD.STATUS = 'P') and :NEW.STATUS = 'C' then  
 update TRIP  
 set NO\_AVAILABLE\_PLACES = NO\_AVAILABLE\_PLACES + 1  
 where TRIP\_ID = :OLD.TRIP\_ID;  
  
 elsif :OLD.STATUS = 'C' and *TRIP\_IS\_AVAILABLE*(:OLD.TRIP\_ID) then  
 update TRIP  
 set NO\_AVAILABLE\_PLACES = NO\_AVAILABLE\_PLACES - 1  
 where TRIP\_ID = :OLD.TRIP\_ID;  
  
 elsif :OLD.STATUS = 'C' and not *trip\_is\_available*(:OLD.TRIP\_ID) then  
 *raise\_application\_error*(-20005, 'cannot change status of reservation with id ' || :NEW.RESERVATION\_ID ||  
 ' , no free places on the trip');  
 else  
 *raise\_application\_error*(-20006, 'unknown error');  
 end if;  
end;

# Procedury

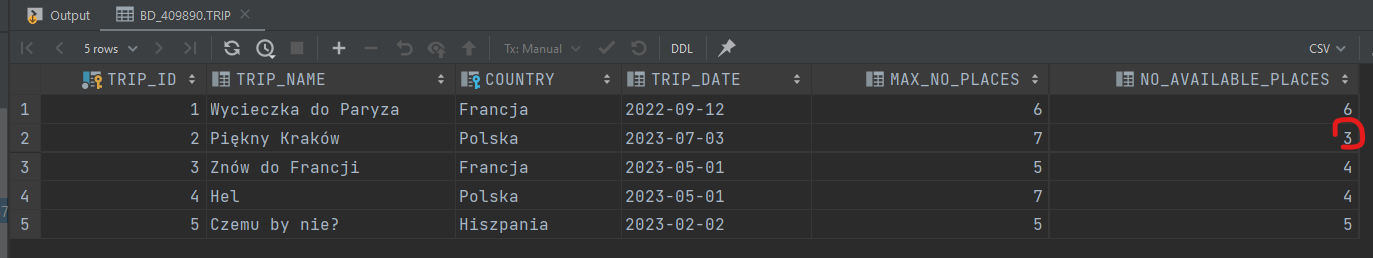
create or replace procedure *add\_reservation\_5*(trip\_id TRIP.TRIP\_ID%type, person\_id PERSON.PERSON\_ID%type)  
 is  
begin  
 if not *trip\_exists*(trip\_id) then  
 *raise\_application\_error*(-20001, 'trip with id ' || trip\_id || ' doesn''t exist');  
 elsif not *PERSON\_EXISTS*(person\_id) then  
 *raise\_application\_error*(-20002, 'person wit id ' || person\_id || ' doesn''t exist');  
 end if;  
  
 insert into RESERVATION (TRIP\_ID, PERSON\_ID, STATUS)  
 VALUES (TRIP\_ID, PERSON\_ID, 'N');  
end;

Przed wstawieniem:

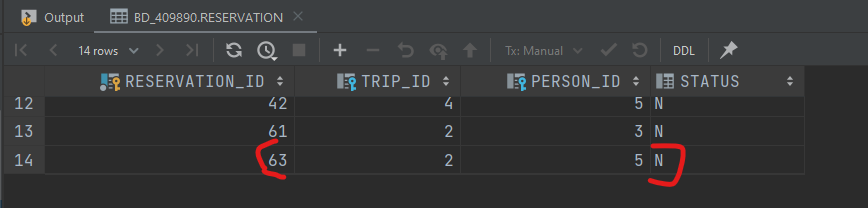
Obraz zawierający tekst

Opis wygenerowany automatycznie

Po wstawieniu:

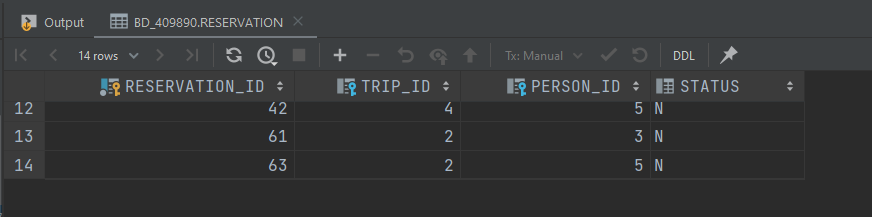


Wstawiona rezerwacja:



CREATE OR REPLACE PROCEDURE *modify\_reservation\_status\_5*(reservation\_id RESERVATION.TRIP\_ID%type,  
 status RESERVATION.STATUS%type)  
 is  
begin  
 if not *reservation\_exists*(reservation\_id) then  
 *raise\_application\_error*(-20001, 'reservation with id ' || reservation\_id || ' doesn''t exist');  
 end if;  
  
 update RESERVATION R  
 set R.STATUS = *modify\_reservation\_status\_5*.status  
 where R.RESERVATION\_ID = *modify\_reservation\_status\_5*.RESERVATION\_ID;  
end;

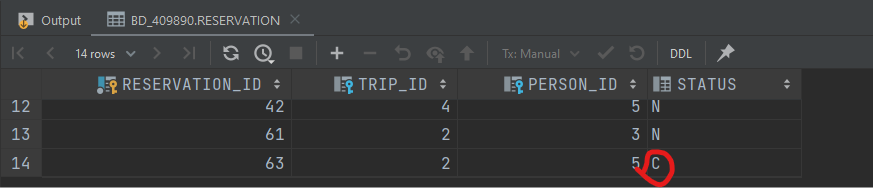
Przed:



Obraz zawierający tekst, zrzut ekranu, elektronika, komputer

Opis wygenerowany automatycznie

Po zmianie statusu na ‘C’:



# Uwagi

Należy przygotować raport z wykonania ćwiczenia. Raport powinien zawierać polecenia SQL (między innymi kod widoków, procedur, …), wynik działania oraz krótki komentarz. Raport należy przesłać w formie pliku PDF.

W raporcie można posłużyć się zrzutami ekranów. Dodatkowo proszę załączyć kod zaimplementowanych widoków/procedur postaci pliku tekstowego (plik tekstowy z rozszerzeniem sql)

Proszę zwrócić uwagę na formatowanie kodu (struktura)

Punktacja za wykonanie pkt. I ćw (wersja dla SZBD Oracle) - max 2pkt