Laboratorium 1,2: Oracle PL/SQL

Adam Mężydło

Spis treści

[Tabele 2](#_Toc131372337)

[Country 2](#_Toc131372338)

[Person 2](#_Toc131372339)

[Trip 2](#_Toc131372340)

[Reservation 3](#_Toc131372341)

[Log 3](#_Toc131372342)

[Przykładowe dane 4](#_Toc131372343)

[Country 4](#_Toc131372344)

[Person 4](#_Toc131372345)

[Trip 5](#_Toc131372346)

[Reservation 5](#_Toc131372347)

[Widoki 6](#_Toc131372348)

[Reservations 6](#_Toc131372349)

[Reservations1 6](#_Toc131372350)

[Trips 7](#_Toc131372351)

[AvailableTrips 7](#_Toc131372352)

[Przykładowe użycia 7](#_Toc131372353)

[Reservations 7](#_Toc131372354)

[Reservations1 8](#_Toc131372355)

[Trips 8](#_Toc131372356)

[AvailableTripsView 8](#_Toc131372357)

[Komentarz 8](#_Toc131372358)

[Funkcje 9](#_Toc131372359)

[TripParticipants 9](#_Toc131372360)

[PersonReservations 10](#_Toc131372361)

[AvailableTrips 11](#_Toc131372362)

[Przykładowe wywołania 12](#_Toc131372363)

[TripParticipants 12](#_Toc131372364)

[PersonReservations 12](#_Toc131372365)

[AvailableTripsView 12](#_Toc131372366)

[AvailableTrips 12](#_Toc131372367)

[Komentarz 12](#_Toc131372368)

# Tabele

## Country

create table country  
(  
 country\_id int generated always as identity not null,  
 country\_name varchar(50) not null,  
 constraint country\_pk primary key (country\_id) enable  
);

## Person

create table person  
(  
 person\_id int generated always as identity not null,  
 firstname varchar(50),  
 lastname varchar(50),  
 constraint person\_pk primary key ( person\_id ) enable  
);

## Trip

create table trip  
(  
 trip\_id int generated always as identity not null,  
 trip\_name varchar(100),  
 country\_id int,  
 trip\_date date,  
 max\_no\_places int,  
 constraint trip\_pk primary key ( trip\_id ) enable  
);  
  
alter table trip  
add constraint country\_fk foreign key  
(country\_id) references country (country\_id) enable;

## Reservation

create table reservation  
(  
 reservation\_id int generated always as identity not null,  
 trip\_id int,  
 person\_id int,  
 status char(1),  
 constraint reservation\_pk primary key ( reservation\_id ) enable  
);  
  
  
alter table reservation  
add constraint reservation\_fk1 foreign key  
( person\_id ) references person ( person\_id ) enable;  
  
alter table reservation  
add constraint reservation\_fk2 foreign key  
( trip\_id ) references trip ( trip\_id ) enable;  
  
alter table reservation  
add constraint reservation\_chk1 check  
(status in ('N','P','C')) enable;

## Log

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

# Przykładowe dane

## Country

insert into country(COUNTRY\_NAME)  
values ('Francja');  
  
insert into country(COUNTRY\_NAME)  
values ('Polska');  
  
insert into country(COUNTRY\_NAME)  
values ('Niemcy');  
  
insert into country(COUNTRY\_NAME)  
values ('Belgia');  
  
insert into country(COUNTRY\_NAME)  
values ('Egipt');

## 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');

## Trip

insert into trip(trip\_name, country\_id, trip\_date, max\_no\_places)  
values ('Lonely trip', 3, *to\_date*('2024-11-01', 'YYYY-MM-DD'), 1);  
  
insert into trip(trip\_name, country\_id, trip\_date, max\_no\_places)  
values ('Auslander', 3, *to\_date*('2023-04-03', 'YYYY-MM-DD'), 3);  
  
insert into trip(trip\_name, country\_id, trip\_date, max\_no\_places)  
values ('W poszukiwaniu Faraona', 5, *to\_date*('2022-05-01', 'YYYY-MM-DD'), 5);  
  
insert into trip(trip\_name, country\_id, trip\_date, max\_no\_places)  
values ('U nas na Podlasiu', 2, *to\_date*('2023-04-06', 'YYYY-MM-DD'), 5);  
  
insert into trip(trip\_name, country\_id, trip\_date, max\_no\_places)  
values ('Disnayland', 1, *to\_date*('2026-12-25', 'YYYY-MM-DD'), 2);

## Reservation

insert into reservation(trip\_id, person\_id, status)  
values (2, 1, 'P');  
  
insert into reservation(trip\_id, person\_id, status)  
values (2, 4, 'C');  
  
insert into reservation(trip\_id, person\_id, status)  
values (3, 4, 'P');  
  
insert into reservation(trip\_id, person\_id, status)  
values (3, 5, 'P');  
  
insert into reservation(trip\_id, person\_id, status)  
values (3, 6, 'C');  
  
insert into reservation(trip\_id, person\_id, status)  
values (4, 1, 'N');  
  
insert into reservation(trip\_id, person\_id, status)  
values (4, 9, 'N');  
  
insert into reservation(trip\_id, person\_id, status)  
values (5, 10, 'P');

# Widoki

## Reservations

create or replace view Reservations  
as  
select c.country\_name,  
 t.trip\_date,  
 t.trip\_name,  
 p.firstname,  
 p.lastname,  
 r.reservation\_id,  
 r.status  
from country c  
 join trip t on c.COUNTRY\_ID = t.COUNTRY\_ID  
 join reservation r on r.TRIP\_ID = t.TRIP\_ID  
 join person p on p.PERSON\_ID = r.PERSON\_ID;

## Reservations1

create or replace view Reservations1  
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  
from country c  
 join trip t on c.COUNTRY\_ID = t.COUNTRY\_ID  
 join reservation r on r.TRIP\_ID = t.TRIP\_ID  
 join person p on p.PERSON\_ID = r.PERSON\_ID;

## Trips

create or replace view Trips  
as  
select c.country\_name,  
 t.TRIP\_DATE,  
 t.TRIP\_NAME,  
 t.MAX\_NO\_PLACES,  
 *nvl*(t.MAX\_NO\_PLACES - (select *count*(rw.RESERVATION\_ID)  
 from reservation rw  
 join trip tw on rw.trip\_id = tw.trip\_id  
 where rw.status != 'C'  
 and t.trip\_id = tw.trip\_id  
 group by rw.Trip\_id), t.MAX\_NO\_PLACES) no\_available\_places  
from trip t  
 join reservation r on t.trip\_id = r.reservation\_id  
 join country c on t.COUNTRY\_ID = c.COUNTRY\_ID;

## Trips1

create or replace view Trips1  
as  
select c.country\_name,  
 t.TRIP\_DATE,  
 t.TRIP\_NAME,  
 t.trip\_id,  
 t.MAX\_NO\_PLACES,  
 *nvl*(t.MAX\_NO\_PLACES - (select *count*(rw.RESERVATION\_ID)  
 from reservation rw  
 join trip tw on rw.trip\_id = tw.trip\_id  
 where rw.status != 'C'  
 and t.trip\_id = tw.trip\_id  
 group by rw.Trip\_id), t.MAX\_NO\_PLACES) no\_available\_places  
from trip t  
 join reservation r on t.trip\_id = r.reservation\_id  
 join country c on t.COUNTRY\_ID = c.COUNTRY\_ID;

## FutureTrips

create or replace view FutureTrips  
as  
select country\_name,  
 trip\_date,  
 trip\_name,  
 trip\_id,  
 max\_no\_places,  
 no\_available\_places  
from Trips1  
where trip\_date > *current\_date*;

## AvailableTrips

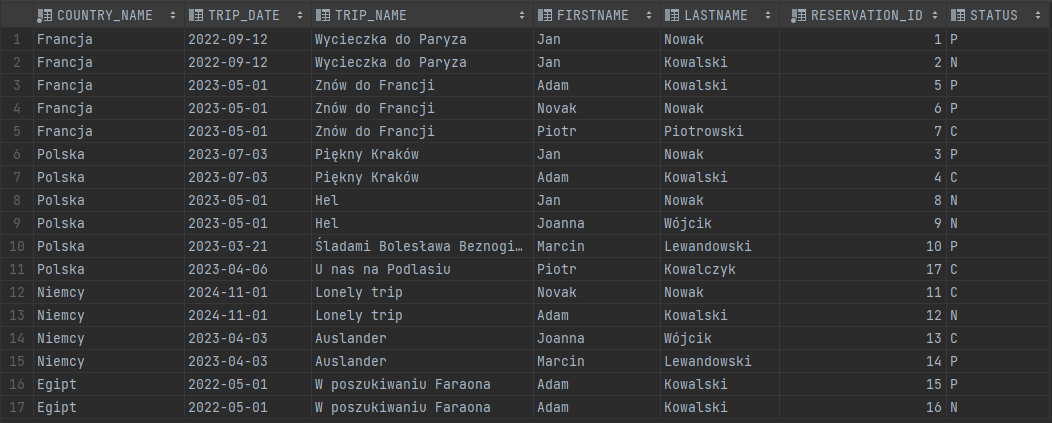
create or replace view AvailableTripsView  
as  
select country\_name,  
 trip\_date,  
 trip\_name,  
 max\_no\_places,  
 no\_available\_places  
from trips  
where trip\_date > *current\_date* and no\_available\_places > 0

## AvailableTrips1

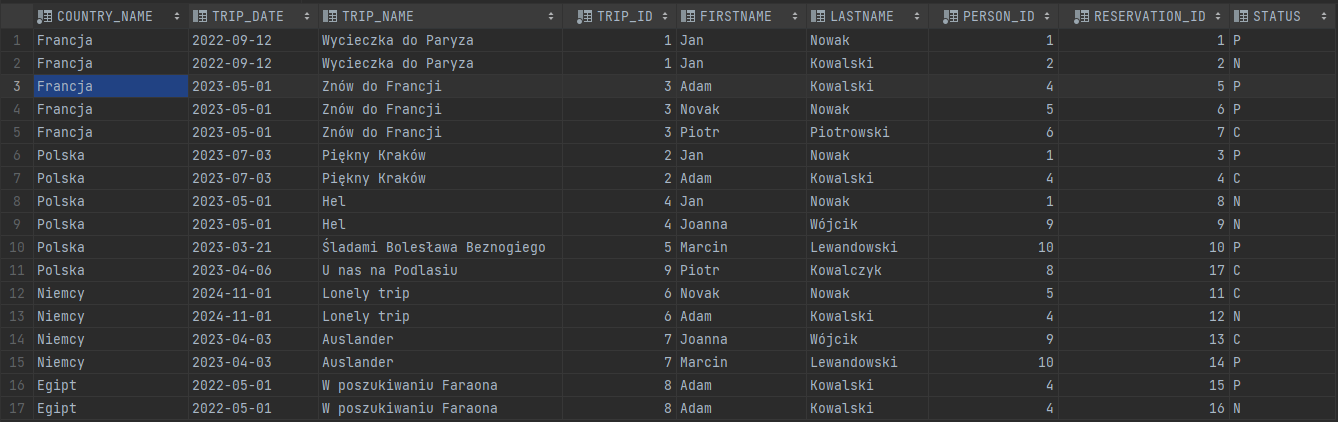
create or replace view AvailableTripsView1  
as  
select \*  
from FutureTrips  
where no\_available\_places > 0;

## Przykładowe użycia

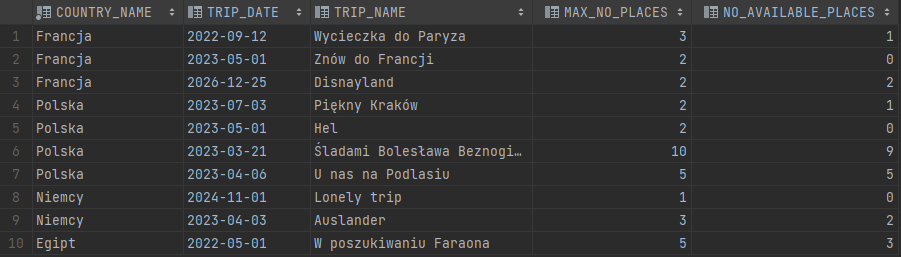
### Reservations



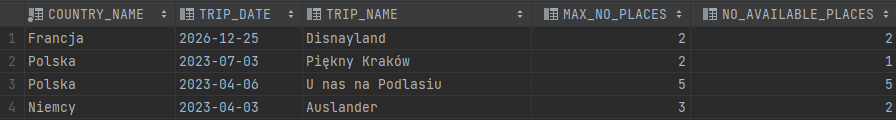
### Reservations1



### Trips



### AvailableTripsView



## Komentarz

Powyższe wyniki działania widoków uzyskane zostały 02.04.2023. Widok AvailableTrips został napisany wykorzystując wcześniej utworzony widok Trips w celu uniknięcia redundancji kodu. Zmieniono nazwę widoku *AvailableTrips* na *AvailableTripsView*, w celu uniknięcia konfliktu nazw z funkcją z kolejnego zadania. Dodany został widok *Reservations1*, który rozszerza pierwotny widok o dodatkowe pola potrzebne funkcjom.

# Funkcje

## TripParticipants

create or replace type TripParticipant as object  
(  
 country\_name varchar2(50),  
 trip\_date date,  
 trip\_name varchar2(50),  
 firstname varchar2(50),  
 lastname varchar2(50),  
 reservation\_id int,  
 status char(1)  
);  
  
  
create or replace type TripParticipantTable is table of TripParticipant;  
  
  
create or replace function *TripParticipants*(trip\_id int)  
 return TripParticipantTable  
 is  
 result TripParticipantTable;  
 valid int;  
begin  
 select *count*(\*) into valid from trip t where t.trip\_id = *TripParticipants*.trip\_id;  
 if valid = 0 then  
 *raise\_application\_error*(-20001, 'trip not found');  
 end if;  
  
 select TripParticipant(country\_name,  
 trip\_date,  
 trip\_name,  
 firstname,  
 lastname,  
 reservation\_id,  
 status) bulk collect  
 into result  
 from reservations1 r  
 where r.trip\_id = *TripParticipants*.trip\_id;  
  
 return result;  
end;

## PersonReservations

create or replace function *PersonReservations*(person\_id int)  
 return TripParticipantTable  
 is  
 result TripParticipantTable;  
 valid int;  
begin  
 select *count*(\*) into valid from person p where p.person\_id = *PersonReservations*.person\_id;  
 if valid = 0 then  
 *raise\_application\_error*(-20001, 'person not found');  
 end if;  
  
 select TripParticipant(country\_name,  
 trip\_date,  
 trip\_name,  
 firstname,  
 lastname,  
 reservation\_id,  
 status) bulk collect  
 into result  
 from reservations1 r  
 where r.person\_id = *PersonReservations*.person\_id;  
  
 return result;  
end;

## AvailableTrips

create or replace type AvailableTrip is object  
(  
 country\_name varchar2(50),  
 trip\_date date,  
 trip\_name varchar2(50),  
 max\_no\_places int,  
 no\_available\_places int  
);  
  
create or replace type AvailableTripTable is table of AvailableTrip;  
  
  
  
create or replace function *AvailableTrips*(country varchar2, date\_from date, date\_to date)  
 return AvailableTripTable  
 is  
 result AvailableTripTable;  
 valid int;  
begin  
 select *count*(\*) into valid from country c where c.country\_name = *AvailableTrips*.country;  
 if valid = 0 then  
 *raise\_application\_error*(-20001, 'country not found');  
 end if;  
  
 select AvailableTrip(country\_name,  
 trip\_date,  
 trip\_name,  
 max\_no\_places,  
 no\_available\_places) bulk collect  
 into result  
 from AVAILABLETRIPSVIEW  
 where COUNTRY\_NAME = *AvailableTrips*.country  
 and trip\_date between *AvailableTrips*.date\_from and *AvailableTrips*.date\_to;  
  
  
 return result;  
  
end;

## Przykładowe wywołania

### TripParticipants

select \* from *TripParticipants*(3);



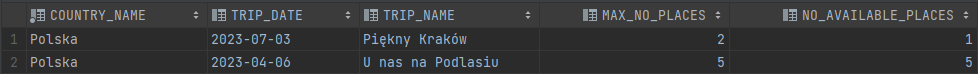
### PersonReservations

select \* from *PERSONRESERVATIONS*(1);



### AvailableTripsView

select \* from AVAILABLETRIPSVIEW where COUNTRY\_NAME = 'Polska';



### AvailableTrips

select \*  
from *AVAILABLETRIPS*('Polska',  
 *to\_date*('2000-01-01', 'YYYY-MM-DD'),  
 *to\_date*('2026-01-01', 'YYYY-MM-DD'));



## 

### CheckAvailablePlaces

create or replace function *CheckAvailablePlaces*(trip\_id int)  
 return boolean  
 is  
 result boolean := false;  
 available\_places int;  
begin  
 select atv.NO\_AVAILABLE\_PLACES  
 into available\_places  
 from AVAILABLETRIPSVIEW1 atv  
 where atv.trip\_id = *CheckAvailablePlaces*.trip\_id;  
  
 if available\_places <> 0 then  
 result := true;  
 end if;  
  
 return result;  
  
end;

## Komentarz

Powyższe funkcje zostały przetestowane dla różnych danych wejściowych, również niepoprawnych, które obsługiwane są za pomocą wyjątku. Funkcja TripParticipants zakłada, że wszystkie stany rezerwacji powinny być wyświetlone w zbiorze wynikowym. Powyższe wywołania związane z funkcją AvailableTrips pokazują działanie tej funkcji jako funkcji filtrującej dane po zakresie dat. W celu sprawdzenia, czy na daną wycieczkę są jeszcze wolne miejsca, powstała funkcja *CheckAvailablePlaces* zwracająca wartość boolean.

# Procedury