### Bazy Danych 2

### Laboratorium 1

Oracle PL/SQL

Mateusz Łopaciński

**Spis treści**

1. **Schemat bazy danych4**
2. **Tworzenie tabel5**
   1. People5
   2. Trips5
   3. Reservations5
   4. ReservationsLog5
3. **Warunki integralnościowe5**
   1. Trips5
   2. Reservations5
   3. ReservationsLog6
4. **Wstawianie danych do tabel6**
   1. People6
   2. Trips7
   3. Reservations7
5. **Widoki8**
   1. ReservationsView8
   2. TripsView9
   3. AvailableTripsView10
6. **Obiekty10**
   1. TripParticipantObject i TripParticipantsTable10
   2. PersonReservationObject i PersonReservationsTable10
   3. AvailableTripObject i AvailableTripsTable11
7. **Funkcje skalarne11**
   1. getBookedPlacesNum11
   2. getAvailablePlacesNum11
   3. doesTripExist12
   4. doesPersonExist12
   5. doesReservationExist13
   6. hasTripTakenPlace13
8. **Funkcje tabelaryczne14**
   1. getTripParticipants14
   2. getPersonReservations15
   3. getAvailableTripsTo16
9. **Procedury (wersja 1.)16**
   1. addReservation16
   2. modifyReservationStatus18
   3. modifyReservationNoPlaces19
   4. modifyMaxNoPlaces20
10. **Triggery21**
    1. AI\_ReservationInsert21
    2. AU\_ReservationStatusUpdate22
    3. AU\_ReservationNoPlacesUpdate22
    4. BI\_ReservationInsert23
    5. BU\_ReservationStatusUpdate23
    6. BU\_ReservationNoPlacesUpdate24
    7. BU\_TripMaxNoPlacesUpdate24
11. **Procedury (wersja 2.)25**
    1. addReservation25
    2. modifyReservationStatus25
    3. modifyMaxNoPlaces26

### Schemat bazy danych

Obraz zawierający stół

Opis wygenerowany automatycznie

### Tworzenie tabel

#### **People**

**CREATE** **TABLE** People **(**

person\_id INT GENERATED ALWAYS **AS** **IDENTITY** **NOT** **NULL,**

firstname VARCHAR2**(**50**)** **NOT** **NULL,**

lastname VARCHAR2**(**50**)** **NOT** **NULL,**

**CONSTRAINT** People\_pk **PRIMARY** **KEY** **(**person\_id**)**

**);**

#### **Trips**

**CREATE** **TABLE** Trips **(**

trip\_id INT GENERATED ALWAYS **AS** **IDENTITY** **NOT** **NULL,**

name VARCHAR2**(**100**)** **NOT** **NULL,**

country VARCHAR2**(**50**)** **NOT** **NULL,**

trip\_date DATE **NOT** **NULL,**

max\_no\_places INT **NOT** **NULL,**

**CONSTRAINT** Trips\_pk **PRIMARY** **KEY** **(**trip\_id**)**

**);**

#### **Reservations**

**CREATE** **TABLE** Reservations **(**

reservation\_id INT GENERATED ALWAYS **AS** **IDENTITY** **NOT** **NULL,**

trip\_id INT **NOT** **NULL,**

person\_id INT **NOT** **NULL,**

status CHAR**(**1**)** **NOT** **NULL,**

no\_places INT **NOT** **NULL,**

**CONSTRAINT** Reservations\_pk **PRIMARY** **KEY** **(**reservation\_id**)**

**);**

#### **ReservationsLog**

**CREATE** **TABLE** ReservationsLog **(**

log\_id INT GENERATED ALWAYS **AS** **IDENTITY** **NOT** **NULL,**

reservation\_id INT **NOT** **NULL,**

log\_date DATE **NOT** **NULL,**

status CHAR**(**1**)** **NOT** **NULL,**

no\_places INT **NOT** **NULL,**

**CONSTRAINT** ReservationsLog\_pk **PRIMARY** **KEY** **(**log\_id**)**

**);**

### Warunki integralnościowe

Poniżej umieściłem warunki integralnościowe, które nie zostały zdefiniowane w kodzie tworzącym tabele.

#### **Trips**

**ALTER** **TABLE** Trips

**ADD** **CONSTRAINT** Trips\_chk1 **CHECK** **(**max\_no\_places **>** 0**);**

#### **Reservations**

**ALTER** **TABLE** Reservations

**ADD** **CONSTRAINT** Reservations\_fk1 **FOREIGN** **KEY** **(**person\_id**)**

**REFERENCES** People**(**person\_id**);**

**ALTER** **TABLE** Reservations

**ADD** **CONSTRAINT** Reservations\_fk2 **FOREIGN** **KEY** **(**trip\_id**)**

**REFERENCES** Trips**(**trip\_id**);**

**ALTER** **TABLE** Reservations

**ADD** **CONSTRAINT** Reservations\_chk1 **CHECK** **(**status **IN** **(**'n'**,** 'p'**,** 'c'**));**

**ALTER** **TABLE** Reservations

**ADD** **CONSTRAINT** Reservations\_chk2 **CHECK** **(**no\_places **>** 0**);**

#### **ReservationsLog**

**ALTER** **TABLE** ReservationsLog

**ADD** **CONSTRAINT** ReservationLog\_chk1 **CHECK** **(**status **IN** **(**'n'**,** 'p'**,** 'c'**));**

**ALTER** **TABLE** ReservationsLog

**ADD** **CONSTRAINT** ReservationsLog\_chk2 **CHECK** **(**no\_places **>** 0**);**

### Wstawianie danych do tabel

#### **People**

**INSERT** **INTO** People **(**firstname**,** lastname**)**

**VALUES** **(**'Adam'**,** 'Kowalski'**);**

**INSERT** **INTO** People **(**firstname**,** lastname**)**

**VALUES** **(**'Jan'**,** 'Nowak'**);**

**INSERT** **INTO** People **(**firstname**,** lastname**)**

**VALUES** **(**'Andrzej'**,** 'Kowalczyk'**);**

**INSERT** **INTO** People **(**firstname**,** lastname**)**

**VALUES** **(**'Anna'**,** 'Klimek'**);**

**INSERT** **INTO** People **(**firstname**,** lastname**)**

**VALUES** **(**'Zbigniew'**,** 'Zygora'**);**

**INSERT** **INTO** People **(**firstname**,** lastname**)**

**VALUES** **(**'Rafał'**,** 'Noga'**);**

**INSERT** **INTO** People **(**firstname**,** lastname**)**

**VALUES** **(**'Aleksandra'**,** 'Sobczak'**);**

**INSERT** **INTO** People **(**firstname**,** lastname**)**

**VALUES** **(**'Maryla'**,** 'Ordon'**);**

**INSERT** **INTO** People **(**firstname**,** lastname**)**

**VALUES** **(**'Piotr'**,** 'Słota'**);**

**INSERT** **INTO** People **(**firstname**,** lastname**)**

**VALUES** **(**'Aleks'**,** 'Stachowiak'**);**

**COMMIT;**

**W rezultacie otrzymujemy tabelę:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | person\_id | firstname | lastname |
| 1 |  | 1 | Adam | Kowalski |
| 2 |  | 2 | Jan | Nowak |
| 3 |  | 3 | Andrzej | Kowalczyk |
| 4 |  | 4 | Anna | Klimek |
| 5 |  | 5 | Zbigniew | Zygora |
| 6 |  | 6 | Rafał | Noga |
| 7 |  | 7 | Aleksandra | Sobczak |
| 8 |  | 8 | Maryla | Ordon |
| 9 |  | 9 | Piotr | Słota |
| 10 |  | 10 | Aleks | Stachowiak |

**Tabela 4.1. People**

#### **Trips**

**INSERT** **INTO** Trips **(**name**,** country**,** trip\_date**,** max\_no\_places**)**

**VALUES** **(**'wycieczka do Paryza'**,** 'Francja'**,** **TO\_DATE(**'2021-09-03'**,** 'yyyy-mm-dd'**),** 5**);**

**INSERT** **INTO** Trips **(**name**,** country**,** trip\_date**,** max\_no\_places**)**

**VALUES** **(**'wycieczka do Krakowa'**,** 'Polska'**,** **TO\_DATE(**'2022-12-05'**,** 'yyyy-mm-dd'**),** 8**);**

**INSERT** **INTO** Trips **(**name**,** country**,** trip\_date**,** max\_no\_places**)**

**VALUES** **(**'wycieczka do Warszawy'**,** 'Polska'**,** **TO\_DATE(**'2022-04-11'**,** 'yyyy-mm-dd'**),** 12**);**

**INSERT** **INTO** Trips **(**name**,** country**,** trip\_date**,** max\_no\_places**)**

**VALUES** **(**'wycieczka do Madrytu'**,** 'Hiszpania'**,** **TO\_DATE(**'2022-07-02'**,** 'yyyy-mm-dd'**),** 8**);**

**COMMIT;**

**W rezultacie otrzymujemy tabelę:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | trip\_id | name | country | trip\_date | max\_no\_places | | |
| 1 |  | 1 | wycieczka do Paryza | Francja | 2021-09-03 | | 5 |
| 2 |  | 2 | wycieczka do Krakowa | Polska | 2022-12-05 | | 8 |
| 3 |  | 3 | wycieczka do Warszawy | Polska | 2022-04-11 | | 12 |
| 4 |  | 4 | wycieczka do Madrytu | Hiszpania | 2022-07-02 | | 8 |

**Tabela 4.2. Trips**

#### **Reservations**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**1**,** 1**,** 1**,** 'n'**);**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**1**,** 2**,** 2**,** 'p'**);**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**2**,** 1**,** 1**,** 'p'**);**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**2**,** 2**,** 1**,** 'c'**);**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**2**,** 4**,** 2**,** 'n'**);**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**3**,** 5**,** 4**,** 'c'**);**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**3**,** 5**,** 3**,** 'n'**);**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**3**,** 6**,** 4**,** 'p'**);**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**4**,** 7**,** 3**,** 'p'**);**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**4**,** 9**,** 1**,** 'c'**);**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** no\_places**,** status**)**

**VALUES** **(**4**,** 8**,** 5**,** 'n'**);**

**COMMIT;**

**W rezultacie otrzymujemy tabelę:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | reservation\_id | trip\_id | person\_id | status | no\_places | |
| 1 |  | 1 | 1 | 1 | n | | 1 |
| 2 |  | 2 | 1 | 2 | p | | 2 |
| 3 |  | 3 | 2 | 1 | p | | 1 |
| 4 |  | 4 | 2 | 2 | c | | 1 |
| 5 |  | 5 | 2 | 4 | n | | 2 |
| 6 |  | 6 | 3 | 5 | c | | 4 |
| 7 |  | 7 | 3 | 5 | n | | 3 |
| 8 |  | 8 | 3 | 6 | p | | 4 |
| 9 |  | 9 | 4 | 7 | p | | 3 |
| 10 |  | 10 | 4 | 9 | c | | 1 |
| 11 |  | 11 | 4 | 8 | n | | 5 |

**Tabela 4.3. Reservations**

### Widoki

#### **ReservationsView**

reservations(country,trip\_date,trip\_name, firstname, lastname,reservation\_id,no\_places,status)

**CREATE** **OR** **REPLACE** **VIEW** ReservationsView

**AS**

**SELECT**

t**.**trip\_id**,**

t**.**country**,**

t**.**trip\_date**,**

t**.**name **AS** trip\_name**,**

p**.**person\_id**,**

p**.**firstname**,**

p**.**lastname**,**

r**.**reservation\_id**,**

r**.**no\_places**,**

r**.**status

**FROM** Trips t

**INNER** **JOIN** Reservations r

**ON** r**.**trip\_id **=** t**.**trip\_id

**INNER** **JOIN** People p

**on** p**.**person\_id **=** r**.**person\_id**;**

**SELECT** **\*** **FROM** ReservationsView**;**

**W rezultacie otrzymujemy tabelę:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | trip\_id | country | trip\_date | trip\_name | person\_id | | firstname | lastname | reservation\_id | no\_places | status |
| 1 |  | 1 | Francja | 2021-09-03 | wycieczka do Paryża | | 1 | Adam | Kowalski | 1 | 1 | n |
| 2 |  | 2 | Polska | 2022-12-05 | wycieczka do Krakowa | | 1 | Adam | Kowalski | 3 | 1 | p |
| 3 |  | 1 | Francja | 2021-09-03 | wycieczka do Paryża | | 2 | Jan | Nowak | 2 | 2 | p |
| 4 |  | 2 | Polska | 2022-12-05 | wycieczka do Krakowa | | 2 | Jan | Nowak | 4 | 1 | c |
| 5 |  | 2 | Polska | 2022-12-05 | wycieczka do Krakowa | | 4 | Anna | Klimek | 5 | 2 | n |
| 6 |  | 3 | Polska | 2022-04-11 | wycieczka do Warszawy | | 5 | Zbigniew | Zygora | 6 | 4 | c |
| 7 |  | 3 | Polska | 2022-04-11 | wycieczka do Warszawy | | 5 | Zbigniew | Zygora | 7 | 3 | n |
| 8 |  | 3 | Polska | 2022-04-11 | wycieczka do Warszawy | | 6 | Rafał | Noga | 8 | 4 | p |
| 9 |  | 4 | Hiszpania | 2022-07-02 | Wycieczka do Madrytu | | 7 | Aleksandra | Sobczak | 9 | 3 | p |
| 10 |  | 4 | Hiszpania | 2022-07-02 | Wycieczka do Madrytu | | 8 | Maryla | Ordon | 11 | 5 | n |
| 11 |  | 4 | Hiszpania | 2022-07-02 | Wycieczka do Madrytu | | 9 | Piotr | Słota | 10 | 1 | c |

**Tabela 5.1. Tabela utworzona przez widok ReservationsView**

#### **TripsView**

trips(country,trip\_date, trip\_name,max\_no\_places, no\_available\_places)

**CREATE** **OR** **REPLACE** **VIEW** TripsView

**AS**

**SELECT**

trip\_id**,**

country**,**

trip\_date**,**

name **AS** trip\_name**,**

max\_no\_places**,**

getAvailablePlaces**(**trip\_id**)** **AS** no\_available\_places

**FROM** Trips**;**

**W rezultacie otrzymujemy tabelę:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | trip\_id | country | trip\_date | trip\_name | max\_no\_places | | no\_available\_places |
| 1 |  | 1 | Francja | 2021-09-03 | wycieczka do Paryża | | 5 | 2 |
| 2 |  | 2 | Polska | 2022-12-05 | wycieczka do Krakowa | | 8 | 5 |
| 3 |  | 3 | Polska | 2022-04-11 | wycieczka do Warszawy | | 12 | 5 |
| 4 |  | 4 | Hiszpania | 2022-07-02 | wycieczka do Madrytu | | 8 | 0 |

**Tabela 5.2. Tabela utworzona przez widok TripsView**

#### **AvailableTripsView**

trips(country,trip\_date, trip\_name, max\_no\_places, no\_available\_places)

**CREATE** **OR** **REPLACE** **VIEW** AvailableTripsView

**AS**

**SELECT** **\***

**FROM** TripsView

**WHERE** trip\_date **>** **SYSDATE**

**AND** no\_available\_places **>** 0**;**

**W rezultacie otrzymujemy tabelę:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | trip\_id | country | trip\_date | trip\_name | max\_no\_places | | no\_available\_places |
| 1 |  | 2 | Polska | 2022-12-05 | wycieczka do Krakowa | | 8 | 5 |
| 2 |  | 3 | Polska | 2022-04-11 | wycieczka do Warszawy | | 12 | 5 |

**Tabela 5.2. Tabela utworzona przez widok AvailableTripsView**

### Obiekty

#### **TripParticipantObject i TripParticipantsTable**

**CREATE** **OR** **REPLACE** **TYPE** TripParticipantObject **AS** **OBJECT** **(**

firstname VARCHAR2**(**50**),**

lastname VARCHAR2**(**50**),**

reservation\_id INT**,**

no\_places INT**,**

status CHAR**(**1**)**

**);**

**CREATE** **OR** **REPLACE** **TYPE** TripParticipantsTable **IS** **TABLE** **OF** TripParticipantObject**;**

#### **PersonReservationObject i PersonReservationsTable**

**CREATE** **OR** **REPLACE** **TYPE** PersonReservationObject **AS** **OBJECT** **(**

trip\_id INT**,**

country VARCHAR2**(**50**),**

trip\_date DATE**,**

trip\_name VARCHAR2**(**100**),**

reservation\_id INT**,**

no\_places INT**,**

status CHAR**(**1**)**

**);**

**CREATE** **OR** **REPLACE** **TYPE** PeopleReservationsTable **IS** **TABLE** **OF** PersonReservationObject**;**

#### **AvailableTripObject i AvailableTripsTable**

**CREATE** **OR** **REPLACE** **TYPE** AvailableTripObject **AS** **OBJECT** **(**

trip\_id INT**,**

name VARCHAR2**(**100**),**

country VARCHAR2**(**50**),**

trip\_date DATE**,**

max\_no\_places INT

**);**

**CREATE** **OR** **REPLACE** **TYPE** AvailableTripsTable **IS** **TABLE** **OF** AvailableTripObject**;**

### Funkcje skalarne

#### **getBookedPlacesNum**

**CREATE** **OR** **REPLACE** **FUNCTION** getBookedPlacesNum**(**

p\_trip\_id Trips**.**trip\_id**%TYPE**

**)**

**RETURN** Reservations**.**no\_places**%TYPE**

**AS**

l\_booked\_places Reservations**.**no\_places**%TYPE;**

**BEGIN**

**SELECT** **NVL(SUM(**no\_places**),** 0**)**

**INTO** l\_booked\_places

**FROM** Reservations

**WHERE** trip\_id **=** p\_trip\_id

**AND** status **!=** 'c'**;**

**RETURN** l\_booked\_places**;**

**END;**

**Przykład działania:**

**BEGIN**

DBMS\_OUTPUT**.**PUT\_LINE**(**getBookedPlacesNum(1)**);**

**END;**

**Wynik:** 3

#### **getAvailablePlacesNum**

**CREATE** **OR** **REPLACE** **FUNCTION** getAvailablePlacesNum**(**

p\_trip\_id Trips**.**trip\_id**%TYPE**

**)**

**RETURN** Reservations**.**no\_places**%TYPE**

**AS**

l\_available\_places Trips**.**max\_no\_places**%TYPE;**

**BEGIN**

**SELECT** max\_no\_places **-** getBookedPlacesNum**(**trip\_id**)**

**INTO** l\_available\_places

**FROM** Trips

**WHERE** trip\_id **=** p\_trip\_id**;**

**RETURN** l\_available\_places**;**

**EXCEPTION**

**WHEN** NO\_DATA\_FOUND **THEN**

RAISE\_APPLICATION\_ERROR**(**-20000**,** 'Trip with id: ' **||** p\_trip\_id **||** ' does not exist'**);**

**RETURN** **NULL;**

**END;**

**Przykład działania:**

**BEGIN**

DBMS\_OUTPUT**.**PUT\_LINE**(**getAvailablePlacesNum(1)**);**

**END;**

**Wynik:** 2

#### **doesTripExist**

**CREATE** **OR** **REPLACE** **FUNCTION** doesTripExist**(**

p\_trip\_id Trips**.**trip\_id**%TYPE**

**)**

**RETURN** BOOLEAN

**AS**

exist NUMBER**;**

**BEGIN**

**SELECT**

**CASE**

**WHEN** **EXISTS(SELECT** **\*** **FROM** Trips **WHERE** trip\_id **=** p\_trip\_id**)** **THEN** 1

**ELSE** 0

**END**

**INTO** exist

**FROM** Dual**;**

**IF** exist **=** 1 **THEN**

**RETURN** **TRUE;**

**ELSE**

**RETURN** **FALSE;**

**END** **IF;**

**END;**

**Przykład działania:**

**BEGIN**

**IF** doesTripExist(1) **THEN**

DBMS\_OUTPUT**.**PUT\_LINE**(**'Trip exists'**);**

**ELSE**

DBMS\_OUTPUT**.**PUT\_LINE**(**'Trip does not exist'**);**

**END IF;**

**END;**

**Wynik:** Trip exists

#### **doesPersonExist**

**CREATE** **OR** **REPLACE** **FUNCTION** doesPersonExist**(**

p\_person\_id People**.**person\_id**%TYPE**

**)**

**RETURN** BOOLEAN

**AS**

exist NUMBER**;**

**BEGIN**

**SELECT**

**CASE**

**WHEN** **EXISTS(SELECT** **\*** **FROM** People **WHERE** person\_id **=** p\_person\_id**)** **THEN** 1

**ELSE** 0

**END**

**INTO** exist

**FROM** Dual**;**

**IF** exist **=** 1 **THEN**

**RETURN** **TRUE;**

**ELSE**

**RETURN** **FALSE;**

**END** **IF;**

**END;**

**Przykład działania:**

**BEGIN**

**IF** doesPersonExist(123) **THEN**

DBMS\_OUTPUT**.**PUT\_LINE**(**'Person exists'**);**

**ELSE**

DBMS\_OUTPUT**.**PUT\_LINE**(**'Person does not exist'**);**

**END IF;**

**END;**

**Wynik:** Person does not exists

#### **doesReservationExist**

**CREATE** **OR** **REPLACE** **FUNCTION** doesReservationExist**(**

p\_reservation\_id Reservations**.**reservation\_id**%TYPE**

**)**

**RETURN** BOOLEAN

**AS**

exist NUMBER**;**

**BEGIN**

**SELECT**

**CASE**

**WHEN** **EXISTS(SELECT** **\*** **FROM** Reservations **WHERE** reservation\_id **=** p\_reservation\_id**)** **THEN** 1

**ELSE** 0

**END**

**INTO** exist

**FROM** Dual**;**

**IF** exist **=** 1 **THEN**

**RETURN** **TRUE;**

**ELSE**

**RETURN** **FALSE;**

**END** **IF;**

**END;**

**Przykład działania:**

**BEGIN**

**IF** doesReservationExist(10) **THEN**

DBMS\_OUTPUT**.**PUT\_LINE**(**'Reservation exists'**);**

**ELSE**

DBMS\_OUTPUT**.**PUT\_LINE**(**'Reservation does not exist'**);**

**END IF;**

**END;**

**Wynik:** Reservation exists

#### **hasTripTakenPlace**

**CREATE** **OR** **REPLACE** **FUNCTION** hasTripTakenPlace**(**

p\_trip\_id Trips**.**trip\_id**%TYPE,**

p\_date DATE **DEFAULT** **SYSDATE**

**)**

**RETURN** BOOLEAN

**AS**

l\_trip\_date DATE**;**

**BEGIN**

**IF** **NOT** doesTripExist**(**p\_trip\_id**)** **THEN**

RAISE\_APPLICATION\_ERROR**(**-20000**,** 'There is no trip with id ' **||** p\_trip\_id **||** ' in the database'**);**

**END** **IF;**

**SELECT** trip\_date

**INTO** l\_trip\_date

**FROM** Trips

**WHERE** trip\_id **=** p\_trip\_id**;**

**IF** l\_trip\_date **<=** p\_date **THEN**

**RETURN** **TRUE;**

**ELSE**

**RETURN** **FALSE;**

**END** **IF;**

**END;**

**Przykład działania:**

**BEGIN**

**IF** hasTripTakenPlace(2) **THEN**

DBMS\_OUTPUT**.**PUT\_LINE**(**'Trip took place before'**);**

**ELSE**

DBMS\_OUTPUT**.**PUT\_LINE**(**'Trip has not taken place yet'**);**

**END IF;**

**END;**

**Wynik:** Trip has not taken place yet

### Funkcje tabelaryczne

#### **getTripParticipants**

**CREATE** **OR** **REPLACE** **FUNCTION** getTripParticipants**(**

p\_trip\_id Trips**.**trip\_id**%TYPE**

**)**

**RETURN** TripParticipantsTable

**AS**

l\_result TripParticipantsTable**;**

**BEGIN**

**IF** **NOT** doesTripExist**(**p\_trip\_id**)** **THEN**

RAISE\_APPLICATION\_ERROR**(**-20000**,** 'There are is no trip with id ' **||** p\_trip\_id **||** ' in the database'**);**

**END** **IF;**

**SELECT** TripParticipantObject**(**

firstname**,**

lastname**,**

reservation\_id**,**

no\_places**,**

status

**)**

**BULK** **COLLECT** **INTO** l\_result

**FROM** ReservationsView

**WHERE** trip\_id **=** p\_trip\_id

**AND** status **!=** 'c'**;**

**RETURN** l\_result**;**

**END;**

**Przykład działania:**

**SELECT** **\*** **FROM** getTripParticipants**(**3**);**

**Rezultat:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | firstname | lastname | reservation\_id | no\_places | status | |
| 1 |  | Zbigniew | Zygora | 7 | 3 | | n |
| 2 |  | Rafał | Noga | 8 | 4 | | p |

**Tabela 8.1. Przykładowa tabela zwrócona przez funkcję getTripParticipants**

#### **getPersonReservations**

**CREATE** **OR** **REPLACE** **FUNCTION** getPersonReservations**(**

p\_person\_id People**.**person\_id**%TYPE**

**)**

**RETURN** PeopleReservationsTable

**AS**

l\_result PeopleReservationsTable**;**

**BEGIN**

**IF** **NOT** doesPersonExist**(**p\_person\_id**)** **THEN**

RAISE\_APPLICATION\_ERROR**(**-20000**,** 'There is no person with id ' **||** p\_person\_id **||** ' in the database'**);**

**END** **IF;**

**SELECT** PersonReservationObject**(**

trip\_id**,**

country**,**

trip\_date**,**

trip\_name**,**

reservation\_id**,**

no\_places**,**

status

**)**

**BULK** **COLLECT** **INTO** l\_result

**FROM** ReservationsView

**WHERE** person\_id **=** p\_person\_id**;**

**RETURN** l\_result**;**

**END;**

**Przykład działania:**

**SELECT** **\*** **FROM** getPersonReservations**(**2**);**

**Rezultat:**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | trip\_id | country | trip\_date | trip\_name | | reservation\_id | | | no\_places | | status |
| 1 | 1 | Francja | 2021-09-03 | wycieczka do Paryża | | | 2 | | 2 | p | |
| 2 | 2 | Polska | 2022-12-05 | | wycieczka do Krakowa | | | 4 | 1 | c | |

**Tabela 8.2. Przykładowa tabela zwrócona przez funkcję getPersonReservations**

#### **getAvailableTripsTo**

-- (When there are no remaining places, a trip is considered unavailable - see Spain in examples)

-- (if p\_from\_date is lower than the current date, a function below will return only trips which

-- date is between the current date and the p\_to\_date)

**CREATE** **OR** **REPLACE** **FUNCTION** getAvailableTripsTo**(**

p\_country\_name Trips**.**country**%TYPE,**

p\_from\_date DATE**,**

p\_to\_date DATE

**)**

**RETURN** AvailableTripsTable

**AS**

l\_result AvailableTripsTable**;**

**BEGIN**

-- Show warning information if the current date is greater than the p\_from\_date

**IF** p\_from\_date **<** **SYSDATE** **THEN**

DBMS\_OUTPUT**.**PUT\_LINE**(**'Warning: Specified trip start date (' **||** p\_from\_date **||**

') is lower than the current date.' **||** 'Current date (' **||** **SYSDATE** **||**

') will be used instead'**);**

**END** **IF;**

**SELECT** AvailableTripObject**(**

trip\_id**,**

trip\_name**,**

country**,**

trip\_date**,**

max\_no\_places

**)**

**BULK** **COLLECT**

**INTO** l\_result

**FROM** AvailableTripsView

**WHERE** country **=** p\_country\_name

**AND** trip\_date **BETWEEN** p\_from\_date **AND** p\_to\_date**;**

**RETURN** l\_result**;**

**END;**

**Przykład działania:**

**SELECT** **\*** **FROM** getAvailableTripsTo**(**'Polska'**,** '2020-01-01'**,** '2022-12-31'**);**

**Rezultat:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | trip\_id | name | country | trip\_date | max\_no\_places | |
| 1 | 2 | Wycieczka do Krakowa | Polska | 2022-12-05 | | 8 |
| 2 | 3 | Wycieczka do Warszawy | Polska | 2022-04-11 | | 12 |

**Tabela 8.3. Przykładowa tabela zwrócona przez funkcję getAvailableTripsTo**

### Procedury (wersja 1.)

#### **addReservation**

Poniżej znajdują się 1. wersje procedur (zanim zostały dodane triggery, które obsługują sprawdzanie poprawności odpowiednich danych).

**CREATE** **OR** **REPLACE** **PROCEDURE** addReservation**(**

p\_trip\_id Trips**.**trip\_id**%TYPE,**

p\_person\_id People**.**person\_id**%TYPE,**

p\_no\_places Reservations**.**no\_places**%TYPE**

**)**

**AS**

l\_available\_places Reservations**.**no\_places**%TYPE;**

**BEGIN**

**IF** **NOT** doesPersonExist**(**p\_person\_id**)** **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'The person with id ' **||** p\_person\_id **||** ' does not exit'**);**

**END** **IF;**

**IF** hasTripTakenPlace**(**p\_trip\_id**)** **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'The trip with id ' **||** p\_trip\_id **||** ' took place before'**);**

**END** **IF;**

**IF** p\_no\_places **<** 1 **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'Cannot book less than 1 place for a trip'**);**

**END** **IF;**

l\_available\_places **:=** getAvailablePlacesNum**(**p\_trip\_id**);**

**IF** l\_available\_places **=** 0 **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There are no available places for a trip with id ' **||** p\_trip\_id**);**

**ELSIF** l\_available\_places **<** p\_no\_places **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There are only ' **||** l\_available\_places **||**

' places available for a trip with id ' **||** p\_trip\_id**);**

**END** **IF;**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** status**,** no\_places**)**

**VALUES** **(**p\_trip\_id**,** p\_person\_id**,** 'n'**,** p\_no\_places**);**

**COMMIT;**

**END;**

**Przykład działania:**

**BEGIN**

addReservation**(**2**,** 1**,** 5**);**

**END;**

**Rezerwacje po użyciu procedury:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | reservation\_id | trip\_id | person\_id | status | no\_places | |
| 1 |  | 1 | 1 | 1 | n | | 1 |
| 2 |  | 2 | 1 | 2 | p | | 2 |
| 3 |  | 3 | 2 | 1 | p | | 1 |
| 4 |  | 4 | 2 | 2 | c | | 1 |
| 5 |  | 5 | 2 | 4 | n | | 2 |
| 6 |  | 6 | 3 | 5 | c | | 4 |
| 7 |  | 7 | 3 | 5 | n | | 3 |
| 8 |  | 8 | 3 | 6 | p | | 4 |
| 9 |  | 9 | 4 | 7 | p | | 3 |
| 10 |  | 10 | 4 | 9 | c | | 1 |
| 11 |  | 11 | 4 | 8 | n | | 5 |
| 12 |  | 12 | 2 | 1 | n | | 5 |

**Tabela 9.1. Rezerwacje po dodaniu nowej rezerwacji**

Jak widzimy, w ostatnim wierszu tabeli **9.1.** pojawiła się nowa rezerwacja o zadanych parametrach.

#### **modifyReservationStatus**

**CREATE** **OR** **REPLACE** **PROCEDURE** modifyReservationStatus**(**

p\_reservation\_id Reservations**.**reservation\_id**%TYPE,**

p\_status Reservations**.**status**%TYPE**

**)**

**AS**

l\_curr\_status Reservations**.**status**%TYPE;**

l\_trip\_id Reservations**.**trip\_id**%TYPE;**

l\_no\_places Reservations**.**no\_places**%TYPE;**

l\_available\_places Reservations**.**no\_places**%TYPE;**

**BEGIN**

**SELECT**

status**,**

trip\_id**,**

no\_places

**INTO**

l\_curr\_status**,**

l\_trip\_id**,**

l\_no\_places

**FROM** Reservations

**WHERE** reservation\_id **=** p\_reservation\_id**;**

**CASE** p\_status

**WHEN** l\_curr\_status **THEN**

DBMS\_OUTPUT**.**PUT\_LINE**(**'The reservation with id ' **||** p\_reservation\_id **||**

' has already the status: ' **||** p\_status**);**

**RETURN;**

**WHEN** 'c' **THEN**

**NULL;**

**WHEN** 'n' **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'Cannot change the status of the reservation with id ' **||**

p\_reservation\_id **||** ' to: n'**);**

**WHEN** 'p' **THEN**

-- Check if can make cancelled reservation available (paid) again

-- (check if there are enough empty places for a trip)

**IF** l\_curr\_status **=** 'c' **THEN**

l\_available\_places **:=** getAvailablePlacesNum**(**l\_trip\_id**);**

**IF** l\_available\_places **<** l\_no\_places **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'Not enough places available to update the cancelled reservation status'**);**

**END** **IF;**

**END** **IF;**

**ELSE**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'Status: ' **||** p\_status **||** ' is not a valid reservation status'**);**

**END** **CASE;**

-- If everything is correct, update the reservation status

**UPDATE** Reservations

**SET** status **=** p\_status

**WHERE** reservation\_id **=** p\_reservation\_id**;**

**EXCEPTION**

**WHEN** NO\_DATA\_FOUND **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There is no reservation with id ' **||** p\_reservation\_id **||** ' in the database'**);**

**COMMIT;**

**END;**

**Przykład działania:**

**BEGIN**

modifyReservationStatus**(**2**,** 'c'**);**

modifyReservationStatus**(**1**,** 'c'**);**

modifyReservationStatus**(**12**,** 'p'**);**

**END;**

**Rezerwacje po użyciu procedury:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | reservation\_id | trip\_id | person\_id | status | no\_places | |
| 1 |  | 1 | 1 | 1 | c | | 1 |
| 2 |  | 2 | 1 | 2 | c | | 2 |
| 3 |  | 3 | 2 | 1 | p | | 1 |
| 4 |  | 4 | 2 | 2 | c | | 1 |
| 5 |  | 5 | 2 | 4 | n | | 2 |
| 6 |  | 6 | 3 | 5 | c | | 4 |
| 7 |  | 7 | 3 | 5 | n | | 3 |
| 8 |  | 8 | 3 | 6 | p | | 4 |
| 9 |  | 9 | 4 | 7 | p | | 3 |
| 10 |  | 10 | 4 | 9 | c | | 1 |
| 11 |  | 11 | 4 | 8 | n | | 5 |
| 12 |  | 12 | 2 | 1 | p | | 5 |

**Tabela 9.2. Rezerwacje po modyfikacji statusu wybranych rezerwacji**

#### **modifyReservationNoPlaces**

**CREATE** **OR** **REPLACE** **PROCEDURE** modifyReservationNoPlaces**(**

p\_reservation\_id Reservations**.**reservation\_id**%TYPE,**

p\_no\_places Reservations**.**no\_places**%TYPE**

**)**

**AS**

l\_curr\_no\_places Reservations**.**no\_places**%TYPE;**

l\_available\_places Reservations**.**no\_places**%TYPE;**

**BEGIN**

**SELECT**

no\_places**,**

getAvailablePlacesNum**(**trip\_id**)**

**INTO**

l\_curr\_no\_places**,**

l\_available\_places

**FROM** Reservations

**WHERE** reservation\_id **=** p\_reservation\_id**;**

**IF** p\_no\_places **<=** 0 **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'The number of booked places should be greater than 0'**);**

**ELSIF** p\_no\_places **-** l\_curr\_no\_places **>** l\_available\_places **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There are not enough free places. Max possible number of places to book: ' **||**

**(**l\_available\_places **+** l\_curr\_no\_places**));**

**END** **IF;**

**UPDATE** Reservations

**SET** no\_places **=** p\_no\_places

**WHERE** reservation\_id **=** p\_reservation\_id**;**

**EXCEPTION**

**WHEN** NO\_DATA\_FOUND **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There is no reservation with id ' **||** p\_reservation\_id **||** ' in the database'**);**

**COMMIT;**

**END;**

**Przykład działania:**

**BEGIN**

modifyReservationNoPlaces**(**1**,** 6**);**

**END;**

**Rezerwacje po użyciu procedury:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | reservation\_id | trip\_id | person\_id | status | no\_places | |
| 1 |  | 1 | 1 | 1 | c | | 6 |
| 2 |  | 2 | 1 | 2 | c | | 2 |
| 3 |  | 3 | 2 | 1 | p | | 1 |
| 4 |  | 4 | 2 | 2 | c | | 1 |
| 5 |  | 5 | 2 | 4 | n | | 2 |
| 6 |  | 6 | 3 | 5 | c | | 4 |
| 7 |  | 7 | 3 | 5 | n | | 3 |
| 8 |  | 8 | 3 | 6 | p | | 4 |
| 9 |  | 9 | 4 | 7 | p | | 3 |
| 10 |  | 10 | 4 | 9 | c | | 1 |
| 11 |  | 11 | 4 | 8 | n | | 5 |
| 12 |  | 12 | 2 | 1 | p | | 5 |

**Tabela 9.3. Rezerwacje po modyfikacji liczby miejsc w rezerwacji o id równym 1**

#### **modifyMaxNoPlaces**

**CREATE** **OR** **REPLACE** **PROCEDURE** modifyMaxNoPlaces**(**

p\_trip\_id Trips**.**trip\_id**%TYPE,**

p\_max\_no\_places Trips**.**max\_no\_places**%TYPE**

**)**

**AS**

l\_booked\_places Reservations**.**no\_places**%TYPE;**

l\_curr\_max\_no\_places Trips**.**max\_no\_places**%TYPE;**

**BEGIN**

**SELECT**

max\_no\_places**,**

getBookedPlacesNum**(**p\_trip\_id**)**

**INTO**

l\_curr\_max\_no\_places**,**

l\_booked\_places

**FROM** Trips

**WHERE** trip\_id **=** p\_trip\_id**;**

**IF** p\_max\_no\_places **=** l\_curr\_max\_no\_places **THEN**

DBMS\_OUTPUT**.**PUT\_LINE**(**'The trip with id ' **||** p\_trip\_id **||**

' has already the maximum number of places set to ' **||** l\_curr\_max\_no\_places**);**

**RETURN;**

**END** **IF;**

**IF** p\_max\_no\_places **<** l\_booked\_places **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'The maximum number of places (' **||** p\_max\_no\_places **||**

') cannot be lower than the total number of booked places (' **||** l\_booked\_places **||**

') for a trip wit id ' **||** p\_trip\_id**);**

**END** **IF;**

**UPDATE** Trips

**SET** max\_no\_places **=** p\_max\_no\_places

**WHERE** trip\_id **=** p\_trip\_id**;**

**EXCEPTION**

**WHEN** NO\_DATA\_FOUND **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There is no trip with id ' **||** p\_trip\_id **||** ' in the database'**);**

**COMMIT;**

**END;**

**Przykład działania:**

**BEGIN**

modifyMaxNoPlaces**(**4**,** 8**);**

modifyMaxNoPlaces**(**3**,** 7**);**

modifyMaxNoPlaces**(**2**,** 10**);**

**END;**

**Wycieczki po użyciu procedury:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | trip\_id | name | country | trip\_date | max\_no\_places | | |
| 1 |  | 1 | wycieczka do Paryza | Francja | 2021-09-03 | | 5 |
| 2 |  | 2 | wycieczka do Krakowa | Polska | 2022-12-05 | | 10 |
| 3 |  | 3 | wycieczka do Warszawy | Polska | 2022-04-11 | | 7 |
| 4 |  | 4 | wycieczka do Madrytu | Hiszpania | 2022-07-02 | | 8 |

**Tabela 9.4. Wycieczki po modyfikacji maksymalnej liczby miejsc**

### Triggery

#### **AI\_ReservationInsert**

**CREATE** **OR** **REPLACE** **TRIGGER** AI\_ReservationInsert

**AFTER** **INSERT**

**ON** Reservations

**FOR** **EACH** **ROW**

**BEGIN**

**INSERT** **INTO** ReservationsLog **(**reservation\_id**,** log\_date**,** status**,** no\_places**)**

**VALUES** **(:NEW.**reservation\_id**,** **SYSDATE,** **:NEW.**status**,** **:NEW.**no\_places**);**

**END;**

**Przykład działania:**

**BEGIN**

addReservation**(**2**,** 1**,** 1**);**

**END;**

**Tabela ReservationsLog po dodaniu rezerwacji do tabeli Reservations:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | log\_id | reservation\_id | log\_date | status | | no\_places | |
| 1 | 1 | 13 | 2022-03-12 02:05:23 | n | 1 | |

**Tabela 10.1. Tabela ReservationsLog po dodaniu nowej rezerwacji**

#### **AU\_ReservationStatusUpdate**

**CREATE** **OR** **REPLACE** **TRIGGER** AU\_ReservationStatusUpdate

**AFTER** **UPDATE**

**OF** status **ON** Reservations

**FOR** **EACH** **ROW**

**BEGIN**

**IF** **:NEW.**status **!=** **:OLD.**status **THEN**

**INSERT** **INTO** ReservationsLog **(**reservation\_id**,** log\_date**,** status**,** no\_places**)**

**VALUES** **(:NEW.**reservation\_id**,** **SYSDATE,** **:NEW.**status**,** **:NEW.**no\_places**);**

**END** **IF;**

**END;**

**Przykład działania:**

**DECLARE**

l\_reservation\_id Reservations**.**reservation\_id**%TYPE;**

**BEGIN**

**SELECT** **MAX(**reservation\_id**)**

**INTO** l\_reservation\_id

**FROM** Reservations**;**

modifyReservationStatus**(**l\_reservation\_id**,** 'p'**);**

**END;**

**Tabela ReservationsLog po zaktualizowaniu statusu rezerwacji:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | log\_id | reservation\_id | log\_date | status | | no\_places | |
| 1 | 1 | 13 | 2022-03-12 02:05:23 | n | 1 | |
| 2 | 2 | 13 | 2022-03-12 02:10:14 | p | 1 | |

**Tabela 10.2. Tabela ReservationsLog po zaktualizowaniu statusu rezerwacji**

#### **AU\_ReservationNoPlacesUpdate**

**CREATE** **OR** **REPLACE** **TRIGGER** AU\_ReservationNoPlacesUpdate

**AFTER** **UPDATE**

**OF** no\_places **ON** Reservations

**FOR** **EACH** **ROW**

**BEGIN**

**IF** **:NEW.**no\_places **!=** **:OLD.**no\_places **THEN**

**INSERT** **INTO** ReservationsLog **(**reservation\_id**,** log\_date**,** status**,** no\_places**)**

**VALUES** **(:NEW.**reservation\_id**,** **SYSDATE,** **:NEW.**status**,** **:NEW.**no\_places**);**

**END** **IF;**

**END;**

**Przykład działania:**

**DECLARE**

l\_reservation\_id Reservations**.**reservation\_id**%TYPE;**

**BEGIN**

**SELECT** **MAX(**reservation\_id**)**

**INTO** l\_reservation\_id

**FROM** Reservations**;**

modifyReservationNoPlaces**(**l\_reservation\_id**,** 2**);**

**END;**

**Tabela ReservationsLog po zaktualizowaniu liczby zarezerwowanych miejsc:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | log\_id | reservation\_id | log\_date | status | | no\_places | |
| 1 | 1 | 13 | 2022-03-12 02:05:23 | n | 1 | |
| 2 | 2 | 13 | 2022-03-12 02:10:14 | p | 1 | |
| 3 | 3 | 13 | 2022-03-12 02:13:16 | p | 2 | |

**Tabela 10.2. Tabela ReservationsLog po zaktualizowaniu liczby zarezerwowanych miejsc**

#### **BI\_** **ReservationInsert**

**CREATE** **OR** **REPLACE** **TRIGGER** BI\_ReservationInsert

**BEFORE** **INSERT**

**ON** Reservations

**FOR** **EACH** **ROW**

**DECLARE**

l\_available\_places Reservations**.**no\_places**%TYPE;**

**BEGIN**

**IF** **NOT** doesPersonExist**(:NEW.**person\_id**)** **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'The person with id ' **||** **:NEW.**person\_id **||** ' does not exit'**);**

**END** **IF;**

**IF** **:NEW.**no\_places **<** 1 **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'Cannot book less than 1 place for a trip'**);**

**END** **IF;**

l\_available\_places **:=** getAvailablePlacesNum**(:NEW.**trip\_id**);**

**IF** l\_available\_places **=** 0 **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There are no available places for a trip with id ' **||** **:NEW.**trip\_id**);**

**ELSIF** l\_available\_places **<** **:NEW.**no\_places **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There are only ' **||** l\_available\_places **||**

' places available for a trip with id ' **||** **:NEW.**trip\_id**);**

**END** **IF;**

**END;**

Ten trigger pozwala na wyodrębnienie części funkcjonalności z procedury **addReservation**. Ponieważ nie wprowadza on nowej funkcjonalności, a jedynie pozwala na uproszczenie kodu procedury (nowy kod procedury zamieściłem w kolejnej sekcji), nie zamieszczam tutaj osobnych przykładów działania (działanie triggera wraz ze zmodyfikowaną procedurą jest analogiczne do działania wcześniejszej implementacji procedury **addReservation**).

#### **BU\_** **ReservationStatusUpdate**

**CREATE** **OR** **REPLACE** **TRIGGER** BU\_ReservationStatusUpdate

**BEFORE** **UPDATE**

**OF** status **ON** Reservations

**FOR** **EACH** **ROW**

**DECLARE**

**PRAGMA** **AUTONOMOUS\_TRANSACTION;**

l\_available\_places Reservations**.**no\_places**%TYPE;**

**BEGIN**

**CASE** **:NEW.**status

**WHEN** **:OLD.**status **THEN**

DBMS\_OUTPUT**.**PUT\_LINE**(**'The reservation with id ' **||** **:NEW.**reservation\_id **||**

' has already the status: ' **||** **:NEW.**status**);**

**RETURN;**

**WHEN** 'c' **THEN**

**NULL;**

**WHEN** 'n' **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'Cannot change the status of the reservation with id ' **||**

**:NEW.**reservation\_id **||** ' to: n'**);**

**WHEN** 'p' **THEN**

-- Check if can make cancelled reservation available (paid) again

-- (check if there are enough empty places for a trip)

**IF** **:OLD.**status **=** 'c' **THEN**

l\_available\_places **:=** getAvailablePlacesNum**(:NEW.**trip\_id**);**

**IF** l\_available\_places **<** **:NEW.**no\_places **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'Not enough places available to update the cancelled reservation status'**);**

**END** **IF;**

**END** **IF;**

**ELSE**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'Status: ' **||** **:NEW.**status **||** ' is not a valid reservation status'**);**

**END** **CASE;**

**END;**

Ponownie, z tego samego powodu, co powyżej, nie zamieszczam przykładów.

#### **BU\_** **ReservationNoPlacesUpdate**

**CREATE** **OR** **REPLACE** **TRIGGER** BU\_ReservationNoPlacesUpdate

**BEFORE** **UPDATE**

**OF** no\_places **ON** Reservations

**FOR** **EACH** **ROW**

**DECLARE**

**PRAGMA** **AUTONOMOUS\_TRANSACTION;**

l\_available\_places Reservations**.**no\_places**%TYPE;**

**BEGIN**

l\_available\_places **:=** getAvailablePlacesNum**(:NEW.**trip\_id**);**

**IF** **:NEW.**no\_places **<=** 0 **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'The number of booked places should be greater than 0'**);**

**ELSIF** **:NEW.**no\_places **-** **:OLD.**no\_places **>** l\_available\_places **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There are not enough free places. Max possible number of places to book: ' **||**

**(**l\_available\_places **+** **:OLD.**no\_places**));**

**END** **IF;**

**END;**

Ponownie, z tego samego powodu, co powyżej, nie zamieszczam przykładów.

#### **BU\_TripMaxNoPlacesUpdate**

**CREATE** **OR** **REPLACE** **TRIGGER** BU\_TripMaxNoPlacesUpdate

**BEFORE** **UPDATE**

**OF** max\_no\_places **ON** Trips

**FOR** **EACH** **ROW**

**DECLARE**

l\_booked\_places Reservations**.**no\_places**%TYPE;**

**BEGIN**

l\_booked\_places **:=** getBookedPlacesNum**(:NEW.**trip\_id**);**

**IF** **:NEW.**max\_no\_places **=** **:OLD.**max\_no\_places **THEN**

DBMS\_OUTPUT**.**PUT\_LINE**(**'The trip with id ' **||** **:NEW.**trip\_id **||**

' has already the maximum number of places set to ' **||** **:OLD.**max\_no\_places**);**

**RETURN;**

**END** **IF;**

**IF** **:NEW.**max\_no\_places **<** l\_booked\_places **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'The maximum number of places (' **||** **:NEW.**max\_no\_places **||**

') cannot be lower than the total number of booked places (' **||** l\_booked\_places **||**

') for a trip wit id ' **||** **:NEW.**trip\_id**);**

**END** **IF;**

**END;**

Ponownie, z tego samego powodu, co powyżej, nie zamieszczam przykładów.

### Procedury (wersja 2.)

Poniżej znajdują się 2. wersje procedur (z procedur została usunięta zawartość przeniesiona do triggerów).

#### **addReservation**

**CREATE** **OR** **REPLACE** **PROCEDURE** addReservation**(**

p\_trip\_id Trips**.**trip\_id**%TYPE,**

p\_person\_id People**.**person\_id**%TYPE,**

p\_no\_places Reservations**.**no\_places**%TYPE**

**)**

**AS**

**BEGIN**

**IF** hasTripTakenPlace**(**p\_trip\_id**)** **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'The trip with id ' **||** p\_trip\_id **||** ' took place before'**);**

**END** **IF;**

**INSERT** **INTO** Reservations **(**trip\_id**,** person\_id**,** status**,** no\_places**)**

**VALUES** **(**p\_trip\_id**,** p\_person\_id**,** 'n'**,** p\_no\_places**);**

**COMMIT;**

**END;**

#### **modifyReservationStatus**

**CREATE** **OR** **REPLACE** **PROCEDURE** modifyReservationStatus**(**

p\_reservation\_id Reservations**.**reservation\_id**%TYPE,**

p\_status Reservations**.**status**%TYPE**

**)**

**AS**

**BEGIN**

**IF** **NOT** doesReservationExist**(**p\_reservation\_id**)** **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There is no reservation with id ' **||** p\_reservation\_id **||** ' in the database'**);**

**END** **IF;**

-- If everything is correct, update the reservation status

**UPDATE** Reservations

**SET** status **=** p\_status

**WHERE** reservation\_id **=** p\_reservation\_id**;**

**COMMIT;**

**END;**

**CREATE** **OR** **REPLACE** **PROCEDURE** modifyReservationNoPlaces**(**

p\_reservation\_id Reservations**.**reservation\_id**%TYPE,**

p\_no\_places Reservations**.**no\_places**%TYPE**

**)**

**AS**

**BEGIN**

**IF** **NOT** doesReservationExist**(**p\_reservation\_id**)** **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There is no reservation with id ' **||** p\_reservation\_id **||** ' in the database'**);**

**END** **IF;**

**UPDATE** Reservations

**SET** no\_places **=** p\_no\_places

**WHERE** reservation\_id **=** p\_reservation\_id**;**

**COMMIT;**

**END;**

#### **modifyMaxNoPlaces**

**CREATE** **OR** **REPLACE** **PROCEDURE** modifyMaxNoPlaces**(**

p\_trip\_id Trips**.**trip\_id**%TYPE,**

p\_max\_no\_places Trips**.**max\_no\_places**%TYPE**

**)**

**AS**

**BEGIN**

**IF** **NOT** doesTripExist**(**p\_trip\_id**)** **THEN**

RAISE\_APPLICATION\_ERROR**(**-20001**,** 'There is no trip with id ' **||** p\_trip\_id **||** ' in the database'**);**

**END** **IF;**

**UPDATE** Trips

**SET** max\_no\_places **=** p\_max\_no\_places

**WHERE** trip\_id **=** p\_trip\_id**;**

**END;**