Programowanie proceduralne - widoki, procedury, triggery itp.

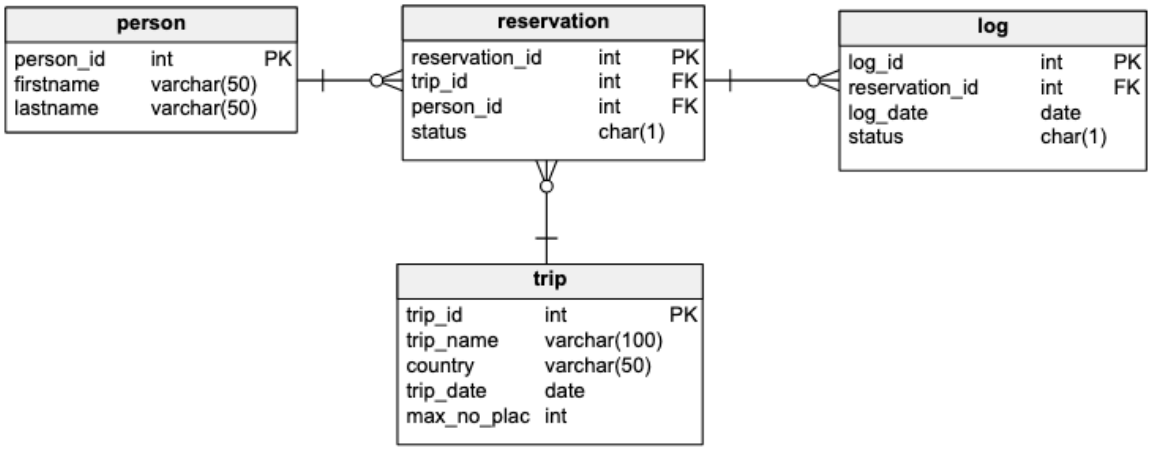
I. Oracle PL/SQL

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

2. Wypełnianie tabele przykładowymi danymi
   4 wycieczki
   10 osób
   10 rezerwacji

   Dane testowe powinny być różnorodne (wycieczki w przyszłości, wycieczki w przeszłości, rezerwacje o różnym statusie itp.) tak, żeby umożliwić testowanie napisanych procedur. W razie potrzeby należy zmodyfikować dane tak żeby przetestować różne przypadki.

3. Tworzenie widoków. Należy przygotować kilka widoków ułatwiających dostęp do danych. Należy zwrócić uwagę na strukturę kodu (należy unikać powielania kodu)

    a) Reservations(country,trip_date,trip_name, firstname, lastname,reservation_id,status)
    b) Trips(country,trip_date, trip_name,no_places, no_available_places)
    c) AvailableTrips(country,trip_date, trip_name,no_places, no_available_places)

…. Wyniki, kod, komentarz

Proponowany zestaw widoków można rozbudować wedle uznania/potrzeb
- np. można dodać nowe/pomocnicze widoki
- np. można zmienić def. widoków, dodając nowe/potrzebne pola

4. Tworzenie procedur/funkcji pobierających dane. Podobnie jak w poprzednim przykładzie należy przygotować kilka procedur ułatwiających dostęp do danych

    a) TripParticipants (trip_id), procedura ma zwracać podobny zestaw danych jak widok Reservations
    b) PersonReservations(person_id), procedura ma zwracać podobny zestaw danych jak widok Reservations
    c) 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.

…. Wyniki, kod, komentarz

5. Tworzenie procedur modyfikujących dane. Należy przygotować zestaw procedur pozwalających na modyfikację danych oraz kontrolę poprawności ich wprowadzania

    a) AddReservation(trip_id, person_id), procedura powinna kontrolować czy wycieczka jeszcze się nie odbyła, i czy sa wolne miejsca
    b) 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)
    c) 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..)

…. Wyniki, kod, komentarz

6. Dodajemy tabelę dziennikującą zmiany statusu rezerwacji

   Log(id, reservation_id, date, status)

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

…. Wyniki, kod, komentarz

7. 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)

…. Wyniki, kod, komentarz

8. 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)

…. Wyniki, kod, komentarz

9. 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)

…. Wyniki, kod, komentarz

10. 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)

…. Wyniki, kod, komentarz

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

II. Postgresql PL/pgSQL

Dla chętnych.

Należy wykonać ćwiczenie przy wykorzystaniu SZBD Postgresql. Podobnie jak w pkt I należy przygotować raport i przesłać kod.

Raport należy uzupełnić własnymi wnioskami stanowiącymi porównanie rozwiązań dostępnych w Oracle PL/SQL oraz Postgres PL/pgSQL. Dodatkowo można pokusić się o porównanie tych rozwiązań z językiem T-SQL

Punktacja za wykonanie pkt. II ćw (wersja dla SZBD Postgresql) - max 2pkt