Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе №5

По дисциплине «Базы данных»

«Триггеры»

Работу выполнил студент группы №43501/4 Кушнаренко П.В. \_\_\_\_\_\_\_\_\_

Работу принял преподаватель Мяснов А.В.\_\_\_\_\_\_\_\_\_\_\_\_\_

Санкт-Петербург

2015

1. Цель работы

Познакомить студентов с возможностями реализации более сложной обработки данных на стороне сервера с помощью хранимых процедур и триггеров.

1. Программа работы
2. Создать два триггера: один триггер для автоматического заполнения ключевого поля, второй триггер для контроля целостности данных в подчиненной таблице при удалении/изменении записей в главной таблице
3. Создать триггер в соответствии с **индивидуальным заданием**, полученным у преподавателя
4. Создать триггер в соответствии с **индивидуальным заданием**, вызывающий хранимую процедуру
5. Выложить скрипт с созданными сущностями в svn
6. Продемонстрировать результаты преподавателю
7. Ход работы
8. Триггер для автоматического заполнения ключевого поля в таблице worker

|  |
| --- |
| CREATE SEQUENCE AUTOINC;  ALTER SEQUENCE AUTOINC RESTART WITH 705;  /\* Old syntax is:  CREATE GENERATOR AUTOINC;  SET GENERATOR AUTOINC TO 705;  \*/  SET TERM ^ ;  CREATE OR ALTER TRIGGER AUTOID\_ADDCOND FOR ADDSERVICECONDITION  ACTIVE BEFORE INSERT POSITION 0  AS  begin  if(new.id is null) then  new.id=gen\_id(autoinc,1);  end  ^  SET TERM ; ^ |

insert into AddServiceCondition (Number,AddServiceId,EmployeeID,StartAS,EndAS,ClientID)

values (202,5,4,'01.08.2013','02.08.2016',4);



Рис.1. Результат эксперимента.

1. Триггер для контроля целостности данных в таблицах rooms и clients при удалении/изменении записей в таблице worker.

|  |
| --- |
| CREATE EXCEPTION modify\_no 'This worker in other tables';  CREATE TRIGGER tr\_modif\_worker\_id FOR clients  BEFORE delete or update  AS  BEGIN  if(old.id  in (select id from rooms  UNION  select clients.id from clients))  then exception modify\_no;  END; |

1. Сделать невозможным бронирование номеров с пересекающимися временными интервалами.

|  |
| --- |
| SET TERM ^ ;  CREATE OR ALTER TRIGGER RIGH\_BRON FOR ROOMCONDITIONS  ACTIVE BEFORE INSERT POSITION 0  AS  declare variable bufdatstart date;  declare variable bufdatend date;  begin  for  select roomconditions.inroom,roomconditions.outroom  from roomconditions  where roomconditions.roomid = new.roomid  into :bufdatstart,:bufdatend  do  begin  if(NOT ( :bufdatstart >= new.outroom OR :bufdatend <= new.inroom ))  then  begin  exception not\_acsesse;  exit;  end  end  end  ^  SET TERM ; ^ |

Проведен эксперимент по проверке работоспособности триггера



Рис.3 Таблица брони номера 108

Попытаемся забронировать номер 108 с пересекающейся датой

insert into RoomConditions (id, Number,RoomID,InRoom,OutRoom,Cost,ClientID,StateArmor)

values (100007,2,108,'24.10.2018','27.10.2018',85000,3,'F');

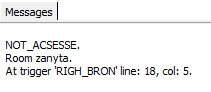


Рис.5. Результат

Попытаемся забронировать номер 108 с не пересекающейся датой

insert into RoomConditions (id, Number,RoomID,InRoom,OutRoom,Cost,ClientID,StateArmor)

values (100007,2,108,'27.10.2018','30.10.2018',85000,3,'F');

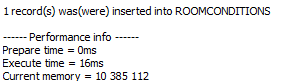




Рис.6. Результат

1. Клиенту, который принес за некоторый период времени более заданного кокличества денег при следующем посещении дарить некоторую доп. Услугу

Создадим процедуру

|  |
| --- |
| SET TERM ^ ;  create or alter procedure COST\_DATE (  INSTRATDATE date,  INENDDATE date,  INID integer)  returns (  OUTCOST integer)  as  begin  select sum(roomconditions.cost)  from roomconditions  where roomconditions.clientid = :inid  and roomconditions.inroom> :instratdate  and roomconditions.inroom< :inenddate into :outcost ;  suspend;  end^  SET TERM ; ^ |

Сам триггер

|  |
| --- |
| SET TERM ^ ;  CREATE OR ALTER TRIGGER FREEADDCOND FOR ROOMCONDITIONS  ACTIVE BEFORE INSERT POSITION 0  AS  declare variable pstart date;  declare variable pend date;  declare variable pcost int;  begin  pstart = '1.1.2014';  pend = '1.1.2016';  execute procedure cost\_date pstart,pend,new.clientid  returning\_values pcost;  if(pcost>600000)  then  insert into AddServiceCondition (Number,AddServiceId,EmployeeID,EndAS,ClientID,cost)  values (869,13,new.clientid,'1.1.2017',new.clientid,0);  end  ^  SET TERM ; ^ |

У клиента с id 108 на данный момент потрачено 568 731 на отель. Забронирует номер на 40 000 и получит бесплатную услугу

insert into RoomConditions (id, Number,RoomID,InRoom,OutRoom,Cost,ClientID,StateArmor)

values (100009,2,108,'24.10.2015','27.10.2015',45000,108,'F');

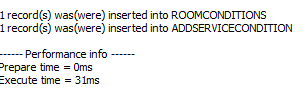




Рис.6. Проверка триггера

Выводы

В данной работе мы познакомились с триггерами. На мой взгляд, триггеры очень сильный инструмент для управления базами данных. Триггеры позволяют автоматизировать процесс заполнения базы данных, так как большинство таблиц в БД связаны. Если в одной из таблиц появляется какая-либо запись, то триггер способен создать запись в другой таблице, изменить запись, или изменить данные. Триггеры позволяют предотвратить добавление данных, которые не вписываются в логику созданной БД. Одной из главных особенностей триггеров является их асинхронность. Они обеспечивают моментальную реакцию системы на какие-либо манипуляции с БД. Это позволяет контролировать корректность вводимых данных непосредственно в момент их добавления. Что исключает необходимость периодических проверок корректности данных в БД.