Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт по лабораторной работе**

**Дисциплина**: Базы данных

**Тема**: Триггеры

Выполнил студент гр. 43501/3 Супронович П.В.

(подпись)

Руководитель Мяснов А.В.

(подпись)

Санкт -Петербург

2016

**Цель работы**

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

**Программа работы**

1. 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;
2. Создать триггер в соответствии с индивидуальным заданием, полученным у преподавателя;
3. Создать триггер в соответствии с индивидуальным заданием, вызывающий хранимую процедуру;
4. Выложить скрипт с созданными сущностями в системе контроля версий;
5. Продемонстрировать результаты преподавателю.

**Индивидуальные задания**

1. Сделать невозможным размещение в одном транспорте одного заказа дважды;
2. При создании заказа назначать ему доступный транспорт с учетом загрузки.

**Ход работы**

1. Триггер для автоматического заполнения ключевого поля

Создадим триггер для таблицы TRANSPORT:

**create** **sequence** tr\_gen**;**

**alter** **sequence** tr\_gen restart **with** 5**;**

**create** **or** **alter** **trigger** tr\_id\_autonic **for** transportation

active **before** **insert** position 0 **as**

**begin**

**if(new.**id **is** **null** **or** **new.**id **=** 0**)** **then** **new.**id **=** **next** **value** **for** tr\_gen**;**

**end**

Следует отметить, что генератор будет увеличиваться даже в том случае, когда добавление записей было отменено командой ROLLBACK. В случае, если создаваемый ID уже существует в таблице, триггер создаёт новый до тех пор, пока не достигнет уникального.

1. Триггер контроля целостности данных в подчиненной таблице при удалении/изменении записей в главной таблице

Так как все связи между таблицами в разработанной схеме БД определены через связи

PK ←FK, а действие на обновление ограничений (CONSTRAINTS) не было определено, то изменить первичный ключ в главной таблице, пока на нее ссылаются подчиненные, невозможно (появится ошибка). Поэтому триггер учитывает только удаление записей. Создадим триггер для таблицы LOAD:

**create** **or** **alter** **trigger** load\_ctl **for** load

active **before** **delete** position 0 **as**

**begin**

**delete** **from** **ord** **where** id\_load **=** **old.**id**;**

**delete** **from** insurance **where** id\_load **=** **old.**id**;**

**end**

1. Сделать невозможным размещение в одном транспорте одного заказа дважды

Для создания данных триггеров предварительно создадим исключение:

**create** **of** **alter** **exception** iscopy 'This entry is already exists!'**;**

**create** **or** **alter** **trigger** tr\_not\_copy **for** transportation

active **before** **insert** position 0 **as**

**begin**

**if((select** id **from** transportation **where** id\_ord **=** **new.**id\_ord **and**

id\_transport **=** **new.**id\_transport**)** **in** **(select** id **from** transportation**))**

**then** **exception** iscopy**;**

**end**

1. При создании заказа назначать ему доступный транспорт с учетом загрузки

**create** **or** **alter** **trigger** load\_stowage **for** **ord**

active **before** **insert** position 0 **as**

**begin**

**execute** **procedure** stowage**;**

**end**

Сама процедура представлена в приложении 1.

**Вывод:**

В ходе выполнения данной лабораторной работы было проведено ознакомление с возможностями реализации более сложной обработки данных на стороне сервера с помощью триггеров, разработаны триггеры автоматического заполнения первичного ключа, контроля целостности БД, остановки добавления записи в таблицу по исключению и триггер, вызывающий процедуру.

Основной областью применения триггеров является контроль целостности базы данных любой сложности. С помощью триггеров существенно упрощаются приложения, т.к. часть логики приложения автоматически активизируется при обновлении БД и выполняется на стороне сервера.

К недостаткам триггеров можно отнести сложность разработки и поддержки, а также скрытая от пользователя функциональность, что может привести к незапланированным последствиям, причины которых потенциально непросто обнаружить. Триггеры влияют на производительность системы, т.к. перед выполнением каждой команды по изменению состояния БД необходимо проверять необходимость запуска триггера.

Таким образом, триггеры — очень полезный инструмент разработки БД, требующий аккуратного к нему подхода.

**Приложение 1. Процедура распределения груза по транспорту**

**create** **or** **alter** **procedure** stowage **as**

**declare** **variable** id\_ord **int;**

**declare** **variable** id\_used\_ord **int;**

**declare** **variable** id\_trans **int;**

**declare** **variable** id\_transportation **int** **=** -1**;**

**declare** **variable** quantity **int** **=** 0**;**

**declare** **variable** load\_quant **int** **=** 0**;**

**declare** **variable** weight **int;**

**declare** **variable** total\_weight **int;**

**declare** **variable** capacity **int;**

**begin**

id\_transportation **=** **(select** **max(**id**)** **from** transportation**);**

**for** **select** id **from** transport **where** transport**.**id **in** **(select** id\_transport **from** transportation**)** **into** **:**id\_trans **do**

**begin**

capacity **=** **(select** capacity **from** transport **where** id **=** **:**id\_trans**);**

**for** **select** id\_ord **from** transportation **where** id\_transport **=** **:**id\_trans **into** **:**id\_used\_ord **do**

**begin**

load\_quant **=** **(select** load\_quant **from** transportation **where** id\_ord **=** **:**id\_used\_ord **and** id\_transport **=** **:**id\_trans**);**

weight **=** **(select** weight **from** load**,** **ord** **where** **ord.**id **=** **:**id\_used\_ord **and** load**.**id **=** **ord.**id\_load**);**

capacity **=** **:**capacity **-** **:**load\_quant **\*** **:**weight**;**

**end**

load\_quant **=** 0**;**

**if(:**capacity **>** 0**)** **then**

**begin**

**if(:**quantity **>** 0**)** **then**

**begin**

weight **=** **:**total\_weight**/:**quantity**;**

**if(:**capacity **>=** **:**total\_weight**)** **then**

**begin**

capacity **=** **:**capacity **-** **:**total\_weight**;**

load\_quant **=** **:**quantity**;**

quantity **=** 0**;**

**end**

**else** **if(:**capacity **>=** **:**weight**)** **then**

**while(:**quantity **>** 0**)** **do**

**begin**

capacity **=** **:**capacity **-** **:**weight**;**

**if(:**capacity **<** 0**)** **then** break**;**

load\_quant **=** **:**load\_quant **+** 1**;**

quantity **=** **:**quantity **-** 1**;**

**end**

**if(:**total\_weight **>** **:**quantity **\*** **:**weight**)** **then**

**begin**

total\_weight **=** **:**quantity **\*** **:**weight**;**

id\_transportation **=** **:**id\_transportation **+** 1**;**

**insert** **into** transportation**(**id**,** id\_transport**,** id\_ord**,** load\_quant**)** **values** **(:**id\_transportation**,** **:**id\_trans**,** **:**id\_ord**,** **:**load\_quant**);**

**end**

**end**

**if(:**quantity **=** 0**)** **then**

**for** **select** id **from** **ord** **where** **ord.**id **not** **in** **(select** id\_ord **from** transportation**)** **into** **:**id\_ord **do**

**begin**

quantity **=** **(select** quantity **from** **ord** **where** id **=** **:**id\_ord**);**

weight **=** **(select** weight **from** load**,** **ord** **where** **ord.**id **=** **:**id\_ord **and** load**.**id **=** **ord.**id\_load**);**

total\_weight **=** **:**quantity **\*** **:**weight**;**

**if(:**capacity **>=** **:**total\_weight**)** **then**

**begin**

capacity **=** **:**capacity **-** **:**total\_weight**;**

load\_quant **=** **:**quantity**;**

quantity **=** 0**;**

**end**

**else** **if(:**capacity **>=** **:**weight**)** **then**

**while(:**quantity **>** 0**)** **do**

**begin**

capacity **=** **:**capacity **-** **:**weight**;**

**if(:**capacity **<** 0**)** **then** break**;**

load\_quant **=** **:**load\_quant **+** 1**;**

quantity **=** **:**quantity **-** 1**;**

**end**

**if(:**total\_weight **>** **:**quantity **\*** **:**weight**)** **then**

**begin**

total\_weight **=** **:**quantity **\*** **:**weight**;**

id\_transportation **=** **:**id\_transportation **+** 1**;**

**insert** **into** transportation**(**id**,** id\_transport**,** id\_ord**,** load\_quant**)** **values** **(:**id\_transportation**,** **:**id\_trans**,** **:**id\_ord**,** **:**load\_quant**);**

**end**

**if(:**capacity **<=** 0**)** **then** break**;**

**end**

**end**

**end**

**for** **select** id **from** transport **where** transport**.**id **not** **in** **(select** id\_transport **from** transportation**)** **into** **:**id\_trans **do**

**begin**

capacity **=** **(select** capacity **from** transport **where** id **=** **:**id\_trans**);**

load\_quant **=** 0**;**

**if(:**quantity **>** 0**)** **then**

**begin**

**if(:**capacity **>=** **:**total\_weight**)** **then**

**begin**

capacity **=** **:**capacity **-** **:**total\_weight**;**

load\_quant **=** **:**quantity**;**

quantity **=** 0**;**

**end**

**else** **if(:**capacity **>=** **:**weight**)** **then**

**while(:**quantity **>** 0**)** **do**

**begin**

capacity **=** **:**capacity **-** **:**weight**;**

**if(:**capacity **<** 0**)** **then** break**;**

load\_quant **=** **:**load\_quant **+** 1**;**

quantity **=** **:**quantity **-** 1**;**

**end**

**if(:**total\_weight **>** **:**quantity **\*** **:**weight**)** **then**

**begin**

total\_weight **=** **:**quantity **\*** **:**weight**;**

id\_transportation **=** **:**id\_transportation **+** 1**;**

**insert** **into** transportation**(**id**,** id\_transport**,** id\_ord**,** load\_quant**)** **values** **(:**id\_transportation**,** **:**id\_trans**,** **:**id\_ord**,** **:**load\_quant**);**

**end**

**end**

**if(:**quantity **=** 0**)** **then**

**for** **select** id **from** **ord** **where** **ord.**id **not** **in** **(select** id\_ord **from** transportation**)** **into** **:**id\_ord **do**

**begin**

quantity **=** **(select** quantity **from** **ord** **where** id **=** **:**id\_ord**);**

weight **=** **(select** weight **from** load**,** **ord** **where** **ord.**id **=** **:**id\_ord **and** load**.**id **=** **ord.**id\_load**);**

total\_weight **=** **:**quantity **\*** **:**weight**;**

**if(:**capacity **>=** **:**total\_weight**)** **then**

**begin**

capacity **=** **:**capacity **-** **:**total\_weight**;**

load\_quant **=** **:**quantity**;**

quantity **=** 0**;**

**end**

**else** **if(:**capacity **>=** **:**weight**)** **then**

**while(:**quantity **>** 0**)** **do**

**begin**

capacity **=** **:**capacity **-** **:**weight**;**

**if(:**capacity **<** 0**)** **then** break**;**

load\_quant **=** **:**load\_quant **+** 1**;**

quantity **=** **:**quantity **-** 1**;**

**end**

**if(:**total\_weight **>** **:**quantity **\*** **:**weight**)** **then**

**begin**

total\_weight **=** **:**quantity **\*** **:**weight**;**

id\_transportation **=** **:**id\_transportation **+** 1**;**

**insert** **into** transportation**(**id**,** id\_transport**,** id\_ord**,** load\_quant**)** **values** **(:**id\_transportation**,** **:**id\_trans**,** **:**id\_ord**,** **:**load\_quant**);**

**end**

**if(:**capacity **<=** 0**)** **then** break**;**

**end**

**end**

**end**