УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Базы Данных»

**Лабораторная работа №5**

Студент

*Мизевич Илья*

*P33202*

Преподаватель

*Шешуков Д. М.*

Санкт-Петербург, 2022 г.

Описание задания

Для выполнения лабораторной работы №5 необходимо:

1. Добавить в ранее созданную базу данных (лр №4) триггеры для обеспечения комплексных ограничений целостности.
2. Реализовать функции и процедуры на основе описания бизнес-процессов, определенных при описании предметной области (лр №1). Должна быть обеспечена проверка корректности вводимых данных для созданных функций и процедур.
3. Необходимо произвести анализ использования созданной базы данных, выявить наиболее часто используемые объекты базы данных, виды запросов к ним. Результаты должны быть представлены в виде текстового описания.
4. На основании полученного описания требуется создать подходящие индексы и доказать, что они будут полезны для представленных в описании случаев использования базы данных.

Описание предметной области

Мы будем разрабатывать информационную систему для дилерского автоцентра, основываясь на открытой информации о таких бизнес-процессах. Дилерский автоцентр предоставляет возможность приобрести новый автомобиль с салона. Так же автомобиль можно купить в разной комплектации и опробовать его на тестдрайве

Код функций, процедур и триггеров

*CREATE PROCEDURE* buying\_car(*IN* eq\_id *INT*, *IN* ncolor *varchar*(20), *IN* nmark *varchar*(20), *IN* nmodel *varchar*(20), *IN* nprice *INT*, *IN* dillers\_id *INT*)  
*language* plpgsql  
*as* $$  
 *DECLARE* newcar\_id *INT*;  
 *BEGIN  
 IF NOT EXISTS*(*SELECT* \* *FROM* dillers *WHERE* id = dillers\_id) *THEN  
 RAISE EXCEPTION* 'Такого диллера нет в базе';  
 *END IF*;  
 *IF NOT EXISTS*(*SELECT* \* *FROM* equipment *WHERE* id = eq\_id) *THEN  
 RAISE EXCEPTION* 'Данная комплектация отсутсвует в базе';  
 *END IF*;  
 *insert into* cars(equipment\_id, color, mark, model, price, availability)  
 *VALUES* (eq\_id,ncolor,nmark,nmodel,nprice,*false*)  
 *RETURNING* id *INTO* newcar\_id;  
 *insert into* list\_purchase(diller\_id, car\_id, status)  
 *VALUES* (dillers\_id,newcar\_id,'В доставке');  
 *END*;  
$$;  
  
*CREATE OR REPLACE FUNCTION* make\_car\_available() *RETURNS TRIGGER AS*$$  
 *BEGIN  
 IF* (*NEW*.availability = *true*) *THEN  
 UPDATE* list\_purchase *SET* status = 'Доставлено'  
 *WHERE* car\_id = *OLD*.id;  
 *END IF*;  
 *RETURN NEW*;  
 *END*$$  
*language* plpgsql;  
  
  
*CREATE OR REPLACE FUNCTION* selling\_car() *RETURNS TRIGGER AS*$$  
 *BEGIN  
 IF EXISTS*(*SELECT* \* *FROM* cars *WHERE* id = *NEW*.car\_id *AND* availability = *true*) *THEN  
 UPDATE* cars *SET* availability = *false  
 WHERE* id = *NEW*.car\_id;  
 *RETURN NEW*;  
 *END IF*;  
 *RAISE EXCEPTION* 'Машина на складе отсутсвует';  
 *END*$$  
*LANGUAGE* plpgsql;  
  
  
*CREATE OR REPLACE FUNCTION* rent() *RETURNS TRIGGER AS*$$  
*DECLARE* status *bool*;  
*BEGIN  
 IF EXISTS*(*SELECT* \* *FROM* test\_drives *WHERE* car\_id = *NEW*.car\_id *AND* start\_date = *NEW*.start\_date) *THEN  
 RAISE EXCEPTION* 'Эта машина уже забранирована на этот день';  
 *END IF*;  
 status := (*SELECT* availability *FROM* cars *WHERE* ID = *NEW*.car\_id);  
 *IF* (status != *true*) *THEN  
 RAISE EXCEPTION* 'Эта машина недоступна';  
 *END IF*;  
 *RETURN NEW*;  
*END*;  
$$  
*LANGUAGE* plpgsql;  
  
*DROP TRIGGER IF EXISTS* rent\_car *ON* test\_drives;  
*DROP TRIGGER IF EXISTS* update\_available *ON* cars;  
*DROP TRIGGER IF EXISTS* sel\_car *ON* cars\_in\_orders;  
  
*CREATE TRIGGER* rent\_car  
 *AFTER UPDATE  
 ON* test\_drives  
 *FOR EACH ROW EXECUTE FUNCTION* rent();  
  
*CREATE TRIGGER* update\_available  
 *AFTER UPDATE  
 ON* cars  
 *FOR EACH ROW EXECUTE FUNCTION* make\_car\_available();  
  
*CREATE TRIGGER* sel\_car  
 *AFTER INSERT  
 ON* cars\_in\_orders  
 *FOR EACH ROW EXECUTE FUNCTION* selling\_car();

Описание наиболее часто используемых сценариев

Самая основная операция, которая будет происходить с базой данных это продажа и закупка автомобилей. Поэтому для корректной работы и корректного отображения наличия товара были предусмотрены следующие триггеры и процедуры.

* Процедура для закупки нового автомобиля. Она добавляет авто в таблицу автомобилей и в список закупок со статусом “В доставке”. пока авто в доставке, оно имеет статус недоступно для продажи (столбец availability в таблице cars)
* Триггер, который не позволяет добавить в заказ машину, которая по каким-то причинам имеет статус available = false.
* Триггер, который не позволяет сдать в прокат машину которая уже арендована на выбранный день или недоступна.
* Триггер, который именит статус доставки при изменении availability у машины на “Доставлено”

Индексы

*CREATE INDEX* car\_mark *ON* cars(mark,model,availability);  
*CREATE INDEX* detail\_name *ON* needed\_detail(*name*);

*CREATE INDEX* clients\_names *ON* clients *USING hash*(first\_name,middle\_name,second\_name);

В таблице car\_mark выполняются запросы с операторами like и по строковым полям имени , названия марки и наличии. Поэтому этой таблице был присвоен стандартный индекс btree. В таблице needed\_detail так же очень часто выполняются запросы с оператором like , поэтому индекс btree там будет оптимальным. Поскольку в таблице с клиентами нас чаще всего интересует полное соответствие поэтому использовали hash.

Вывод

В рамках данной лабораторной работы были созданы функции и триггеры, позволяющие удобно пользоваться базой данных и устраняет возможные ошибки при работе с ней. Также мы оптимизировали нашу лабораторную работу за счет создания индексов для самых часто используемых столбцов таблиц