**Задание 1.**

Для базы данных «Спортивный магазин» из практического задания создайте следующие хранимые процедуры:

■ При добавлении нового товара триггер проверяет его наличие на складе, если такой товар есть и новые данные о товаре совпадают с уже существующими данными, вместо добавления происходит обновление информации о количестве товара;

**DELIMITER //**

**create** **trigger** CheckProductAvailabilityTrigger **after** **insert** **on** products

**for** **each** **row**

**begin**

**declare** ProductId\_ **int**;

**declare** ProductName\_ **varchar**(100);

**declare** ProductType\_ **varchar**(100);

**declare** QuantityInStock\_ **int**;

**declare** CostPrice\_ **decimal**(10,2);

**declare** Manufacturer\_ **varchar**(100);

**declare** SalePrice\_ **decimal**(10,2);

**set** ProductId\_ = **new**.ProductId;

**set** ProductName\_ = **new**.ProductName;

**set** ProductType\_ = **new**.ProductType;

**set** QuantityInStock\_ = **new**.QuantityInStock;

**set** CostPrice\_ = **new**.CostPrice;

**set** Manufacturer\_ = **new**.Manufacturer;

**set** SalePrice\_ = **new**.SalePrice;

**if** **exists**(**select** 1 **from** products **where** ProductId = ProductId\_ **and** ProductName = ProductName\_ **and** ProductTypet = ProductType\_) **then**

**update** products

**SET** QuantityInStock = QuantityInStock + QuantityInStock\_,

CostPrice = **case** **when** (CostPrice\_ **is** **not** **null** **and** CostPrice\_ > 0) **then** CostPrice\_ **else** CostPrice **end**,

Manufacturer = **case** **when** (Manufacturer\_ **is** **not** **null** **and** Manufacturer\_ <> '') **then** Manufacturer\_ **else** Manufacturer **end**,

SalePrice = **case** **when** (SalePrice\_ **is** **not** **null** **and** SalePrice\_ > 0) **then** SalePrice\_ **else** SalePrice **end**

**where** ProductId = ProductId\_ **and** ProductName = ProductName\_ **and** ProductTypet = ProductType\_;

**end** **if**;

**end** //

**DELIMITER ;**

■ При увольнении сотрудника триггер переносит информацию об уволенном сотруднике в таблицу «Архив сотрудников»;

**DELIMITER //**

**create** **trigger** WorkerArchiveTrigger **before** **delete** **on** workers

**for** **each** **row**

**begin**

**insert** **into** ArchiveWorkers (WorkerId, FIOWorker, Post, DateOfEmployment, Gender, Salary, ArchiveDate)

**values** (**old**.WorkerId, **old**.FIOWorker, **old**.Post, **old**.DateOfEmployment, **old**.Gender, **old**.Salary, **now**());

**end** //

**DELIMITER ;**

■ Триггер запрещает добавлять нового продавца, если количество существующих продавцов больше 6.

**DELIMITER //**

**create** **trigger** MaxWorkersCountTrigger **before** **insert** **on** workers

**for** **each** **row**

**begin**

**declare** workers\_count **int**;

**select** **count**(\*) **as** *workers\_count* **from** workers;

**if** workers\_count >= 6 **then**

**signal** **sqlstate** '45000'

**set** message\_text = 'Добавление нового продавца запрещено! Превышено максимальное количество продавцов – 6!';

**end** **if**;

**end** //

**DELIMITER ;**

**Задание 2.**

Для базы данных «Музыкальная коллекция» из практического задания создайте следующие хранимые процедуры:

■ Триггер, не позволяющий добавить уже существующий в коллекции альбом;

**DELIMITER //**

**create** **trigger** CheckExistingAlbum **before** **insert** **on** musicdiscs

**for** **each** **row**

**begin**

**declare** existing\_count **int** **default** 0;

**select** **count**(\*) **as** *existing\_count* **from** musicdiscs **where** NameMusicDisc = **new**.NameMusicDisc;

**if** existing\_count > 0 **then**

**signal** **sqlstate** '45000'

**set** message\_text = 'Музыкальный диск уже есть в коллекции!';

**end** **if**;

**end** //

**DELIMITER ;**

■ Триггер, не позволяющий удалить диск группы TheBeatles;

**DELIMITER //**

**create** **trigger** PreventDeletingTheBeatles **before** **delete** **on** musicdiscs

**for** **each** **row**

**begin**

**declare** NameMusician\_ **varchar**(100)

**select** \* **from** musicdiscs md **join** musicians m **on** md.MusicianId = m.MusicianId

**where** m.NameMusicDisc = NameMusician\_

**if** NameMusicDisc\_ = 'The Beatles' **then**

**signal** **sqlstate** '45000'

**set** message\_text = 'Диск группы The Beatles удалять нельзя!';

**end** **if**;

**end** //

**DELIMITER ;**

■ При удалении диска триггер переносит информацию об удаленном диске в таблицу «Архив»;

**DELIMITER //**

**create** **trigger** MoveDiskToArchive **after** **delete** **on** musicdiscs

**for** **each** **row**

**begin**

**insert** **into** Archive(MusicDiscId, NameMusicDisc, MusicianId, ReleaseDate, StyleId, PublisherId)

**values** (**old**.MusicDiscId, **old**.NameMusicDisc, **old**.MusicianId, **old**.ReleaseDate, **old**.StyleId, **old**.PublisherId);

**end** //

**DELIMITER ;**

■ Триггер, не позволяющий добавлять в коллекцию диски музыкального стиля «Dark Power Pop».

**DELIMITER //**

**create** **trigger** PreventAddingDarkPowerPopDisk **before** **insert** **on** musicdiscs

**for** **each** **row**

**begin**

**declare** **exit** Handler **for** **sqlexception**

**begin**

**get** **diagnostics** **condition** 1 @msg = message\_text;

**select** @msg **AS** Msg;

**end**;

**if** **new**.NameStyle = 'Dark Power Pop' **then**

**signal** **sqlstate** '45000'

**set** message\_text = 'Нельзя добавить диск жанра Dark Power Pop!';

**end** **if**;

**end** //

**DELIMITER ;**