**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«Санкт–Петербургский национальный исследовательский университет**

**информационных технологий, механики и оптики»**

**Факультет информационных технологий и программирования**

**Кафедра информационных систем**

**Выполнил студент группы №M3209**

*Новопашин Алексей Александрович*

**Проверил**

*Повышев Владислав Вячеславович*

**Санкт-Петербург 2018**

**Процедуры**

**CREATE procedure** CreateOrder  
 @CustomerID **int**,  
 @ProductIDs **varchar**(1000),  
 @Address **varchar**(100),  
 @Phone **varchar**(10),  
 @Comment text  
**as begin** -- Добавляем информацию в соответствующую таблицу  
 **insert into** [**Order**] **values** (@Address, @Phone, @Comment, @CustomerID)  
  
 -- Связываем только что созданный заказ с указанными в нём товарами  
 **insert into** OrderProduct  
 **select** OrderID, SplitString.Name **as** ProductID  
 **from** SplitString(@ProductIDs)  
 **inner join** [**Order**] **on** OrderID = @@IDENTITY  
**end  
go**

**CREATE procedure** DisplayDangleCursors **as  
 begin  
 SELECT** creation\_time, cursor\_id, **name**, **c**.session\_id, login\_name  
 **FROM** sys.dm\_exec\_cursors(0) **AS c  
 JOIN** sys.dm\_exec\_sessions **AS** s **ON c**.session\_id = s.session\_id  
 **end**;

**Функции**

**CREATE** FUNCTION dbo.IsLoginTaken(@Login **VARCHAR**(26))  
 RETURNS  
 **bit AS  
 BEGIN  
 DECLARE** @IsTaken **bit** if **exists**(**select** Login **from** Customer **where** Login = @Login)  
 **set** @IsTaken = 1  
 **else set** @IsTaken = 0;  
  
 return @IsTaken  
 **END  
go**

**CREATE** FUNCTION dbo.SplitString(@stringToSplit **VARCHAR**(**MAX**))  
 RETURNS  
 @returnList **TABLE**([**Name**] [nvarchar](500))  
**AS  
 BEGIN  
  
 DECLARE** @name NVARCHAR(255)  
 **DECLARE** @pos **INT** WHILE CHARINDEX(',', @stringToSplit) > 0  
 **BEGIN  
 SELECT** @pos = CHARINDEX(',', @stringToSplit)  
 **SELECT** @name = *SUBSTRING*(@stringToSplit, 1, @pos - 1)  
  
 **INSERT INTO** @returnList **SELECT** @name  
  
 **SELECT** @stringToSplit = *SUBSTRING*(@stringToSplit, @pos + 1, LEN(@stringToSplit) - @pos)  
 **END  
  
 INSERT INTO** @returnList **SELECT** @stringToSplit  
  
 RETURN  
 **END  
go**

**Триггеры**

**CREATE** trigger CheckLoginAvailability  
 **on** Customer  
 instead **of INSERT as  
 begin  
 DECLARE** @Login **varchar**(255)  
 **select** @Login = Login **from** inserted  
  
 if dbo.IsLoginTaken(@Login) = 1  
 throw 50500, 'Login already taken!', 1;  
  
 **insert** \* **from** inserted  
 **end  
go**

**CREATE** trigger CheckLoginAvailabilityUpdate  
 **on** Customer  
 instead **of UPDATE as  
 begin  
 DECLARE** @Login **varchar**(255)  
 **DECLARE** @CustomerID **int  
 select** @Login = Login, @CustomerID = inserted.CustomerID **from** inserted  
  
 **select** \* **from** inserted  
 **select** CustomerID **from** Customer **where** CustomerID = @CustomerID **and** Login = @Login  
  
 if dbo.IsLoginTaken(@Login) = 1 **and not** (**exists**(**select** CustomerID **from** Customer **where** CustomerID = @CustomerID **and** Login = @Login))  
 throw 50500, 'Login already taken!', 1;  
  
 **update** Customer  
 **set** Customer.FullName = inserted.FullName,  
 Customer.Login = inserted.Login,  
 Customer.Password = inserted.Password  
 **from** inserted  
 **where** Customer.CustomerID = inserted.CustomerID  
 **end  
go**

**CREATE** trigger WarningAccountDeletion  
 **on** Customer  
 after **DELETE as  
 begin  
 DECLARE** @CustomerID **int  
 DECLARE** @FullName **varchar**(40)  
 **DECLARE** @Login **varchar**(26)  
 **DECLARE** @Password **varchar**(26)  
  
 **select** @CustomerID = deleted.CustomerID,  
 @FullName = deleted.FullName,  
 @Login = deleted.Login,  
 @Password = deleted.Password  
 **from** deleted  
  
 **delete from** Customer  
 **where** CustomerID = @CustomerID  
  
 print FORMATMESSAGE('Successful Account deletion: %i %s %s %s', @CustomerID, @FullName, @Login, @Password)  
 **end  
go**

**Курсоры**

**create procedure** Customers @cursor **CURSOR VARYING OUTPUT  
as  
 SET** @cursor = **CURSOR** FORWARD\_ONLY DYNAMIC **FOR SELECT** FullName, Login, Password **FROM** Customer  **OPEN** @cursor   
**go**

**declare** AllCustomers **cursor local** forward\_only static  
**for select** FullName, Phone, Address  
 **from** Customer  
 **left join** [**Order**] O **on** Customer.CustomerID = O.CustomerID  
  
**DECLARE** @FullName **varchar**(40)  
 **DECLARE** @Phone **VARCHAR**(8)  
**DECLARE** @Address **VARCHAR**(80)  
  
**DECLARE** @Message **varchar**(**max**)  
  
**open** AllCustomers;  
  
**fetch next from** AllCustomers  
**into** @FullName, @Phone, @Address  
  
WHILE @@FETCH\_STATUS = 0  
 **begin** print FORMATMESSAGE('Fullname: %s; Phone: %s; Address: %s;', @FullName, @Phone, @Address)  
 **FETCH next from** AllCustomers  
 **into** @FullName, @Phone, @Address  
 **end  
  
close** AllCustomers;  
**deallocate** AllCustomers;

**Представления**

**CREATE VIEW** CustomerOrder  
 **AS  
 SELECT** FullName, Address, Phone, Comment  
 **from** Customer  
 **left join** [**Order**] O **on** Customer.CustomerID = O.CustomerID  
**go**

**CREATE VIEW** HitProducts  
 **AS  
 select** P.ProductID, *count*(\*) **as** Purchases  
 **from** OrderProduct  
 **left join** Product P **on** OrderProduct.ProductID = P.ProductID  
 **group by** P.ProductID  
**go**

**CREATE VIEW** MostReviewableProducts  
 **AS  
 select** P.ProductID, P.Title, *count*(\*) **as Count  
 from** Comment  
 **left join** Product P **on** Comment.ProductID = P.ProductID  
 **group by** P.ProductID, P.Title  
**go**