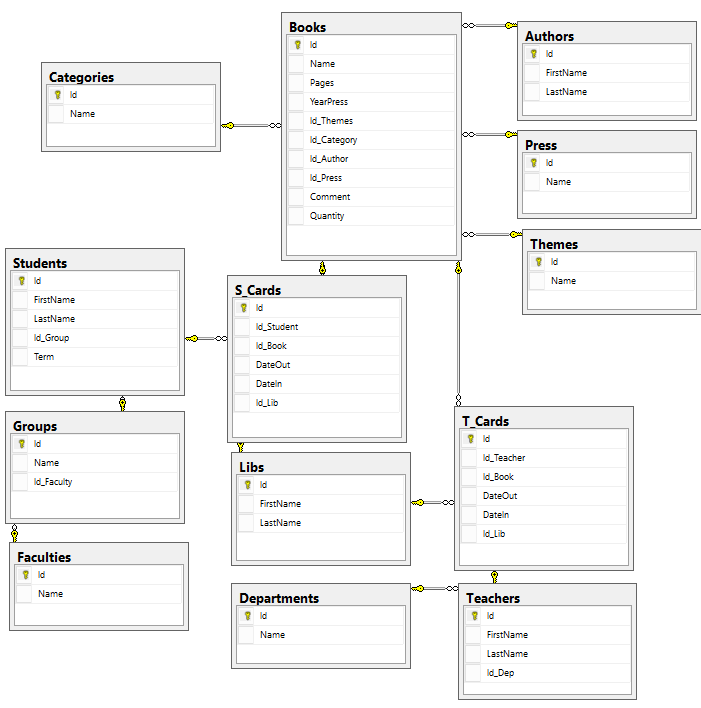
***Группа ИТ-2707. Леженко С.А.***

use library;



--Написать такие хранимые процедуры:

--Написать хранимую процедуру, выводящую на экран список студентов, не сдавших книги.

CREATE PROCEDURE getStudentsNotPassBooks

AS

BEGIN

SELECT FirstName + ' ' + LastName AS 'Name',

Name AS 'Group'

FROM Students INNER JOIN Groups

ON Students.Id\_Group=Groups.Id

WHERE Students.Id IN

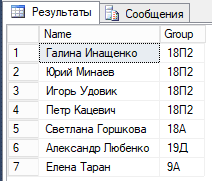
(SELECT Id\_Student

FROM S\_Cards

WHERE DateIn IS NULL);

END;

EXECUTE getStudentsNotPassBooks;



--Написать хранимую процедуру, возвращающую имя и фамилию библиотекаря, выдавшего наибольшее кол-во книг.

CREATE PROCEDURE getLibrarianWhoIssuedMoreBooks

AS

BEGIN

DECLARE @tempLibs TABLE(Id\_Lib int);

INSERT INTO @tempLibs

SELECT Id\_Lib AS 'Id\_Lib'

FROM S\_Cards

UNION ALL

SELECT Id\_Lib

FROM T\_Cards;

DECLARE @LibsWithCounts TABLE(Id\_Lib int, CountBooks int);

INSERT INTO @LibsWithCounts

SELECT Id\_Lib, COUNT(Id\_Lib)

FROM @tempLibs

GROUP BY Id\_Lib;

SELECT FirstName + ' ' + LastName AS 'Librarian'

FROM Libs

WHERE Id =

(SELECT Id\_Lib

FROM @LibsWithCounts

WHERE CountBooks =

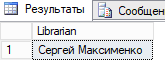
(SELECT MAX(CountBooks)

FROM @LibsWithCounts)

)

END;

EXECUTE getLibrarianWhoIssuedMoreBooks;



--Написать хранимую процедуру, подсчитывающую факториал числа. (5! = 1\*2\*3\*4\*5 = 120) (0! = 1)

--(факториала отрицательного числа не существует).

CREATE PROCEDURE getFactorial

@num int

AS

BEGIN

DECLARE @res int;

IF(@num=0)

BEGIN

SET @res=1;

END

ELSE

BEGIN

IF(@num<0)

BEGIN

PRINT 'The factorial of a negative number does not exist'

END

ELSE

BEGIN

DECLARE @tmp int;

SET @tmp = @num-1;

EXECUTE @res = getFactorial @tmp;

SET @res = @res \* @num;

END

END

RETURN @res;

END

DECLARE @Fact int;

EXECUTE @Fact=getFactorial 5;

SELECT 'Factorial 5 =', @Fact;



--Написать такие функции:

--Функцию, возвращающую кол-во студентов, которые не брали книги.

CREATE FUNCTION getCountStudentsNotTakeBooks

()

RETURNS int

AS

BEGIN

DECLARE @res int;

SELECT @res=COUNT(Id)

FROM Students

WHERE NOT EXISTS

(SELECT Id\_Student

FROM S\_Cards

WHERE Students.Id=S\_Cards.Id\_Student)

RETURN @res;

END

PRINT dbo.getCountStudentsNotTakeBooks();



--Функцию, возвращающую минимальное из трех переданных параметров.

CREATE FUNCTION getMin

(@n1 int, @n2 int, @n3 int)

RETURNS int

AS

BEGIN

DECLARE @res int;

IF(@n1<@n2)

SET @res=@n1;

ELSE

SET @res=@n2;

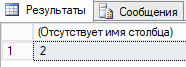
IF(@n3<@res)

SET @res=@n3;

RETURN @res;

END

SELECT dbo.getMin(15,4,2);



--Функцию, которая принимает в качестве параметра двухразрядное число и определяет

--какой из разрядов больше, либо они равны. (используйте % - деление по модулю. Например: 57%10=7)

CREATE FUNCTION getMaxRazr

(@num int)

RETURNS int

AS

BEGIN

DECLARE @res int;

IF(@num%10 > @num/10)

SET @res = @num%10

ELSE

SET @res = @num/10;

RETURN @res;

END

PRINT dbo.getMaxRazr(57);



--Функцию, возвращающую кол-во взятых книг по каждой из групп и по каждой из кафедр (departments).

CREATE FUNCTION getCountBooksByGroupOrDepartments

()

RETURNS TABLE

AS

RETURN(

(SELECT g.Name AS 'Name', COUNT(c.Id) AS 'Count'

FROM Groups AS g, Students As s, S\_Cards AS c

WHERE g.Id=s.Id\_Group AND s.Id=c.Id\_Student

GROUP BY g.Name)

UNION

(SELECT d.Name AS 'Name', COUNT(c.Id) AS 'Count'

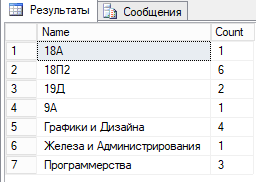
FROM Departments AS d, Teachers AS t, T\_Cards AS c

WHERE d.Id=t.Id\_Dep AND t.Id=c.Id\_Teacher

GROUP BY d.Name)

)

SELECT \* FROM dbo.getCountBooksByGroupOrDepartments();



--Функцию, возвращающую список книг, отвечающих набору критериев

--(например, имя автора, фамилия автора, тематика, категория),

--и отсортированный по номеру поля, указанному в 5-м параметре, в направлении, указанном в 6-м параметре.

CREATE FUNCTION getBooksByAuthorThemeCategory

(@AuthorName varchar(50),

@AuthorSurname varchar(50),

@Theme varchar(50),

@Category varchar(50),

@SortCol int,

@SortDirection int)

RETURNS @Books TABLE

(

[Id] [int],

[Name] [varchar](100),

[Pages] [int],

[YearPress] [int],

[Id\_Themes] [int],

[Id\_Category] [int],

[Id\_Author] [int],

[Id\_Press] [int],

[Comment] [varchar](50),

[Quantity] [int]

)

AS

BEGIN

DECLARE @res TABLE

(

[Id] [int],

[Name] [varchar](100),

[Pages] [int],

[YearPress] [int],

[Id\_Themes] [int],

[Id\_Category] [int],

[Id\_Author] [int],

[Id\_Press] [int],

[Comment] [varchar](50),

[Quantity] [int]

);

INSERT INTO @res(Id,Name,Pages,YearPress,Id\_Themes,Id\_Category,Id\_Author,Id\_Press,Comment,Quantity)

(SELECT \*

FROM Books AS b

WHERE b.Id\_Category =

(SELECT Id

FROM Categories

WHERE Name LIKE @Category)

AND

b.Id\_Author =

(SELECT Id

FROM Authors

WHERE FirstName LIKE @AuthorName

AND LastName LIKE @AuthorSurname)

AND

b.Id\_Themes =

(SELECT Id

FROM Themes

WHERE Name Like @Theme)

);

IF(@SortDirection=0)

BEGIN

INSERT INTO @Books(Id,Name,Pages,YearPress,Id\_Themes,Id\_Category,Id\_Author,Id\_Press,Comment,Quantity)

SELECT Id,Name,Pages,YearPress,Id\_Themes,Id\_Category,Id\_Author,Id\_Press,Comment,Quantity

FROM @res

ORDER BY (CASE @SortCol

WHEN 1 THEN Id

WHEN 2 THEN Name

WHEN 3 THEN Pages

WHEN 4 THEN YearPress

WHEN 5 THEN Id\_Themes

WHEN 6 THEN Id\_Category

WHEN 7 THEN Id\_Author

WHEN 8 THEN Id\_Press

WHEN 9 THEN Comment

WHEN 10 THEN Quantity

ELSE NULL

END) ASC

END

ELSE

BEGIN

IF(@SortDirection=1)

BEGIN

INSERT INTO @Books(Id,Name,Pages,YearPress,Id\_Themes,Id\_Category,Id\_Author,Id\_Press,Comment,Quantity)

SELECT Id,Name,Pages,YearPress,Id\_Themes,Id\_Category,Id\_Author,Id\_Press,Comment,Quantity

FROM @res

ORDER BY (CASE @SortCol

WHEN 1 THEN Id

WHEN 2 THEN Name

WHEN 3 THEN Pages

WHEN 4 THEN YearPress

WHEN 5 THEN Id\_Themes

WHEN 6 THEN Id\_Category

WHEN 7 THEN Id\_Author

WHEN 8 THEN Id\_Press

WHEN 9 THEN Comment

WHEN 10 THEN Quantity

ELSE NULL

END) DESC

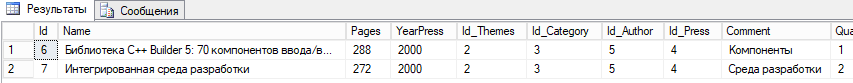
END

END

RETURN

END

SELECT \* FROM getBooksByAuthorThemeCategory('Алексей','Архангельский','Программирование','C++ Builder',2,0);



--Функцию, которая возвращает список библиотекарей и кол-во выданных каждым из них книг.

CREATE FUNCTION getLibrarianWithCountBooks ()

RETURNS @res TABLE(Name varchar(100), CountBooks int)

AS

BEGIN

INSERT INTO @res

SELECT FirstName + ' ' + LastName AS 'Name',

COUNT(S\_Cards.Id)+COUNT(T\_Cards.Id) AS 'CountBooks'

FROM (Libs INNER JOIN T\_Cards ON Libs.Id=T\_Cards.Id\_Lib)

INNER JOIN S\_Cards ON Libs.Id=S\_Cards.Id\_Lib

GROUP BY Libs.FirstName, Libs.LastName;

RETURN

END

SELECT \* FROM getLibrarianWithCountBooks();

