***Группа ИТ-2707. Леженко С.А.***

1. При помощи Enterprise manager создать базу данных Shop. В ней должна храниться информация о товаре, его название, цена, кол-во на складе, ID категории товара, ID продавца, который продал товар и кол-во которое он продал. Должна быть таблица с продавцами (имя, фамилия, возраст, телефон) и таблица с категориями (название категории). Необходимо их связать и в каждую из них, при помощи Query Analizer записать минимум по 3 записи.

--Создаём базу

CREATE DATABASE Shop ON PRIMARY

(name = Shop,

filename = 'C:\Program Files\Microsoft SQL Server\MSSQL12.SQLEXPRESS1\MSSQL\DATA\Shop.mgf',

size = 12MB,

maxsize = 20MB,

filegrowth = 2MB)

LOG ON

(name = Shop\_Log,

filename = 'C:\Program Files\Microsoft SQL Server\MSSQL12.SQLEXPRESS1\MSSQL\DATA\Shop\_Log.ldf',

size = 2MB,

maxsize = 12MB,

filegrowth = 1Mb);

--Подключаемся к ней

USE Shop;

--Создаём таблицу категорий

CREATE TABLE Categories(

id int identity(1,1) not null

Constraint PK\_Categories primary key(id),

name nvarchar(50)

Constraint UniCat unique(name));

--Заполняем таблицу категорий

INSERT INTO Categories(name)

VALUES('CPU'),

('Motherboard'),

('Monitor'),

('VideoCard');

--Создаём таблицу продавцов

CREATE TABLE Sellers(

id int identity(1,1) not null

Constraint PK\_Producers primary key(id),

firstname nvarchar(50) not null,

lastname nvarchar(50) not null,

age int,

tel varchar(15) not null);

--Заполняем таблицу продавцов

INSERT INTO Sellers(firstname, lastname, age, tel)

VALUES('Petro','Kuzmenko',24,'0504542365'),

('Kiril','Mefodiy',53,'0612548653'),

('Vasiliy','Zradnuk',34,'0675623684');

--Создаём таблицу товаров

CREATE TABLE Products(

id int identity(1,1) not null

Constraint PK\_Products primary key(id),

name varchar(255) not null,

price float,

quantity int,

idCategory int not null

Constraint FK\_Category foreign key(idCategory)

references Categories(id));

--Создаём функцию для получения idCategory по её имени

CREATE FUNCTION idFromCategoryName

(@name nvarchar(50))

RETURNS int

AS

BEGIN

RETURN (SELECT id

FROM Categories

WHERE name LIKE @name)

END;

--Заполняем таблицу товаров

INSERT Products(name,price,quantity,idCategory)

VALUES

('Intel Core i3',4253,10,dbo.idFromCategoryName('CPU')),

('AMD FX5422',5623,8,dbo.idFromCategoryName('CPU')),

('Philips 226CL',6532,4,dbo.idFromCategoryName('Monitor'));

--Создаём таблицу продаж

CREATE TABLE Sales(

id int identity(1,1) not null

Constraint PK\_Sale primary key(id),

idProduct int not null

Constraint FK\_Product foreign key(idProduct)

references Products(id),

idSeller int not null

Constraint FK\_Seller foreign key(idSeller)

references Sellers(id),

quantity int not null,

price float not null,

saleDate date not null);

--Заполняем таблицу продаж

INSERT Sales(idProduct,idSeller,quantity,price,saleDate)

VALUES(1,1,2,5832,'12/01/2017'),

(1,2,1,5563,'01/02/2017'),

(2,3,2,5786,'05/04/2017'),

(3,2,2,6853,'04/03/2017');

1. Используя Shop написать запросы:

-вычитать всю информацию о работе и работниках магазина

-вычитать всю информацию о работниках магазина

CREATE VIEW ShopInfo

AS

SELECT saleDate AS 'Date',

Products.name AS 'Product',

Sales.quantity AS 'Quantity',

Sales.quantity\*Sales.price AS 'Summ',

Sellers.firstname+' '+Sellers.lastname AS 'Seller'

FROM (Sales INNER JOIN Products

ON Sales.idProduct=Products.id)

INNER JOIN Sellers

ON Sales.idSeller=Sellers.id;

CREATE VIEW SellersInfo

AS

SELECT firstname+' '+lastname AS 'Seller',

age AS 'Age',

tel AS 'Telephone'

FROM Sellers;

1. Используя Books (многотабличную базу данных) написать запросы:

--Создаём базу

CREATE DATABASE Books ON PRIMARY

(name = Books,

filename = 'C:\Program Files\Microsoft SQL Server\MSSQL12.SQLEXPRESS1\MSSQL\DATA\Books.mgf',

size = 12MB,

maxsize = 20MB,

filegrowth = 2MB)

LOG ON

(name = Books\_Log,

filename = 'C:\Program Files\Microsoft SQL Server\MSSQL12.SQLEXPRESS1\MSSQL\DATA\Books\_Log.ldf',

size = 2MB,

maxsize = 12MB,

filegrowth = 1Mb);

--Подключаемся

USE Books;

--Импортируем с помощью мастера таблицу books из файла базы

--Создаём таблицу Издательств

CREATE TABLE Press(

id int identity(1,1) not null

Constraint PK\_Press primary key(id),

izd varchar(40));

--Заполняем таблицу Издательств

INSERT INTO Press(izd)

(SELECT DISTINCT izd

FROM books

WHERE name is not null);

--Создаём таблицу Тематик

CREATE TABLE Themes(

id int identity(1,1) not null

Constraint PK\_Themes primary key(id),

name varchar(40));

--Заполняем таблицу тематик

INSERT INTO Themes(name)

(SELECT DISTINCT Themes

FROM books

WHERE Themes is not null);

--Создаем таблицу Категорий

CREATE TABLE Categories(

id int identity(1,1) not null

Constraint PK\_Cat primary key(id),

name varchar(40));

--Заполняем таблицу категорий

INSERT INTO Categories(name)

(SELECT DISTINCT Category

FROM books

WHERE Category is not null);

--Добавляем в таблицу books поля для внешних индексов

ALTER TABLE books ADD idPress int;

ALTER TABLE books ADD idTheme int;

ALTER TABLE books ADD idCategory int;

--Заполняем индексы

UPDATE books

SET books.idPress=Press.id

FROM books INNER JOIN Press

ON books.Izd=Press.name;

UPDATE books

SET books.idTheme=Themes.id

FROM books INNER JOIN Themes

ON books.Themes=Themes.name;

UPDATE books

SET books.idCategory=Categories.id

FROM books INNER JOIN Categories

ON books.Category=Categories.name;

--Удаляем не нужные уже поля из books

ALTER TABLE books DROP COLUMN Izd;

ALTER TABLE books DROP COLUMN Themes;

ALTER TABLE books DROP COLUMN Category;

--все издательства, которые издали новинки с ценой >40р.

SELECT name AS 'Press'

FROM Press

WHERE id IN

(SELECT idPress

FROM books

WHERE books.New=1 AND books.Price>40);

--Вычитать все тематики, у которых не указана категория,

-- при этом из результата исключить повторения.

SELECT DISTINCT name

FROM Themes

WHERE id IN

(SELECT idTheme

FROM books

WHERE idCategory IS NULL);

--Вытащить название учебников, которые издавались издательством

-- 'BHV', а первая буква тематики находится в диапазоне от Е до К.

SELECT Name

FROM books

WHERE (Name BETWEEN 'Е%' AND 'К%')

AND idPress IN

(SELECT id

FROM Press

WHERE name LIKE '%BHV%');

--Вычитать название учебников, в названии издательств которых

-- более 3 слов.

SELECT Name

FROM books

WHERE idPress IN

(SELECT id

FROM Press

WHERE LEN(name)-LEN(REPLACE(name, ' ', '')) > 1);

--Вычитать название учебников, в названии издательств которых

-- ровно 3 слова.

SELECT Name

FROM books

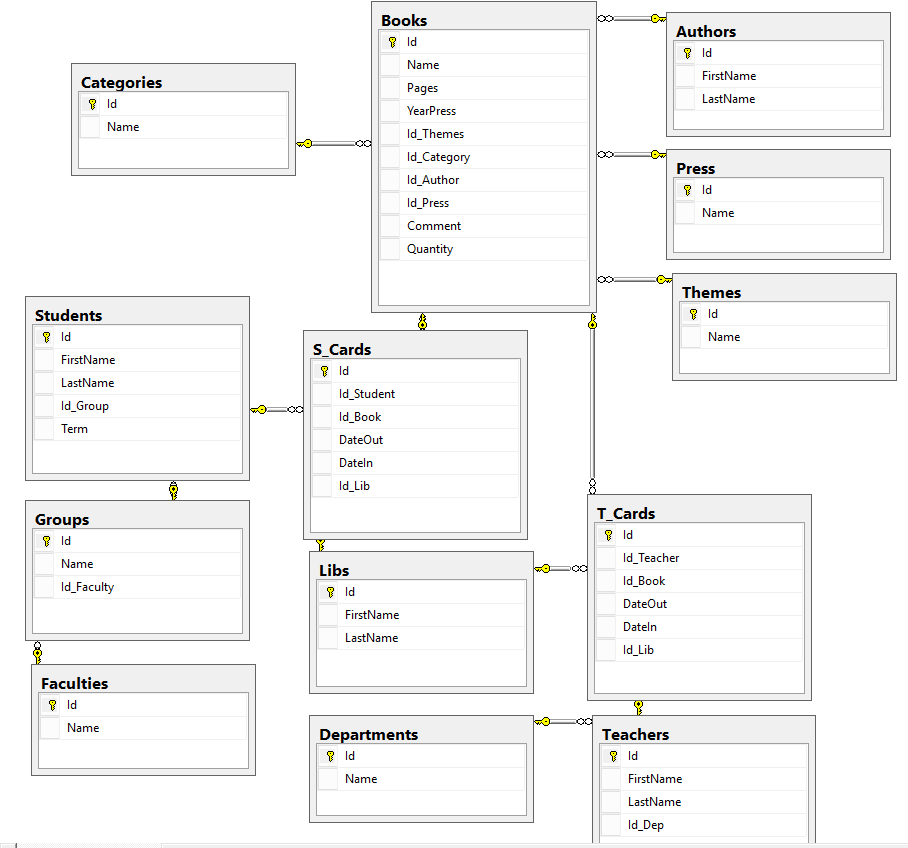
WHERE idPress IN

(SELECT id

FROM Press

WHERE LEN(name)-LEN(REPLACE(name, ' ', '')) = 2);

1. Написать следующие запросы, используя базу данных Library



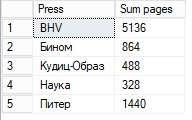
-- Показать издательства и сумму страниц по каждому из них

SELECT Press.Name AS 'Press', SUM(Pages) AS 'Sum pages'

FROM Books INNER JOIN Press

ON Press.Id=Id\_Press

GROUP BY Press.Name;



-- Показать общее кол-во книг, взятых студентами факультета 'Программирования'

SELECT COUNT(Id) AS 'Count books'

FROM S\_Cards

WHERE Id\_Student IN

(SELECT Id

FROM Students

WHERE Id\_Group IN

(SELECT Groups.Id

FROM Groups

INNER JOIN Faculties

ON Groups.Id\_Faculty=Faculties.Id

WHERE Faculties.Name LIKE 'Программирования'));



-- Вывести кол-во книг и сумму страниц этих книг по каждому

--из издательств 'Питер','Наука' и 'Кудиц-Образ'

SELECT Press.Name AS 'Press',

COUNT(Books.Id) AS 'Count books',

SUM(Pages) AS 'Sum pages'

FROM Books

INNER JOIN Press

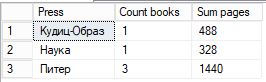
ON Books.Id\_Press=Press.Id

WHERE Press.Name LIKE 'Питер'

OR Press.Name LIKE 'Наука'

OR Press.Name LIKE 'Кудиц-Образ'

GROUP BY Press.Name;



-- Вывести информацию о книге по программированию с наибольшим количеством страниц

SELECT Books.Name AS 'Book',

YearPress AS 'Year',

Categories.Name AS 'Category',

Authors.FirstName+' '+Authors.LastName AS 'Author',

Press.Name AS 'Press',

Pages AS 'Pages',

Comment AS 'Comment'

FROM (((Books

INNER JOIN Categories ON Books.Id\_Category=Categories.Id)

INNER JOIN Authors ON Books.Id\_Author=Authors.Id)

INNER JOIN Press ON Books.Id\_Press=Press.Id)

WHERE Books.Pages =

(SELECT MAX(Pages)

FROM Books

WHERE Id\_Themes =

(SELECT Id

FROM Themes

WHERE Name LIKE 'Программирование'))

--На случай если попадётся такоеже количество страниц в тругой тематике

--снова отбираем по теме

AND Id\_Themes =

(SELECT Id

FROM Themes

WHERE Name LIKE 'Программирование');



--Вывести на экран кол-во взятых книг по каждой из кафедр

SELECT COUNT(T\_Cards.Id) AS 'Count books',

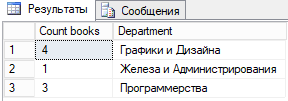
Departments.Name AS 'Department'

FROM (Teachers

INNER JOIN T\_Cards ON Teachers.Id = T\_Cards.Id\_Teacher)

INNER JOIN Departments ON Teachers.Id\_Dep = Departments.Id

GROUP BY Departments.Name;



--Показать издательства и самую старую книгу для каждого из них

SELECT Press.Name AS 'Press',

Books.Name AS 'Book',

Books.YearPress AS 'Year'

FROM Press INNER JOIN Books ON Press.Id=Books.Id\_Press

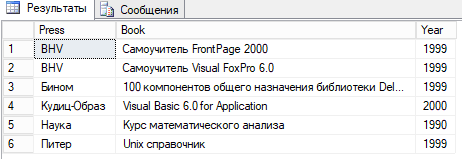
WHERE YearPress =

(SELECT MIN(YearPress)

FROM Books

WHERE Books.Id\_Press=Press.Id)

GROUP BY Press.Name, Books.Name, Books.YearPress



--Показать книги, которые брали и преподаватели и студенты (исключить повторения)

SELECT Books.Name AS 'Books'

FROM Books

WHERE EXISTS

(SELECT Id

FROM S\_Cards

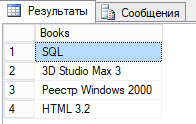
WHERE Id\_Book=Books.Id)

AND EXISTS

(SELECT Id

FROM T\_Cards

WHERE Id\_Book=Books.Id);



--Показать название книги с максимальным кол-вом страниц по каждому

--из издательств.(с начала найти максимум по издательствам, после чего

--вложить этот запрос внутрь поиска по книгам)

SELECT Press.Name AS 'Press',

Books.Name AS 'Book'

FROM Books INNER JOIN Press ON Books.Id\_Press=Press.Id

WHERE Pages =

(SELECT MAX(Pages)

FROM Books

WHERE Id\_Press=Press.Id)

GROUP BY Press.Name, Books.Name;

