ВЕЛИКОТЪРНОВСКИ УНИВЕРСИТЕТ "СВ. СВ. КИРИЛ И МЕТОДИЙ"



ФАКУЛТЕТ ,,МАТЕМАТИКА И ИНФОРМАТИКА”

**КУРСОВА РАБОТА**

по

учебната дисциплина

БАЗА ОТ ДАННИ

**Тема:** База от данни за библиотека.

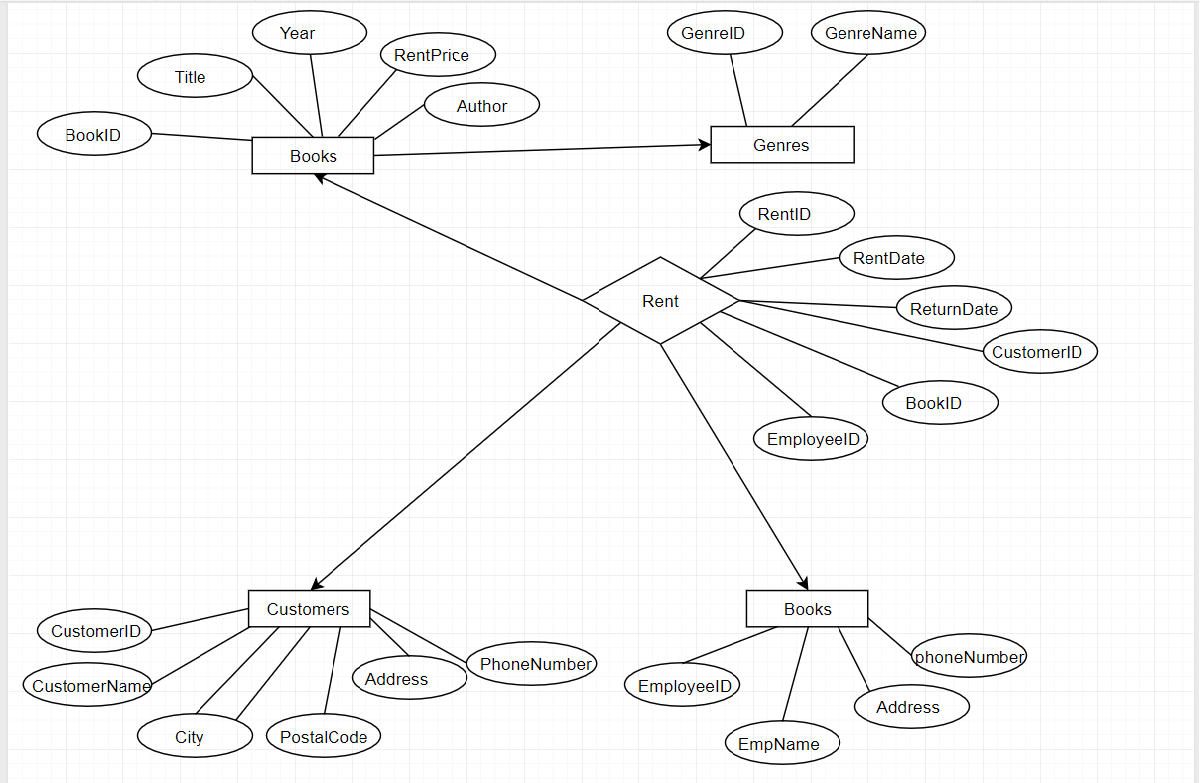
**Изготвил:**Стоян Иванов Стоянов, специалност:Софтуерно инженерство, факултетен №:1809010142

**Проверил:** доц. д-р Цветанка Любомирова Георгиева-Трифонова

**1.Описание на предметната област**

Предметната област на настоящия курсов проект обхваща работата на библиотека, занимаваща се с отдаването под наем на книги за определено време.Всеки служител при даването под наем на книга записва данните на клиента и периода на ползване на книгата.Всички клиенти имат правото на вземане на повече от една книга, но с различен период за прочитане на книгата.Времето за прочитане на книгата се определя от служителя на произволен принцип.

По този начин библиотеката поддържа база от данни за книгите отдадени от своите служители и данните на клиетните.С информацията за клиетна е възможно да се установи връзка с клиента за да му бъде напомнено, че трябва да върне книгата преди да изтече срока за връщане.С базата от данни работи само служител, назначен от управителя на библиотеката.

**2.**Е-R модел на базата от данни

Employees

**3**.Описание на таблиците

Таблица ,,Employees” има данни за служител:име на служител, адрес и телефонен номер.За телефонен номер са отделени 10 цифри.

Таблица ,,Rent” има данни за датата на взетата книга и датата за връщане на книгата.Rent съединява обекти от класовете Books, Customers и Employees и описва наемането на книги от клиент, осъществено от даден служител.

Таблица ,,Customers” има данни за клиента: името на клиента, града, адреса, пощенски код и телефонен номер.За името на клиента имаме право само на главни и малки букви при въвеждане на име.

Таблица ,,Books” има данни за книгата: заглавието на книгата, годината, цената на наем и автора. Класа Books с обекти Genres представя книгите, отнасящи се към съответните жанрове.

Таблица ,,Genre” има данни за името на жанра.

4.Създаване на таблиците и описание на заявките

--Създаване на таблица Employees

create table Employees

(EmployeeID int NOT NULL PRIMARY KEY IDENTITY,

EmpName varchar(50) NOT NULL,

address varchar(50) NOT NULL,

phoneNumber varchar(50) NOT NULL)

--Вкарване на данни в таблица Employees, чрез команда INSERT INTO.

INSERT INTO Employees(EmpName, address, phoneNumber)

VALUES ('Aleksandra', 'Oborishe', '0896142564' )

SELECT \* From Employees

--Ъпдейтване на таблица Employees с промяна на номера, чрез команда UPDATE.

UPDATE Employees

SET phoneNumber = '0896302565'

WHERE EmployeeID = 1

Select \*

From Employees

--Създаване на таблица Customers

create table Customers

(CustomerID int NOT NULL PRIMARY KEY IDENTITY,

CustomerName varchar(50) NOT NULL,

City varchar(50) NOT NULL,

postalCode varchar(50) NOT NULL,

address varchar(50) NOT NULL,

phoneNumber varchar(50) NOT NULL)

--Вкарване на данни в таблица Customers, чрез команда INSERT INTO.

INSERT INTO Customers(CustomerName, City, postalCode, address, phoneNumber)

VALUES ('Desislava', 'Velico Tarnovo', '5000', 'Balgaria-12', '0897142065' )

SELECT \* From Customers

--Ъпдейтване на таблица Customers с промяна на името на клиента, чрез команда UPDATE.

UPDATE Customers

SET CustomerName = 'Antoinette'

WHERE CustomerID = 1

Select \*

From Customers

--Създаване на таблица Genres

create table Genres

(GenreID int NOT NULL PRIMARY KEY IDENTITY,

GenreName varchar(50) NOT NULL)

--Вкарване на данни в таблица Genres, чрез команда INSERT INTO.

INSERT INTO Genres(GenreName)

VALUES ('Epicno Fentazi' )

SELECT \* From Genres

--Ъпдейтване на таблица Genres с промяна на името на жанра, чрез команда UPDATE.

UPDATE Genres

SET GenreName = 'Roman'

WHERE GenreID = 1

Select \*

From Genres

--Създаване на таблица Books

create table Books

(BookID int NOT NULL PRIMARY KEY IDENTITY,

Title varchar(50) NOT NULL,

Year int NOT NULL,

RentPrice decimal(5,2) NOT NULL,

author varchar(50) NOT NULL,

GenreID int NOT NULL REFERENCES Genres(GenreID))

--Вкарване на данни в таблица Books, чрез команда INSERT INTO.

INSERT INTO Books(Title, Year, RentPrice, author, GenreID)

VALUES ('Game of Thrones', '2012', 30, 'Djordj R. R. Martin',1 )

SELECT \* From Books

--Ъпдейтване на таблица Books с промяна на цената за наем, чрез команда UPDATE.

UPDATE Books

SET RentPrice = '9'

WHERE BookID = 1

Select \*

From Books

--Създаване на таблица Rents

create table Rents

(RentID int NOT NULL PRIMARY KEY IDENTITY,

RentDate date NOT NULL,

ReturnDate date NOT NULL,

CustomersID int NOT NULL REFERENCES Customers(CustomerID),

BookID int NOT NULL REFERENCES Books(BookID),

EmployeeID int NOT NULL REFERENCES Employees(EmployeeID))

--Вкарване на данни в таблица Rents, чрез команда INSERT INTO.

INSERT INTO Rents(RentDate, ReturnDate, CustomersID, BookID, EmployeeID)

VALUES ('2018-03-12', '2018-04-12', 1, 1, 1 )

SELECT \* From Rents

--Предназначението на заявката е изтриването на цялата информация на точно определено ID.

--Изтрива цялата информация за ID=2 от таблица Customers.

DELETE FROM Customers

WHERE CustomerID = 2;

--Предназначението на заявката е изтриването на точно определена иформация.

--Изтрива заглавието от таблица Books.

DELETE FROM Books

WHERE Title = 'Game of Thrones';

--Предназначението на заявката е изтриването на всичко над определено число .

--Изтрива всяка цена над 8.

DELETE Books

WHERE RentPrice > 8

--Предназначението на заявката е изтриването на таблица.

--Изтрива таблица Rents.

DELETE Rents

--Предназначението на заявката е изкарване на ид с най-малката стойност .

--Изкарва книгата с най-малка цена под наем.

SELECT \* FROM Books

WHERE RentPrice =

( SELECT MIN(RentPrice) FROM Books )

-- Предназначението на заявката е изкарване на средно аритметичното

--Изкарва средно аритметичното на цената под наем

SELECT AVG(RentPrice)

FROM [Books]

-- Предназначението на заявката е сумиране.

--Събира всички цени под наем.

SELECT SUM(RentPrice)

FROM Books

-- Предназначението на заявката е да изкарва дата влизаща между двете дати.

--Изкарва върнатите книги между двете дати.

Select \*

From Rents

WHERE RentID IN (1,5,12,90) AND ReturnDate BETWEEN '2018-03-12' AND '2018-05-12'

-- Предназначението на заявката е да изкарва дата влизаща между двете дати.

--Изкарва наеманите книги между двете дати.

Select \*

From Rents

WHERE RentID IN (1,5,12,90) AND RentDate BETWEEN '2018-02-12' AND '2018-06-12'

-- Предназначението на заявката е да покаже заглавието с най-малка и най-голяма цена за наемане.

--Изкарва заглавието с максималната и минималната цена за наемане.

SELECT Title, RentPrice, 'maxsimalna' AS Label

FROM Books

WHERE RentPrice =

(SELECT MAX(RentPrice) FROM Books)

UNION

SELECT Title, RentPrice,

'minimalna' AS Label

FROM Books

WHERE RentPrice =

(SELECT MIN(RentPrice)

FROM Books)

ORDER BY Title

-- Предназначението на заявката е да покаже броя.

--Изкарва броя на клиентите

SELECT COUNT(CustomerName) AS CustomerName

FROM Customers

-- Предназначението на заявката е да ограничава.

--Заявката ограничава номера до е до 10 цифри.

ALTER TABLE Employees

ADD CONSTRAINT valid\_phoneNumber

CHECK(phoneNumber LIKE'[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')

-- Предназначението на заявката е да ограничава.

--Заявката ограничава да е само с главни и малки букви.

ALTER TABLE Customers

ADD CONSTRAINT valid\_CustomerName

CHECK(CustomerName not LIKE '%[^a-zA-Z]%')

-- Предназначението на заявката е да изкарва всяко ид започващо с буквата.

--Заявката изкарва всички ид-та започващи с буквата R.

SELECT TOP 1 \* FROM Genres

WHERE GenreID NOT LIKE 'R%'

-- Предназначението на заявката е да групира.

--Заявката изкарва цената под наем в подреден ред от най-малката цена под наем до най-висока цена под наем.

SELECT COUNT(BookID), RentPrice

FROM Books

GROUP BY RentPrice

ORDER BY COUNT(BookID) DESC

-- Предназначението на тригера е да забрани триенето на ред от таблица

--Тригера забранява изтриване на 2 и 4 ред .След опит за изтриване, чрез заявка изписна :” Не е разрешено изтриване на редове в таблицата”.

CREATE TRIGGER No\_DeleteBooks

ON Books

INSTEAD OF DELETE

AS

IF @@ROWCOUNT = 0 RETURN

RAISERROR('Не е разрешено изтриване на

редове в таблицата.', 2,4 )