# Съдържание:

# Описание на предметната област и на заданието

Разработената база от данни в този документ показва как една библиотека лесно и достъпно може да следи своите налично данни. Базата от данни на библиотеката предоставя цялостна информация за книгите, които притежава тя, както и информация за нейните читатели. Базата още показва кой читател коя книга е взел за по лесна справка от страна на служителите. Също така всеки читател му се предоставя читателска карта за която данните се намират в базата от данни за справка. Всяка библиотека има различни секции които разделят различните видове книги и трябва да се следи за наличието на книгите в съответна секция. За тази цел базата предоставя точно тези данни за по лесни и достъпни справки.

Цялостната идея за разработването на тази база е улесняването на служителите на библиотеката с цел бърз достъп до данните на книгите и читателите и по дълъг период на запазване на информацията за тях.

# Дефиниране на схемата на релациите

- Създаване на релации

Създаване на база от данни Library

```
GO
CREATE DATABASE Library
```

Дефиниране на релациите:

Reader която съдържа данните за читателите в библиотеката:

```
USE Library

GO
CREATE TABLE Reader
(
	id INT PRIMARY KEY NOT NULL,
	egn NCHAR(10) UNIQUE NOT NULL,
	name VARCHAR(256),
	adres VARCHAR(256),
	phone NVARCHAR(13) UNIQUE NOT NULL,
	code_card INT FOREIGN KEY REFERENCES Reader_card(code),
	record_data DATETIME,
	count_get_book INT
);
```

Значение на атрибутите:

id е от тип integer и съдържа уникалните идентификатори на за всяка стойност (зададена е като първичен ключ и не може да съдържа празна стойност);

egn е от тип nchar от 10 знака и съдържа ЕГН на читателя (зададена е като уникален атрибут и не може да съдържа празна стойност);

name е от тип varchar от 256 знака и съдържа данни за името на читателите (може и да съдържа празна стойност);

adres е от тип varchar от 256 знака и съдържа данни за адреса на читателите (може и да съдържа празна стойност);

phone е от тип nvarchar от 13 знака и съдържа телефонните номера на читателя (зададена е като уникален атрибут и не може да съдържа празна стойност);

code_card е от тип integer и съдържа специфичния код на читателската карта на всеки читател (зададен е като вторичен ключ реферирящ към кода на картата и може да съдържа празна стойност);

record_data е от тип datetime и съдържа датата на вкарване на данните за читателя (може да съдържа празна стойност);

count_get_book е от тип integer и съдържа актуалния брой на книгите които дадения читател е взел.

Book която съдържа данни за книгите в библиотеката:

```
USE Library

GO
CREATE TABLE Book

(

	code INT PRIMARY KEY NOT NULL,

	author VARCHAR(256),

	title VARCHAR(256),

	year_create INT,

	id_section INT FOREIGN KEY REFERENCES Section(id),

	available CHAR(2),

	CHECK ((available = 'Y' OR available = 'N') AND (year_create >=
	1000 AND year_create <= 9999))

);
```

Значение на атрибутите:

code е от тип integer и съдържа данните за кода на книгите (зададен е като първичен ключ и не може да съдържа празна стойност);

author е от тип varchar от 256 знака и съдържа данните за автора на книгите (може да съдържа празна стойност);

title е от тип varchar от 256 знака и съдържа данни за заглавието на книгите (може да съдържа празна стойност);

year_create е от тип integer и съдържа данни за годината на създаване (написване) на книгата (може да съдържа празна стойност);

id_section е от тип integer и съдържа id-та на конкретна секция (зададена е като вторичен ключ реферирящ към id на секция и може да съдържа празна стойност);

available е от тип char от 1 знак и показва дали книгата е налична или не, за тази цел тя съдържа само Y/N данни (може да съдържа празна стойност);

Забележка: В таблица Book има проверка на ниво таблица която проверява дали available съдържа само Y/N стойности и дали годината на създаване (написване) на книгата има валидна стойност.

Section която показва наличието на книгите във всяка една секция в библиотеката (с цел справка):

```
USE Library

GO

CREATE TABLE Section

(

    id INT PRIMARY KEY NOT NULL,

    name VARCHAR(256),

    count_availabal_book INT DEFAULT 0,

    count_get_book INT DEFAULT 0

);
```

Значение на колоните:

id е от тип integer и съдържа уникалните идентификатори на за всяка стойност (зададена е като първичен ключ и не може да съдържа празна стойност);

name е от тип varchar от 256 знака и съдържа данни за името на секцията (може да съдържа празна стойност);

count_availabal_book е от тип integer и съдържа данни за броя на наличните книги в секцията (може да съдържа празна стойност);

count_get_book е от тип integer и съдържа данни за броя на заетите книги в секцията (може да съдържа празна стойност);

Reader_card която дава данни за картата на читателя:

```
USE Library

GO
CREATE TABLE Reader_card
(
	code INT PRIMARY KEY NOT NULL,
	date_create DATETIME DEFAULT GETDATE(),
	date_renewal DATETIME
);
```

Значение на колоните:

code е от тип integer и съдържа данни за кода на читателската карта (зададен е като първичен ключ и не може да съдържа празна стойност);

date_create е от тип datetime и съдържа данни за датата на създаване на читателската карта (може да съдържа празна стойност);

date_renewal е от тип datetime и съдържа данни за датата на последното подновяване на читателската карта (може да съдържа празна стойност);

Забележка: Ако не се въведат данни в date_create автоматично се генерира (по подразбиране) датата на настоящият ден.

Reader_book която дава данни за читателите които са взели книга от библиотеката и коя книга са взели:

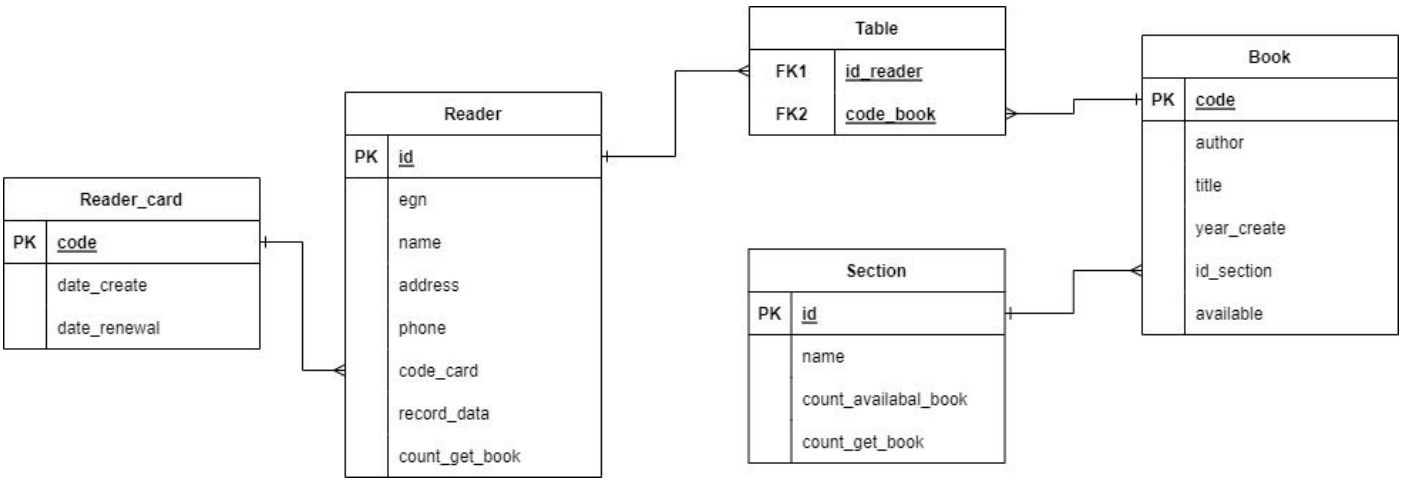
```
USE Library

GO

CREATE TABLE Reader_book

(

    id_reader INT FOREIGN KEY REFERENCES Reader(id),

    code_book INT FOREIGN KEY REFERENCES Book(code)

);
```

Значение на колоните:

id_reader е от тип integer и съдържа данни за id-то на читателя който е взел книгата (може и да съдържа празна стойност);

code_book е от тип integer и съдържа данни за кода на взетата книга (може и да съдържа празна стойност);

- Илюстриране на схемата на релациите




Схемата представя по какъв начин си комуникират отделните колони в всяка една релация.

# Добавяне на примерно съдържание

- ## Създаване на тригери

Преди въвеждането на данните в релациите ще се създадат тригерите с цел по ефикасно функциониране на базата.

```
USE Library

GO
CREATE TRIGGER trg_readerbook_insert
ON Reader_book
AFTER INSERT
AS
BEGIN
	UPDATE Reader SET count_get_book = count_get_book + 1
	WHERE id IN (SELECT i.id_reader FROM inserted i)

	UPDATE Book SET available = 'N'
	WHERE code IN (SELECT i.code_book FROM inserted i)
END;
```

Създава се тригер в reader_book който при добавяне на данни в тази релация да добавя 1 в count_get_book от Reader и да сменя available от Book на ‘N’.

```
USE Library

GO
CREATE TRIGGER trg_book_insert
ON Book
AFTER INSERT
AS
BEGIN
	UPDATE Section SET count_availabal_book = count_availabal_book + 1
	WHERE id IN (SELECT i.id_section FROM inserted i WHERE i.available = 'Y');

	UPDATE Section SET count_get_book = count_get_book + 1
	WHERE id IN (SELECT i.id_section FROM inserted i WHERE i.available = 'N');
END;
```

Създава се тригер в Book който при добавяне на данни в тази релация да добавя 1 към count_availabal_book от Section са когато available от Book е ‚Y‘ и добавя 1 към count_get_book от Section само ако available от Book е ‘N’.

```
USE Library

GO
CREATE TRIGGER trg_readerbook_delete
ON reader_book
AFTER DELETE
AS
BEGIN
	UPDATE Reader SET count_get_book = count_get_book - 1
	WHERE id IN (SELECT d.id_reader FROM deleted d);

	UPDATE Book SET available = 'Y'
	WHERE code IN (SELECT d.code_book FROM deleted d)
END;
```

Създава се тригер в reader_book който при изтриване на данни от тази релация намалява с 1 count_get_book от Reader и променя available от Book на ‚Y‘.

```
USE Library

GO
CREATE TRIGGER trg_book_delete
ON Book
AFTER DELETE
AS
BEGIN
	UPDATE Section SET count_availabal_book = count_availabal_book - 1
	WHERE id IN (SELECT i.id_section FROM deleted i WHERE i.available = 'Y');

	UPDATE Section SET count_get_book = count_get_book - 1
	WHERE id IN (SELECT i.id_section FROM deleted i WHERE i.available = 'N');
END;
```

Създава се тригер в Book който при изтриване на данни от тази релация премахва 1 от count_availabal_book в Section само ако available в Book е ‚Y‘ и премахва 1 от count_get_book само ако available от Book е ‚N‘.

```
USE Library

GO
CREATE TRIGGER trg_reader_update
ON Reader
AFTER UPDATE
AS
BEGIN
	UPDATE reader_book SET id_reader = (SELECT i.id FROM inserted i)
	WHERE id_reader IN (SELECT d.id FROM deleted d);
END;
```

Създава тригер в Reader който при обновяване на данните в тази релация променя данните в id_reader от reader_book.

```
USE Library

GO
CREATE TRIGGER trg_book_update
ON Book
AFTER UPDATE
AS
BEGIN
	UPDATE reader_book SET code_book = (SELECT i.code FROM inserted i)
	WHERE code_book IN (SELECT d.code FROM deleted d);

	UPDATE Section SET count_availabal_book = count_availabal_book + 1
	WHERE name IN (SELECT i.id_section FROM inserted i
				JOIN deleted d ON i.code = d.code
				WHERE i.available = 'Y' AND d.available = 'N');

	UPDATE Section SET count_get_book = count_get_book + 1
	WHERE name IN (SELECT i.id_section FROM inserted i
				JOIN deleted d ON i.code = d.code
				WHERE i.available = 'N' AND d.available = 'Y');

	UPDATE Section SET count_availabal_book = count_availabal_book - 1
	WHERE name IN (SELECT d.id_section FROM deleted d
				JOIN inserted i ON d.code = i.code
				WHERE i.available = 'N' AND d.available = 'Y');

	UPDATE Section SET count_get_book = count_get_book - 1
	WHERE name IN (SELECT d.id_section FROM deleted d
				JOIN inserted i ON i.code = d.code
				WHERE i.available = 'Y' AND d.available = 'N');
END;
```

Създава тригер в Book който при обновяване на данните в тази релация променя:

- code_book от Reader_book;
- увеличава count_availabal_book от Section, ако новата стойност в available от book е ‚Y‘ и старата е била ‚N‘;
- увеличава count_get_book, ако новата стойност в available от book е ‚N‘ и старата е била ‚Y‘;
- намалява count_availabal_book, ако новата стойност в available от book е ‚N‘ и старата е била ‚Y‘;
- намалява count_get_book, ако новата стойност в available от book е ‚Y‘ и старата е била ‚N‘;

- Добавяне на данни в релациите

Добавяне на данни в релацията Section:

```
USE Library

GO

INSERT INTO Section VALUES (1,'Art and Creativity',0,0);

INSERT INTO Section VALUES (2,'Business',0,0);

INSERT INTO Section VALUES (3,'Fantasy',0,0);

INSERT INTO Section VALUES (4,'Fitness',0,0);

INSERT INTO Section VALUES (5,'History',0,0);

INSERT INTO Section VALUES (6,'Novel',0,0);

INSERT INTO Section VALUES (7,'Philosophy',0,0);

INSERT INTO Section VALUES (8,'Science',0,0);
```

## Добавяне на данни в релацията Book:

```
USE Library

GO

INSERT INTO Book VALUES (1,'Robert Kiyosaki','Rich Dad Poor Dad',1997,2,'Y');

INSERT INTO Book VALUES (2,'George S. Clason','The Richest Man in Babylon ',1926,2,'N');

INSERT INTO Book VALUES (3,'J. R. R. Tolkien','The Lord of the Rings',1937,3,'N');

INSERT INTO Book VALUES (4,'Frederic Delavier','Strength Training Anatomy',1998,4,'Y');

INSERT INTO Book VALUES (5,'Leo Tolstoy','War and Peace',1865,5,'N');

INSERT INTO Book VALUES (6,'Carl Sagan','Cosmos',1980,8,'Y');

INSERT INTO Book VALUES (7,'Napoleon Hill','Think and Grow Rich',1937,2,'Y');

INSERT INTO Book VALUES (8,'F. Scott Fitzgerald','The Great Gatsby',1925,6,'Y');

INSERT INTO Book VALUES (9,'Daniel Defoe','Robinsons Crusoe',1719,6,'Y');
```

## Добавяне на данни в релацията Reader:

```
USE Library

GO

INSERT INTO Reader VALUES (1,'9910253454','Desislava Ivanova','Sofia','0882343543',10,'2008-11-11',0);

INSERT INTO Reader VALUES (2,'9504253410','Georgi Georgiev','Pleven','0881246532',11,'2011-5-11',0);

INSERT INTO Reader VALUES (3,'9510103420','Emilia Yancheva','Sofia','0881251956',12,'2015-9-6',0);

INSERT INTO Reader VALUES (4,'0210253620','Galin Petrov','Dobrich','0881263245',13,'2017-10-29',0);

INSERT INTO Reader VALUES (5,'9210303352','Yavor Yanakiev','Varna','0881276547',14,'2019-10-29',0);

INSERT INTO Reader VALUES (6,'9805203152','Yanica Vulkova','Ruse','0881281296',15,'2020-3-15',0);
```

Добавяне на данни в релацията Reader_card:

```
USE Library
GO
INSERT INTO Reader_card VALUES (10,'2008-11-11 00:00:00.000','2010-11-11 00:00:00.000');
INSERT INTO Reader_card VALUES (11,'2011-05-11 00:00:00.000','2014-05-11 00:00:00.000');
INSERT INTO Reader_card VALUES (12,'2015-09-06 00:00:00.000','2017-09-06 00:00:00.000');
INSERT INTO Reader_card VALUES (13,'2017-10-29 00:00:00.000','2019-10-29 00:00:00.000');
INSERT INTO Reader_card VALUES (14,'2019-10-29 00:00:00.000','2019-10-29 00:00:00.000');
INSERT INTO Reader_card VALUES (15,'2020-03-15 00:00:00.000','2020-03-15 00:00:00.000');
```

Добавяне на данни в релацията Reader_book:

```
USE Library
GO
INSERT INTO Reader_book VALUES (4,1);
INSERT INTO Reader_book VALUES (5,2);
INSERT INTO Reader_book VALUES (6,3);
INSERT INTO Reader_book VALUES (3,7);
INSERT INTO Reader_book VALUES (2,4);
INSERT INTO Reader_book VALUES (2,5);
```

Забележка: Данните в релациите са въвеждани в последователността в която са в документа с цел избягване на ненужни усложнения, също така връзките са създадени след създаването на тригерите и въвеждането на данните в релациите.

# Примерни заявки

- ## Прости заявки

Заявка 1:

```
USE Library

GO

SELECT b.title AS 'title_of_book', b.author AS 'author_of_book'

FROM Book b

WHERE b.year_create >= 1926

ORDER BY b.title ASC;
```

Резултат:

| | title_of_book | author_of_book |
|---|---|---|
| 1 | Cosmos | Carl Sagan |
| 2 | Rich Dad Poor Dad | Robert Kiyosaki |
| 3 | Strength Training Anatomy | Frederic Delavier |
| 4 | The Lord of the Rings | J. R. R. Tolkien |
| 5 | The Richest Man in Babylon | George S. Clason |
| 6 | Think and Grow Rich | Napoleon Hill |

Обосновка:

Заявката извежда заглавието на книгите и автора им които са написани след 1926 сортирани низходящо по заглавието на книгата.

Заявка 2:

```
USE Library

GO

SELECT r.name AS 'name_of_reader', r.address AS 'address_of_reader', r.phone AS 'phone_of_reader'

FROM Reader r
```

```
WHERE r.address = 'Sofia'

ORDER BY r.name DESC;
```

Резултат:

| | name_of_reader | address_of_reader | phone_of_reader |
|---|---|---|---|
| 1 | Emilia Yancheva | Sofia | 0881251956 |
| 2 | Desislava Ivanova | Sofia | 0882343543 |

Обосновка:

Заявката извежда имената, адресите, телефоните на тези читатели които имат адрес Sofia сортирани възходящо по името на читателя.

Заявка 3:

```
USE Library

GO

SELECT *

FROM Section s

WHERE s.name NOT LIKE 'Art and Creativity';
```

Резултат:

| | id | name | count_availabal_book | count_get_book |
|---|---|---|---|---|
| 1 | 2 | Business | 0 | 0 |
| 2 | 3 | Fantasy | 0 | 0 |
| 3 | 4 | Fitness | 0 | 0 |
| 4 | 5 | History | 0 | 0 |
| 5 | 6 | Novel | 0 | 0 |
| 6 | 7 | Philosophy | 0 | 0 |
| 7 | 8 | Science | 0 | 0 |

Обосновка:

Заявката извежда цялата информация за секциите без тази на Art and Creativity.

Заявка 4:

```
USE Library

GO

SELECT *

FROM Book b

WHERE b.available = 'Y';
```

Резултат:

| | code | author | title | year_create | id_section | available |
|---|---|---|---|---|---|---|
| 1 | 1 | Robert Kiyosaki | Rich Dad Poor Dad | 1997 | 2 | Y |
| 2 | 4 | Frederic Delavier | Strength Training Anatomy | 1998 | 4 | Y |
| 3 | 6 | Carl Sagan | Cosmos | 1980 | 8 | Y |
| 4 | 7 | Napoleon Hill | Think and Grow Rich | 1937 | 2 | Y |
| 5 | 8 | F. Scott Fitzgerald | The Great Gatsby | 1925 | 6 | Y |
| 6 | 9 | Daniel Defoe | Robinsons Crusoe | 1719 | 6 | Y |

Обосновка:

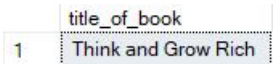
Заявката извежда цялата информация за книгите които са свободни.

- Заявки върху две и повече релации

Заявка 1:

```
USE Library

GO

SELECT b.title AS 'title_of_book'

FROM Book b

JOIN Reader_book rb ON b.code = rb.code_book

JOIN Reader r ON rb.id_reader = r.id

WHERE r.name = 'Emilia Yancheva';
```

Резултат:



| | title_of_book |
|---|---|
| 1 | Think and Grow Rich |

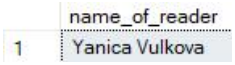
Обосновка:

Заявката извежда заглавието на книгите които Emilia Yancheva е взела.

Заявка 2:

```
USE Library

GO

SELECT r.name AS 'name_of_reader'

FROM Reader r

JOIN Reader_book rb ON r.id = rb.id_reader

JOIN Book b ON rb.code_book = b.code

JOIN Section s ON b.id_section = s.id

WHERE s.name = 'Fantasy';
```

Резултат:



| | name_of_reader |
|---|---|
| 1 | Yanica Vulkova |

Обосновка:

Заявката извежда имената на читателите които са взели книги само от секция Fantasy.

Заявка 3:

```
USE Library

GO

SELECT r.name AS 'name_of_reader', FORMAT(rc.date_create,N'dd.MM.yyyy') AS 'Date_of_create_profile'

FROM Reader r

JOIN Reader_card rc ON r.code_card = rc.code

WHERE rc.date_renewal < GETDATE();
```

Резултат:

| | name_of_reader | Date_of_create_profile |
|---|---|---|
| 1 | Desislava Ivanova | 11.11.2008 |
| 2 | Georgi Georgiev | 11.05.2011 |
| 3 | Emilia Yancheva | 06.09.2015 |
| 4 | Galin Petrov | 29.10.2017 |
| 5 | Yavor Yanakiev | 29.10.2019 |
| 6 | Yanica Vulkova | 15.03.2020 |

Обосновка:

Заявката извежда имената на читателите и дата на създаване на читателската им карта  който са подновени преди днешната дата.

Забележка:
FORMAT() извежда датата на създаване на профила на читателя в формат „[ден].[месец].[година]“.

Заявка 4:

```
USE Library

GO

SELECT r.name AS 'name_of_reader'

FROM Reader r

JOIN Reader_book rb ON rb.id_reader = r.id

JOIN Book b ON rb.code_book = b.code

WHERE b.author = 'Robert Kiyosaki' OR b.author = 'Napoleon Hill';
```

Резултат:

| | name_of_reader |
|---|---|
| 1 | Galin Petrov |
| 2 | Emilia Yancheva |

Обосновка:

Заявката извежда имената на читателите които са взели книги с автори Robert Kiyosaki или Napoleon Hill.

- Подзаявки

Заявка 1:

```
USE Library

GO

SELECT r.name AS 'name_of_reader', r.egn AS 'egn_of_reader'

FROM Reader r

WHERE r.code_card IN (SELECT rc.code FROM Reader_card rc WHERE YEAR(rc.date_create)
= FORMAT(rc.date_renewal, N'yyyy'))

ORDER BY name DESC;
```

Резултат:

| | name_of_reader | egn_of_reader |
|---|---|---|
| 1 | Yavor Yanakiev | 9210303352 |
| 2 | Yanica Vulkova | 9805203152 |

Обосновка:

Заявката извежда името и егенето на читателите на които годината на създаване съвпада с годината на подновяване на читателските им карти.

Забележка:

YEAR() връща единствено годината на подадената му дата.

Заявка 2:

```
USE Library

GO

SELECT r.name AS 'name_of_reader', r.address AS 'address_of_reader'

FROM Reader r

WHERE r.code_card IN (SELECT rc.code FROM Reader_card rc WHERE
FORMAT(rc.date_create, N'MM') = 05)

AND r.id IN (SELECT rb.id_reader FROM Reader_book rb GROUP BY
rb.id_reader HAVING COUNT(rb.id_reader) > 1);
```

Резултат:

| | name_of_reader | address_of_reader |
|---|---|---|
| 1 | Georgi Georgiev | Pleven |

Обосновка:

Заявката извежда имената и адресите на читателите чийто карти са направени през месец Май и броя на взетите от него книги да е повече от 1.

Заявка 3:

```
USE Library

GO

SELECT b.title AS 'title_of_book', b.author AS 'author_of_book'

FROM Book b

WHERE b.id_section IN (SELECT s.id FROM Section s GROUP BY s.id HAVING
(SELECT COUNT(b.code) FROM Book b WHERE b.available = 'Y') >= 1);
```

Резултат:

| | title_of_book | author_of_book |
|---|---|---|
| 1 | Rich Dad Poor Dad | Robert Kiyosaki |
| 2 | The Richest Man in Babylon | George S. Clason |
| 3 | The Lord of the Rings | J. R. R. Tolkien |
| 4 | Strength Training Anatomy | Frederic Delavier |
| 5 | War and Peace | Leo Tolstoy |
| 6 | Cosmos | Carl Sagan |
| 7 | Think and Grow Rich | Napoleon Hill |
| 8 | The Great Gatsby | F. Scott Fitzgerald |
| 9 | Robinsons Crusoe | Daniel Defoe |

Обосновка:

Заявката извежда заглавието и автора на книгите чиито секции имат поне една свободна книга.

Заявка 4:

```
USE Library

GO

SELECT r.name AS 'name_of_reader'

FROM Reader r

WHERE r.id IN (SELECT rb.id_reader FROM Reader_book rb WHERE
rb.code_book IN (SELECT b.code FROM Book b WHERE b.year_create < 1937));
```

Резултат:

| | name_of_reader |
|---|---|
| 1 | Georgi Georgiev |
| 2 | Yavor Yanakiev |

Обосновка:

Заявката извежда имената на читателите които са взели книга която е написана през или преди 1937.

## Заявка 5:

```
USE Library

GO

SELECT b.title AS 'title_of_book', b.author AS 'author_of_book', b.year_create AS 'year_of_create_of_book'

FROM Book b

WHERE b.code IN (SELECT rb.code_book FROM Reader_book rb WHERE rb.id_reader IN (SELECT r.id FROM
Reader r WHERE r.name = 'Georgi Georgiev')

AND b.id_section IN (SELECT s.id FROM Section s GROUP BY s.id HAVING (SELECT COUNT(b.code) FROM Book b
WHERE b.available = 'Y') >= 1

OR (SELECT COUNT(b.code) FROM Book b WHERE b.available = 'N') >= 1));
```

## Резултат:

| | title_of_book | author_of_book | year_of_create_of_book |
|---|---|---|---|
| 1 | Strength Training Anatomy | Frederic Delavier | 1998 |
| 2 | War and Peace | Leo Tolstoy | 1865 |

## Обосновка:

Заявката изважда заглавието, автора, годината на създаване (написване) на книгите, които са взети от Georgi Georgiev и секцията им има поне 1 взета или не взета книга.

- Съединения

Заявка 1:

```
USE Library

GO
CREATE INDEX idx_book_year
ON book(code,year_create);

GO

SELECT r.name AS 'name_of reader', r.address AS 'address_of_reader',
FORMAT(rc.date_create,'dd-MM-yyyy') AS 'date_of_create_reader_card'

FROM Reader r

JOIN Reader_card rc ON rc.code = r.code_card

JOIN Reader_book rb ON r.id = rb.id_reader

JOIN Book b ON b.code = rb.code_book

WHERE b.year_create = 1937;
```

Резултат:

| | name_of reader | address_of_reader | date_of_create_reader_card |
|---|---|---|---|
| 1 | Yanica Vulkova | Ruse | 15-03-2020 |
| 2 | Emilia Yancheva | Sofia | 06-09-2015 |

Обосновка:

Създаваме индекс в book с цел по бърз достъп до годината на създаване (написване) на книгата. Заявката извежда името, адреса и датата на създаване на читателската карта на читателите които са взели книга написана през 1937.

Заявка 2:

```
USE Library

GO

SELECT s.name AS 'name_of_section', b.title AS 'title_of_book'

FROM Section s

FULL OUTER JOIN Book b ON s.id = b.id_section;
```

Резултат:

| | name_of_section | title_of_book |
|---|---|---|
| 1 | Art and Creativity | NULL |
| 2 | Business | Rich Dad Poor Dad |
| 3 | Business | The Richest Man in Babylon |
| 4 | Business | Think and Grow Rich |
| 5 | Fantasy | The Lord of the Rings |
| 6 | Fitness | Strength Training Anatomy |
| 7 | History | War and Peace |
| 8 | Novel | The Great Gatsby |
| 9 | Novel | Robinsons Crusoe |
| 10 | Philosophy | NULL |
| 11 | Science | Cosmos |

Обосновка:

Заявката извежда всички секции и заглавията на всички книги към тях дори и секциите да не съдържат нито една книга книга.

Заявка 3:

```
USE Library

GO
CREATE INDEX idx_book_aval_ns
ON book(code,available,id_section);

GO
CREATE INDEX idx_readercard_dr
ON reader_card(code,date_renewal);

GO

SELECT rc.code AS 'code_of_reader_card'

FROM Reader_card rc

JOIN Reader r ON rc.code = r.code_card

JOIN Reader_book rb ON rb.id_reader = r.id

JOIN Book b ON b.code = rb.code_book

JOIN Section s ON b.id_section = s.id

WHERE b.available = 'N' AND s.name = 'History' AND YEAR(rc.date_renewal) >= 2010;
```

Резултат:

| | code_of_reader_card |
|---|---|
| 1 | 11 |

Обосновка:

Създават се 2 индекса първият в book е с цел по бърз достъп до наличните книги и техните секции, а вторият в reader_card е с цел по бърз достъп до датата на последно подновяване на читателска карта. Заявката извежда кодовете на читателските карти на тези читатели които са взели книга от секция History и картата е последно подновена през или след 2010 год.

Заявка 4:

```
USE Library

GO
CREATE INDEX idx_reader_cgb
ON reader(name,count_get_book);

GO
CREATE INDEX idx_book_aut
ON book(code,author);

GO

SELECT r.name AS 'name_of_reader', b.title AS 'book_title'

FROM Reader r

JOIN Reader_book rb ON rb.id_reader = r.id

JOIN book b ON b.code = rb.code_book

WHERE r.count_get_book <= 2 AND b.author = 'Frederic Delavier' OR b.author = 'Daniel Defoe';
```

Резултат:

| | name_of_reader | book_title |
|---|---|---|
| 1 | Georgi Georgiev | Strength Training Anatomy |

Обосновка:

Създаваме 2 индекса, първият в reader с цел по бърз достъп до броя на книгите които е взел всеки един читател, а вторият в book с цел по бърз достъп до авторите на всяка една книга книгите. Заявката извежда заглавието на книгите и имената на читателите, които имат не повече от 2 взети книги, които са написани от Frederic Delavier или Daniel Defoe.

Заявка 5:

```
USE Library

GO
CREATE INDEX idx_section_cd
ON section(name,count_get_book);

GO

SELECT s.name AS 'name_of_section', b.title AS 'title_of_book', r.name AS 'name_of_reder', r.egn AS 'reader_EGN', FORMAT(rc.date_create,'MMMM') AS 'month_of_create_of_profile'

FROM section s

JOIN Book b ON b.id_section = s.id

JOIN Reader_book rb ON rb.code_book = b.code

JOIN Reader r ON r.id = rb.id_reader

JOIN Reader_card rc ON rc.code = r.code_card

WHERE s.count_get_book >= 1 AND b.year_create > 1865 AND b.available = 'N';
```

Резултат:

| | name_of_section | title_of_book | name_of_reder | reader_EGN | month_of_create_of_profile |
|---|---|---|---|---|---|
| 1 | Business | The Richest Man in Babylon | Yavor Yanakiev | 9210303352 | October |
| 2 | Fantasy | The Lord of the Rings | Yanica Vulkova | 9805203152 | March |

Обосновка:

Създава се индекс в section с цел по бърз достъп до броя на книгите които са взети в момента за всяка една секция. Заявката извежда имената на секциите, които имат поне една взета книга и взетите книги от тази секция да са написани след 1865, извежда и заглавието на тези книги, имената на читателите, които са ги взели и тяхното ЕГН и месеца на създаване на читателската им карта изписана с думи.

Забележка:

FORMAT() позволява и изписването на дните и месеците с думи (зависи от зададения формат на датата).

- Групиране и аграгация

Преди заявките създаваме изгледите с цел по голяма пълнота:

Изглед 1

```
USE Library

GO
CREATE VIEW readerbook_book
AS
SELECT r.name, b.title
FROM Reade r
JOIN Reader_book rb  ON r.id = rb.id_reader
JOIN Book b ON b.code = rb.code_book;
```

Обосновка:

Създава изглед който показва името на читателите които са взели книга и заглавията на съответните книги.

Изглед 2:

```
USE Library

GO
CREATE VIEW reader_readercard_readerbook
AS
SELECT r.name, r.egn, r.address, rc.date_renewal
FROM Reader r
JOIN Reader_book rb ON r.id_reader = r.id
JOIN Reader_card rc ON rc.code = r.code_card;
```

Обосновка:

Създава изглед който включва имената на читателите, които са взели поне една книга, ЕГН-то им, адреса им и датата на последното подновяване на картата им.

Изглед 3:

```
USE Library

GO
CREATE VIEW reader_readercard
AS
SELECT *
FROM Reader r
JOIN Reader_card rc ON rc.code = r.code_card
WHERE MONTH(rc.date_create) % 2 <> 0 AND MONTH(rc.date_renewal) % 2<>
0;
```

Обосновка:

Създава изглед който е за данните на читателя, чиито месеци на създаване и подновяване на картите е нечетно число

Изглед 4:

```
USE Library

GO
CREATE VIEW book_section
AS
SELECT *
FROM Book b
JOIN Section s ON s.id= b.id_section
WHERE s.count_get_book >= 1 AND s.name LIKE '%e%';
```

Обосновка:

Създава изглед който е за данните на книгите чиито секции имат поне 1 взета книга и в името си съдържат 'e'.

Заявка 1:

```
USE Library

GO

SELECT b.title AS 'title_of_book', b.author AS 'author_of_book', b.year_create
AS 'year_of_writing'

FROM Book b

JOIN

(SELECT id_section FROM Book

GROUP BY id_section

HAVING COUNT(id_section) > 1) bs ON bs.id_section = b.id_section;
```

Резултат:

| | title_of_book | author_of_book | year_of_writing |
|---|---|---|---|
| 1 | Rich Dad Poor Dad | Robert Kiyosaki | 1997 |
| 2 | The Richest Man in Babylon | George S. Clason | 1926 |
| 3 | Think and Grow Rich | Napoleon Hill | 1937 |
| 4 | The Great Gatsby | F. Scott Fitzgerald | 1925 |
| 5 | Robinsons Crusoe | Daniel Defoe | 1719 |

Обосновка:

Заявката извежда информация за заглавието, автора, годината на написване на книгите чиито секции имат поне 1 взета книга.

Заявка 2:

```
USE Library

GO

SELECT r.name AS 'name_of_reader'

FROM Reader r

JOIN (SELECT id_reader FROM Reader_book
```

```
GROUP BY id_reader

HAVING COUNT(id_reader) > 1) rb ON rb.id_reader = r.id
```

Резултат:

| | name_of_reader |
|---|---|
| 1 | Georgi Georgiev |

Обосновка:

Заявката извежда имената на читателите които са взели повече от една книга.

Заявка 3:

```
USE Library

GO

SELECT r.name AS 'name_of_reader', r.egn AS 'egn', r.address AS 'address', r.phone AS 'phone'

FROM Reader r

JOIN

(SELECT address

FROM Reader

GROUP BY address

HAVING COUNT(1) > 1) r1 ON r1.address = r.address

JOIN Reader_card rc ON rc.code = r.code_card

WHERE RIGHT(r.phone,1) % 2 <> 0 AND MONTH(rc.date_create) % 2 <> 0 AND
MONTH(rc.date_renewal) % 2 <> 0;
```

Резултат:

| | name_of_reader | egn | address | phone |
|---|---|---|---|---|
| 1 | Desislava Ivanova | 9910253454 | Sofia | 0882343543 |

Обосновка:

Заявката извежда данните на читателите, чиито месеци на създаване и подновяване на читателската им картите е нечетно число с еднакви адреси и телефоните им завършват на нечетна цифра

Забележка:

RIGHT() връща определен брой елементи от променливата която му е подадена като този брой е точно дефиниран и броенето е от дясно на ляво.

Заявка 4:

```
USE Library

GO

SELECT DISTINCT r.name AS 'name_of_reader', r.egn AS 'egn', r.address AS 'reader_address',
rc.date_renewal AS 'date_of_create_profile', b.title AS 'title_of_book', b.author AS 'author_of_book',
b.year_create 'year_of_create_book', s.name AS 'book_section'

FROM Reader r

JOIN Reader_card rc ON rc.code = r.code_card

JOIN Reader_book rb ON rb.id_reader = r.id

JOIN Book b ON b.code = rb.code_book

JOIN Section s ON b.id_section = s.id

JOIN (SELECT id_section

FROM Book

GROUP BY id_section

HAVING COUNT(id_section) > 1) b1 ON b1.id_section = b.id_section

GROUP BY b.author, r.name, r.egn, r.address, rc.date_renewal, b.title, b.year_create,s.name

HAVING LEN(b.author) > (SELECT RIGHT(MIN(YEAR(date_renewal)),2) FROM Reader_card);
```

Резултат:

| | name_of_reader | egn | reader_address | date_of_create_profile | title_of_book | author_of_book | year_of_create_book | book_section |
|---|---|---|---|---|---|---|---|---|
| 1 | Yavor Yanakiev | 9210303352 | Varna | 2019-10-29 00:00:00.000 | The Richest Man in Babylon | George S. Clason | 1926 | Business |
| 2 | Emilia Yancheva | 9510103420 | Sofia | 2017-09-06 00:00:00.000 | Think and Grow Rich | Napoleon Hill | 1937 | Business |
| 3 | Galin Petrov | 0210253620 | Dobrich | 2019-10-29 00:00:00.000 | Rich Dad Poor Dad | Robert Kiyosaki | 1997 | Business |

Обосновка:

Заявката извежда имената на читателите, които са взели поне една книга, ЕГН-то им, адреса им и датата на последното подновяване на картата им и информация за книгата която са взели, като извежда само тези които дължината на името на автора е по-голяма от последните 2 цифри на най-малката година на създаване на профила на читателя, също така само информацията за книгите и читателите от секция с повече от 1 взета книга.

Забележка:

LEN() връща дължината на посочената променлива.

Заявка 5:

```
USE Library

GO

SELECT COUNT(s.name) AS 'count_of_section'

FROM Book b

JOIN Section s ON s.id= b.id_section

WHERE s.count_availabal_book >= 1 AND LEFT(b.year_create,2) = 19;
```

Резултат:

| | Count of name section |
|---|---|
| 1 | 2 |

Обосновка:

Заявката извежда броя на секциите които имат поне 1 налична книга и годината на написване на книга от тази секция да са от 20 век.

## Заявка 6:

```
USE Library

GO

SELECT r.name AS 'name_of_reader', r.address AS 'address_of_reader', r.egn AS 'egn', r.phone AS 'reader_phone', r.record_data AS 'date_of_create_acount',

rc.date_create AS 'date_of_create_reader_card', rc.date_renewal AS 'date_of_last_renewal'

FROM Reader r

JOIN Reader_card rc ON r.code_card = rc.code

JOIN

(SELECT r.record_data

FROM Reader r

JOIN Reader_card rc ON rc.code = r.code_card

WHERE YEAR(r.record_data) BETWEEN (SELECT MIN(YEAR(date_renewal)) FROM reader_card) AND (SELECT MAX(YEAR(date_create)) FROM Reader_card)

) r2 ON r2.record_data = r.record_data

JOIN(

SELECT address

FROM Reader

GROUP BY address

HAVING COUNT(1) > 1

) r1 ON r1.address = r.address;
```

## Резултат:

| | name_of_reader | address_of_reader | egn | reader_phone | date_of_create_acount | date_of_create_reader_card | date_of_last_renewal |
|---|---|---|---|---|---|---|---|
| 1 | Emilia Yancheva | Sofia | 9510103420 | 0881251956 | 2015-09-06 00:00:00.000 | 2015-09-06 00:00:00.000 | 2017-09-06 00:00:00.000 |

## Обосновка:

Заявката извежда информация за читателите и читателските им карти като извежда единствено тези на които годината на създаване на профила е между най-малката година на подновяване и най-голямата на създаване на читателската карта и читателите с еднакви адреси.

Забележка:

BETWEEN [val1] AND [val2] указва в какъв диапазон трябва да варира дадена стойност.

Заявка 7:

```
USE Library

GO
SELECT b.author AS 'author_of_book'
FROM Book b
JOIN
(SELECT year_create
FROM Book
GROUP BY year_create
HAVING COUNT(1)>1) b1 ON b.year_create = b1.year_create
WHERE b.available = 'N';
```

Резултат:

| | author_of_book |
|---|---|
| 1 | J. R. R. Tolkien |

Обосновка:

Заявката извежда имената на авторите на книгите написани през една и съща година които не са налични.

Заявка 8:

```
USE Library

GO

SELECT COUNT(r.name) AS 'name_of_reader'

FROM Reader r

JOIN Reader_book rb ON rb.id_reader = r.id

JOIN Book b ON b.code = rb.code_book
```

```
JOIN Section s ON b.id_section = s.id

WHERE r.count_get_book = 1 AND s.name = 'Business';
```

Резултат:

| | name_of_reader |
|---|---|
| 1 | 3 |

Обосновка:

Заявката извежда броя на читателите взели по 1 книга от секция Business.

Заявка 10:

```
USE Library

GO

SELECT r.name AS 'name_of_reader', r.address AS 'address_of_reader', r.egn AS 'egn',
r.phone AS 'phone', r.record_data AS 'date_of_create_profile', rc.date_create AS
'date_of_create_reader_card', rc.date_renewal AS 'date_RRC', b.title AS 'title', b.author AS
'author', b.year_create AS 'year_create_book', s.name AS 'name_of_section',
s.count_get_book AS 'section_get_book'

FROM Reader r

JOIN Reader_card rc ON rc.code = r.code_card

JOIN Reader_book rb ON rb.id_reader = r.id

JOIN Book b ON b.code = rb.code_book

JOIN Section s ON s.id = b.id_section

JOIN (SELECT id_section

FROM Book

GROUP BY id_section

HAVING COUNT(1) > 1

) b1 ON b1.id_section = b.id_section

WHERE b.year_create >= (SELECT AVG(year_create) FROM Book)

ORDER BY r.name DESC;
```

Резултат:

| | name_of_reader | address_of_reader | egn | phone | date_of_create_profile | date_of_create_reader_card | date_RRC | title | author | year_create_book | name_of_section | section_get_book |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Yavor Yanakiev | Varna | 9210303352 | 0881276547 | 2019-10-29 00:00:00.000 | 2019-10-29 00:00:00.000 | 2019-10-29 00:00:00.000 | The Richest Man in Babylon | George S. Clason | 1926 | Business | 0 |
| 2 | Galin Petrov | Dobrich | 0210253620 | 0881263245 | 2017-10-29 00:00:00.000 | 2017-10-29 00:00:00.000 | 2019-10-29 00:00:00.000 | Rich Dad Poor Dad | Robert Kiyosaki | 1997 | Business | 0 |
| 3 | Emilia Yancheva | Sofia | 9510103420 | 0881251956 | 2015-09-06 00:00:00.000 | 2015-09-06 00:00:00.000 | 2017-09-06 00:00:00.000 | Think and Grow Rich | Napoleon Hill | 1937 | Business | 0 |

Обосновка:

Заявката извежда информация за читателите и книгите които са взели на които секциите се повтарят, годините на написване на книгите да надвишават средната им годишна стойност и са сортирани по името на читателя възходящо.