

База данни за Библиотека

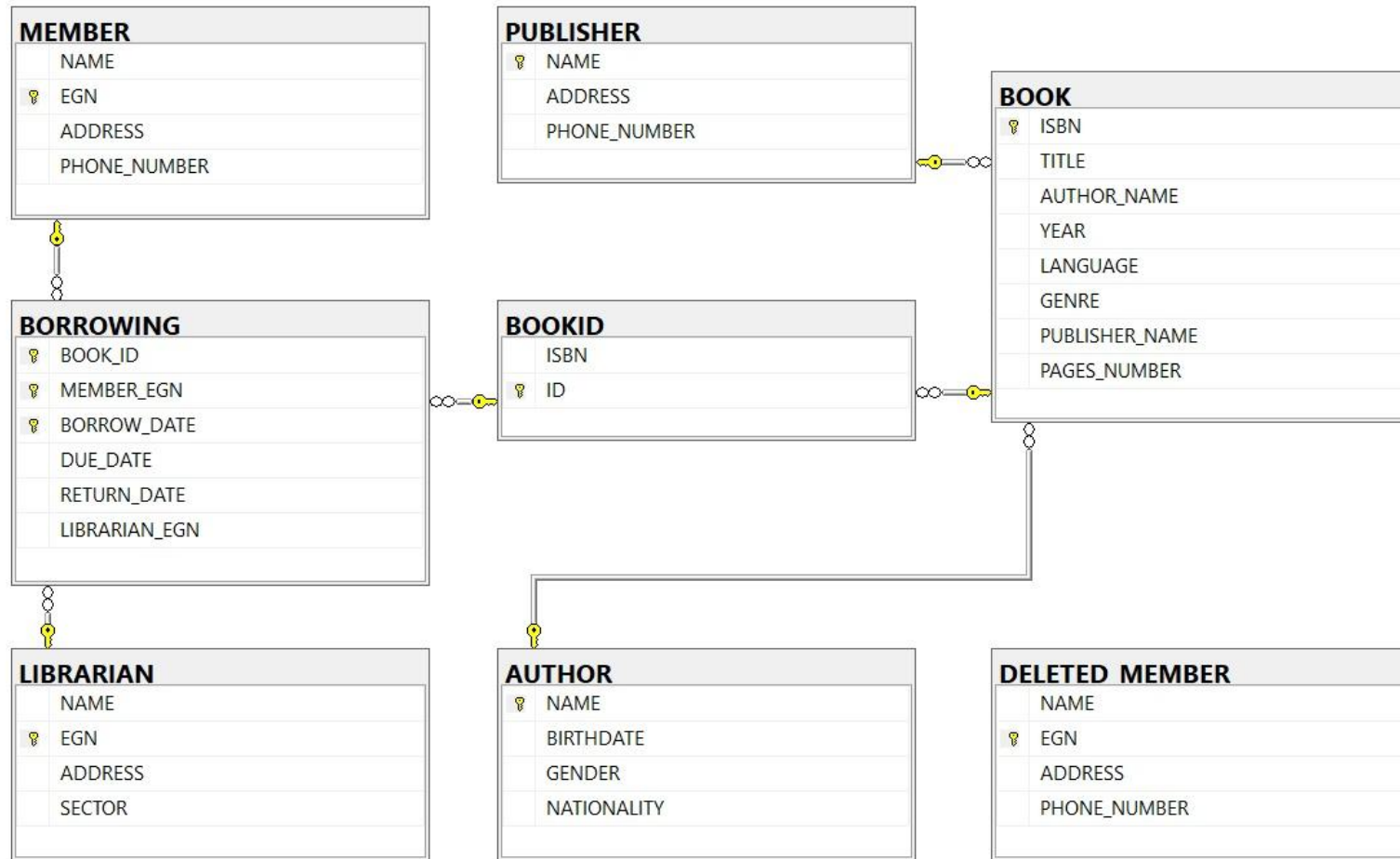
Валерия Кирилова 62305

Елиана Щерева 62249

Мария Ташкова 62294

Станислав Петров 62285

Database model



Изгледи

Създаваме виртуална таблица, която има атрибути всички атрибути от таблица BORROWING, както и ISBN от BOOKID

```
CREATE VIEW BORROWING_ISBN AS
SELECT BORROWING.*, BOOKID.ISBN
FROM BORROWING JOIN BOOKID ON BORROWING.BOOK_ID = BOOKID.ID;
```

Създаваме виртуална таблица, която има атрибути всички атрибути от таблица BORROWING без ID, както тези от BOOK

```
CREATE VIEW BORROWING_BOOK AS
SELECT BORROWING_ISBN.BOOK_ID, BORROWING_ISBN.MEMBER_EGN, BORROWING_ISBN.BORROW_DATE,
BORROWING_ISBN.DUE_DATE, BORROWING_ISBN.RETURN_DATE, BORROWING_ISBN.LIBRARIAN_EGN, BOOK.*
FROM BORROWING_ISBN JOIN BOOK ON BORROWING_ISBN.ISBN = BOOK.ISBN;
```

Създаваме виртуална таблица, която има атрибути ISBN от BOOK, както и колко пъти е четена книгата

```
CREATE VIEW ISBN_TIMES_READ AS
SELECT BOOK.ISBN, COUNT(*) AS TIMES_READ
FROM BORROWING_ISBN JOIN BOOK ON BORROWING_ISBN.ISBN = BOOK.ISBN
GROUP BY BOOK.ISBN;
```



Индекси

С цел оптимизация на търсенето по най-често използвани колони в заявките, дефинираме следните индекси:

- `CREATE INDEX BORROWING_BOOK_ID_IND ON BORROWING(BOOK_ID);`
- `CREATE INDEX BOOKID_ISBN_IND ON BOOKID(ISBN);`



Някои функционалности, които покрива нашата Библиотека

- Пазим архив с хронология на заемането на книги
- Можем да правим справки кой читател кои книги е заемал и кога, както и от кой библиотекар
- Следим за нарушители (читатели, които не са върнали взетите книги след изтичането на определен срок)
- Пазим информация за издателствата и авторите на книгите



Някои функционалности, които покрива нашата Библиотека

- Правим статистики за заемането на книги (напр. кои книги и автори са най-търсени, средно време за връщане на книги от читателите, в кой период са заемани най-много книги и т.н.)
- При надвишаване на определен брой нарушения читателят няма право да заема повече книги и да се регистрира отново
- Когато читателят търси определена книга, можем да му дадем информация към кого от библиотекарите да се обърне
- Следим за броя копия на книгите



Прости заявки

Заявка, която извежда името и пола на библиотекарите.

```
USE library
```

```
SELECT NAME, IIF(SUBSTRING(LIBRARIAN.EGN, 9, 1) % 2 = 0,  
'M', 'F') AS GENDER  
FROM LIBRARIAN;
```

	NAME	GENDER
1	Nikoleta Simeonova	F
2	Simona Lazarova	F
3	Ivan Asenov	M
4	Simeon Stefanov	M
5	Alexandra Andonova	F
6	Ivan Raev	M
7	Stanka Ivanova	F
8	Stancho Ivanov	M
9	Maria Ivanova	F
10	Preslava Raeva	F
11	Daniel Davidov	M
12	Silvia Ilieva	F
13	Yoanna Peeva	F
14	Stela Stoyanova	F

Прости заявки

Заявка, която извежда идентификационен номер на книгата и с колко дни е закъсняло връщането и.

USE library

```
SELECT BOOK_ID, DATEDIFF(day, BORROWING.DUE_DATE, GETDATE())  
AS DELAY_IN_DAYS  
FROM BORROWING  
WHERE RETURN_DATE IS NULL AND GETDATE() > BORROWING.DUE_DATE  
ORDER BY DELAY_IN_DAYS;
```

	BOOK_ID	DELAY_IN_DAYS
1	108	2
2	66	3
3	46	9
4	89	53
5	90	53
6	91	53
7	51	54
8	48	56
9	39	56
10	23	78
11	29	262
12	6	262
13	4	452
14	18	1103

Прости заявки

Заявка, която извежда името и възрастта на непълнолетните читатели

USE library

```
SELECT NAME, DATEDIFF(year, dbo.GET_BIRTHDATE(EGN), GETDATE())  
AS AGE  
FROM MEMBER  
WHERE DATEDIFF(year, dbo.GET_BIRTHDATE(EGN), GETDATE()) < 18  
ORDER BY AGE DESC;
```

	NAME	AGE
1	Petar Kojuharov	17
2	Mickey Moude	17
3	Georgi Dimitrov	16
4	Miglena Demirova	16
5	Sofia Basheva	14
6	Petar Sokratov	11
7	Radoslav Kamenov	10


Заявки върху две и повече релации

Заявка, която извежда заглавията на книгите и имената на библиотекарите, които отговарят за тях, сортирани по заглавие на книгата.

```
USE library
```

```
SELECT DISTINCT BOOK.TITLE, LIBRARIAN.NAME  
FROM LIBRARIAN, BOOK  
WHERE LIBRARIAN.SECTOR = BOOK.GENRE  
ORDER BY TITLE;
```

*Резултатът от заявката е прекалено дълъг за да се покаже визуално.



Заявки върху две и повече релации

Заявка, която извежда имената и адресите на читателите, за които има друг читател, който живее на същия адрес и ги сортира по адрес.

USE library

```
SELECT M1.NAME, M1.ADDRESS
FROM MEMBER M1, MEMBER M2
WHERE M1.EGN != M2.EGN AND M1.ADDRESS = M2.ADDRESS
ORDER BY M1.ADDRESS;
```

	NAME	ADDRESS
1	Herbert L Cook	Geo Milev 56 Plovdiv
2	Elis L Cook	Geo Milev 56 Plovdiv
3	Krum Kirilov	Kapitan Petko Voyvoda 31 Rudozem
4	Valeria Kirilova	Kapitan Petko Voyvoda 31 Rudozem
5	Martin Hristozov	Stefan Stambolov 18 Sofia
6	Robin M Jennings	Stefan Stambolov 18 Sofia
7	Ana Stoyanova	Vasil Levski 12 Varna
8	Ivana Stoyanova	Vasil Levski 12 Varna

Подзаявки

Заявка, която показва читателите, които са взели поне една книга в момента.

```
USE library
```

```
SELECT NAME  
FROM MEMBER  
WHERE EGN IN (SELECT DISTINCT MEMBER_EGN  
              FROM BORROWING  
              WHERE RETURN_DATE IS NULL);
```

	NAME
1	Petar Kojuharov
2	Herbert L Cook
3	Ivan Mihov
4	Nikolai Velinov
5	Ivona Petrova
6	Mitko Hristov
7	Inanina Radeva-Petkanova
8	Robin M Jennings
9	Denica Plamenova
10	Valentina Terzieva
11	James Hetfield
12	Preslav Liubenov
13	Mitko Hristov
14	Yoanna Mladenova
15	Ana Stoyanova
16	Ivana Stoyanova
17	Stela Stoyanova
18	Valeria Kirilova

Подзаявки

Заявка, която извежда всички книги, които не са вземани никога.

```
USE library
```

```
SELECT DISTINCT TITLE
FROM BOOK
WHERE ISBN IN (SELECT ISBN FROM BOOK
               EXCEPT
               SELECT ISBN FROM BORROWING_ISBN);
```

	TITLE
1	Epic of the forgotten
2	Rainwater
3	Smoke Screen

Подзаявки

Заявка, която извежда името на най-четения автор (чиито книги са заемани най-много пъти) и колко пъти са заемани книгите му.

	AUTHOR_NAME	TIMES_READ
1	Stephen King	33

USE library

```
SELECT *
FROM (SELECT AUTHOR_NAME, COUNT(*) AS TIMES_READ
      FROM BORROWING_BOOK
      GROUP BY AUTHOR_NAME) AS T
WHERE TIMES_READ = (SELECT TOP 1 COUNT(*) AS TIMES_READ
                   FROM BORROWING_BOOK
                   GROUP BY AUTHOR_NAME
                   ORDER BY TIMES_READ DESC);
```

Подзаявки

Заявка, която за всяка книга, която все още не е върната и срокът за връщане е изтекъл, извежда ID на книга, ЕГН, дни на закъснение, име на читател.

USE library

```
SELECT T.*, NAME
FROM ((SELECT BOOK_ID, MEMBER_EGN, DATEDIFF(DAY, DUE_DATE,
GETDATE()) AS DELAY
      FROM BORROWING
      WHERE RETURN_DATE IS NULL AND GETDATE() >
DUE_DATE) T JOIN MEMBER ON T.MEMBER_EGN = MEMBER.EGN)

ORDER BY MEMBER.EGN;
```

	BOOK_ID	MEMBER_EGN	DELAY	NAME
1	108	0347116501	3	Petar Kojuharov
2	39	4412166828	57	Herbert L Cook
3	4	7206176872	453	Ivona Petrova
4	51	8107046470	55	Denica Plamenova
5	66	8510183350	4	Valentina Terzieva
6	89	9001206501	54	James Hetfield
7	90	9001206501	54	James Hetfield
8	91	9001206501	54	James Hetfield
9	23	9508304489	79	Preslav Liubenov
10	48	9906200279	57	Ana Stoyanova
11	29	9907051992	263	Ivana Stoyanova
12	6	9907051992	263	Ivana Stoyanova
13	18	9907051992	1104	Ivana Stoyanova
14	46	9910160659	10	Stela Stoyanova

Съединения

Заявка, която извежда имената на читателите, които не са върнали книга и са просрочили датата ѝ за връщане.

```
USE library
```

```
SELECT DISTINCT NAME
```

```
FROM BORROWING JOIN MEMBER ON BORROWING.MEMBER_EGN = MEMBER.EGN
```

```
WHERE RETURN_DATE IS NULL AND GETDATE() > DUE_DATE;
```

	NAME
1	Ana Stoyanova
2	Denica Plamenova
3	Herbert L Cook
4	Ivana Stoyanova
5	Ivona Petrova
6	James Hetfield
7	Petar Kojuharov
8	Preslav Liubenov
9	Stela Stoyanova
10	Valentina Terzieva

Съединения

Заявка, която извежда имената на издателствата, от които не е вземана нито 1 книга, ако има такива.

PUBLISHER_NAME

```
USE library
```

```
SELECT PUBLISHER_NAME  
FROM BOOK  
JOIN BOOKID ON BOOK.ISBN=BOOKID.ISBN  
WHERE BOOKID.ID NOT IN (SELECT ID FROM BORROWING);
```



Групиране и агрегация

Заявка, която извежда имената и егн на първите 7 читатели, които са заемали най-много книги някога.

USE library

```
SELECT TOP 7 NAME, EGN, BORROWED_BOOKS_EVER
FROM MEMBER JOIN (SELECT MEMBER_EGN, COUNT(*) AS BORROWED_BOOKS_EVER
                  FROM BORROWING
                  GROUP BY BORROWING.MEMBER_EGN) T ON MEMBER.EGN = T.MEMBER_EGN
ORDER BY BORROWED_BOOKS_EVER DESC;
```

	NAME	EGN	BORROWED_BOOKS_EVER
1	Ivona Petrova	7206176872	10
2	Herbert L Cook	4412166828	8
3	Ivana Stoyanova	9907051992	7
4	Radostina Penevska	8209196555	5
5	Georgi Dimitrov	0442257183	5
6	Miglena Demirova	0452236613	5
7	Robin M Jennings	8007026260	5

Групиране и агрегация

Заявка, която показва най-скорошните година и месец, в които има най-много взети книги.

	YEAR_MONTH	BOOKS_BORROWED
1	2020-2	19

USE library

```
SELECT TOP 1 *, COUNT(*) AS BOOKS_BORROWED
FROM (SELECT CONCAT(YEAR(BORROW_DATE), '-',
MONTH(BORROW_DATE)) AS YEAR_MONTH FROM BORROWING) AS T
GROUP BY T.YEAR_MONTH
ORDER BY COUNT(*) DESC, T.YEAR_MONTH DESC;
```

Групиране и агрегация

Заявка, която извежда книгите, които са най-четени за всеки автор.

USE library

```
SELECT U.* FROM (SELECT AUTHOR_NAME, MAX(TIMES_READ) AS MAX_TIMES_READ
FROM (SELECT AUTHOR_NAME, TITLE, COUNT(*) AS TIMES_READ
FROM (SELECT BORROWING_BOOK.AUTHOR_NAME, BORROWING_BOOK.TITLE FROM BORROWING_BOOK) AS T
GROUP BY AUTHOR_NAME, TITLE) AS T
GROUP BY AUTHOR_NAME) AS T JOIN (SELECT AUTHOR_NAME, TITLE, COUNT(*) AS TIMES_READ
FROM (SELECT BORROWING_BOOK.AUTHOR_NAME, BORROWING_BOOK.TITLE FROM BORROWING_BOOK) AS T
GROUP BY AUTHOR_NAME, TITLE) AS U
ON T.AUTHOR_NAME = U.AUTHOR_NAME AND T.MAX_TIMES_READ = U.TIMES_READ

ORDER BY U.AUTHOR_NAME;
```

	AUTHOR_NAME	TITLE	TIMES_READ
1	Agatha Christie	And Then There Were None	2
2	Alan Alexander Milne	Winnie the Pooh	1
3	Alan Alexander Milne	The House at Pooh Corner	1
4	Antoine de Saint-Exupery	The Little Prince	1
5	Dan Brown	The Da Vinci Code	6
6	David Walliams	The Boy in the Dress	5
7	Ivan Vazov	Under the Yoke	5
8	James Clavell	Gai-Jin - Tome 1 & 2	6
9	James Clavell	Tai-Pan - Tome 1 & 2	6
10	Jeff Kinney	Diary of a Wimpy Kid - book 13: The Meltdown	1
11	Jeff Kinney	Diary of a Wimpy Kid - book 5: The Ugly Truth	1
12	Jojo Moyes	The Last Letter from Your Lover	2
13	Nicholas Sparks	Dear John	9
14	Sandra Brown	Where There's Smoke	2
15	Sandra Brown	Hidden Fires	2
16	Stephen King	The Eyes of the Dragon	9
17	Terry Pratchett	Eric	12
18	William Shakespeare	Hamlet	1
19	William Shakespeare	Romeo and Juliet	1
20	William Shakespeare	Sonnets	1

Групиране и агрегация

Заявка, която извежда издателството, чиято книга е вземана най-често и нейното заглавие.

USE library

```
SELECT PUBLISHER_NAME, TITLE, T.TIMES_READ FROM (  
  SELECT * FROM ISBN_TIMES_READ  
  WHERE TIMES_READ = (SELECT MAX(TIMES_READ)  
                      FROM ISBN_TIMES_READ)) AS T JOIN BOOK ON  
T.ISBN = BOOK.I
```

	PUBLISHER_NAME	TITLE	TIMES_READ
1	Ciela	Eric	12

Групиране и агрегация

Заявка, която извежда имената на читателите, които за просрочили датата за връщане поне два пъти и броя на тези нарушения.

USE library

```
SELECT NAME, COUNT(*) AS DELAYS
FROM BORROWING JOIN MEMBER ON BORROWING.MEMBER_EGN = MEMBER.EGN
WHERE RETURN_DATE > DUE_DATE OR (RETURN_DATE IS NULL AND
GETDATE() > DUE_DATE)
GROUP BY EGN, NAME
HAVING COUNT(*) > 1;
```

	NAME	DELAYS
1	Georgi Dimitrov	3
2	Radostina Penevska	3
3	James Hetfield	3
4	Georgi Dimitrov	2
5	Preslav Liubenov	2
6	Ivana Stoyanova	5

Тригери

Тригер, който вдига грешка, ако някой читател иска да вземе книга, която вече е взета от друг или ако вече е взел 7 книги

```
CREATE TRIGGER BORROWING_INSERT
ON BORROWING
INSTEAD OF INSERT
AS
IF (@@ROWCOUNT = 0)
RETURN;
IF EXISTS (SELECT *
FROM BORROWING JOIN inserted ON BORROWING.BOOK_ID = inserted.BOOK_ID
WHERE BORROWING.RETURN_DATE IS NULL AND inserted.RETURN_DATE > BORROWING.BORROW_DATE OR
BORROWING.BORROW_DATE < inserted.BORROW_DATE AND inserted.BORROW_DATE < BORROWING.RETURN_DATE OR
BORROWING.BORROW_DATE < inserted.RETURN_DATE AND inserted.RETURN_DATE < BORROWING.RETURN_DATE OR
inserted.BORROW_DATE <= BORROWING.BORROW_DATE AND inserted.RETURN_DATE >= BORROWING.RETURN_DATE
) OR EXISTS
(SELECT *
FROM inserted
WHERE inserted.BORROW_DATE > inserted.RETURN_DATE OR
inserted.BORROW_DATE > inserted.DUE_DATE
) OR EXISTS
(SELECT COUNT(*)
FROM BORROWING JOIN inserted ON BORROWING.MEMBER_EGN = inserted.MEMBER_EGN
WHERE BORROWING.RETURN_DATE IS NULL
GROUP BY BORROWING.MEMBER_EGN
HAVING COUNT(*) >= 7
)
BEGIN
RAISERROR ('Insert failed', 16, 1);
ROLLBACK TRANSACTION;
END
ELSE
INSERT INTO BORROWING (BOOK_ID, MEMBER_EGN, BORROW_DATE, DUE_DATE, RETURN_DATE, LIBRARIAN_EGN)
SELECT * FROM inserted;
```

Тригери

```
CREATE TRIGGER BORROWING_UPDATE
ON BORROWING
AFTER UPDATE
AS
```

```
    IF (@@ROWCOUNT = 0)
        RETURN;
    IF EXISTS (SELECT *
               FROM (SELECT * FROM BORROWING EXCEPT SELECT * FROM INSERTED) AS other JOIN inserted ON
other.BOOK_ID = inserted.BOOK_ID
               WHERE other.RETURN_DATE IS NULL AND inserted.RETURN_DATE > other.BORROW_DATE OR
other.BORROW_DATE < inserted.BORROW_DATE AND inserted.BORROW_DATE < other.RETURN_DATE OR
other.BORROW_DATE < inserted.RETURN_DATE AND inserted.RETURN_DATE <
other.RETURN_DATE OR
               inserted.BORROW_DATE <= other.BORROW_DATE AND inserted.RETURN_DATE >=
other.RETURN_DATE
               ) OR EXISTS
        (SELECT *
        FROM inserted
        WHERE inserted.BORROW_DATE > inserted.RETURN_DATE OR
        inserted.BORROW_DATE > inserted.DUE_DATE
        )
    BEGIN
        RAISERROR ('Update failed', 16, 1);
        ROLLBACK TRANSACTION;
    END;
```

Тригер, който вдига грешка, ако по погрешка се напише, че читател е върнал книга преди да я е взел

Тригери

```
CREATE TRIGGER MEMBER_DELETE
ON MEMBER
AFTER DELETE
AS
    IF (@@ROWCOUNT = 0)
        RETURN;
    INSERT INTO DELETED_MEMBER
    (NAME, EGN, ADDRESS, PHONE_NUMBER)
    SELECT *
    FROM deleted;
```

Тригер, който след изтриване на записи от таблицата MEMBER ги прехвърля в таблицата DELETED_MEMBER, за да не губим данните им завинаги



Тригери

```
CREATE TRIGGER MEMBER_INS_UPD
ON MEMBER
AFTER INSERT, UPDATE
AS
    IF (@@ROWCOUNT = 0)
        RETURN;
    IF EXISTS (SELECT * FROM DELETED_MEMBER
JOIN inserted ON DELETED_MEMBER.EGN =
inserted.EGN)
        BEGIN
            RAISERROR ('Insert/Update failed -
banned member', 16, 1);
            ROLLBACK TRANSACTION;
        END;
```

Тригер, който при INSERT или UPDATE на таблицата MEMBER, проверява дали данните му се съдържат в таблицата DELETED_MEMBER, за да не позволява повече да бъде добавен в MEMBER



Препоръки за възможни подобрения

- разширяване дейността на системата
- добавяне на абонаментна такса на читателите
- следене за работното време на библиотекарите
- автоматизация на процеса за установяване на нарушителите и прекратяването на правата им като членове

