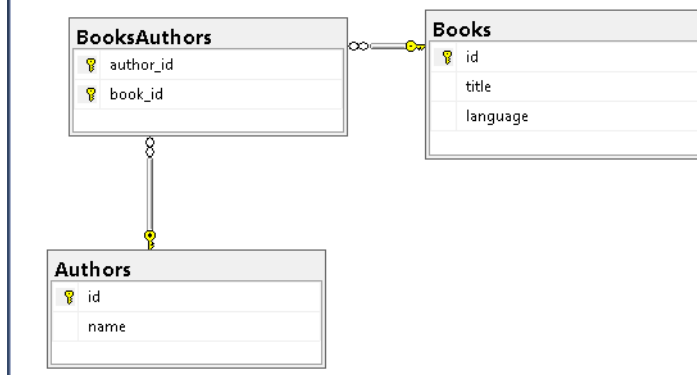


Lab 3

The first 3 problems have to be solved in SQL SERVER. The last problem will have to be solved in C#.

- Creați o procedură stocată ce inserează date pentru entități ce se află într-o relație m-n. Dacă o operație de inserare eșuează, trebuie făcut *roll-back* pe întreaga procedură stocată. (nota: 3)

We consider the database



Create functions for validation: for example - check the language to have some values (for table Books)

```
CREATE FUNCTION uf_ValidateLanguage (@language varchar(100)) RETURNS INT AS
BEGIN
DECLARE @return INT
SET @return = 0
IF (@language IN ('English', 'Romanian', 'French'))
SET @return = 1
RETURN @return
END
```

Create the stored procedure with the following restrictions:

- Do not take the Id's as parameters (here id from Authors, id from Books, author_id and book_id from BooksAuthors)
- Take the parameters all the rest of the fields from the tables (here title, language, name)
- Create validation functions for the parameters (all you consider necessary), like:
 - a field apart to a domain of values (language IN ('English','Romanian','French'))
 - the fields of varchar type to be not null, start with a upper type, ..
 - the fields of int to be positive, ...
 - validation functions for telephone numbers, e-mail, ...
 - or, whatever do you need
- first we insert values in the tables Authors and Books (the order is not important) and then in BooksAuthors (the intermediate table), by taking the id from both of the tables. We can take the id from one of the tables in a variable or if the field is identity like the maximum value of that field.

Next, we give an example for a stored procedure for table Books.

```
CREATE PROCEDURE AddBookAuthor @title varchar(50), @language varchar(50) AS
BEGIN
BEGIN TRAN
BEGIN TRY
IF (dbo.uf_ValidateLanguage(@language)<>1)
```

```

BEGIN
    RAISERROR('Language must be Romanian, English or French',14,1)
END
INSERT INTO Books (title, language) VALUES (@title, @language)
COMMIT TRAN
SELECT 'Transaction committed'
END TRY

BEGIN CATCH
ROLLBACK TRAN
SELECT 'Transaction rolledback'
END CATCH
END

```

You MUST prepare 2 scenarios for the verification of this function: one with commit and one with rollback. The rollback can be obtain from the validation conditions given by the validation functions. You MUST return the history of the operations executed. You can use Select/PRINT messages or use Select * from table_name or any other solution that you consider.

Execution:

- with success (commit)

```

Select * from Books
EXEC AddBookAuthor 'Harry Potter and The Chamber of Secrets', 'English'
Select * from Books

```

	id	title	language
1	4	Panda	English
2	7	Codul lui Davinci	English

	(No column name)
1	Transaction committed

	id	title	language
1	4	Panda	English
2	7	Codul lui Davinci	English
3	9	Harry Potter an...	English

- without success (rollback)

```

Select * from Books
EXEC AddBookAuthor 'Inferno', 'Spanish'
Select * from Books

```

	id	title	language
1	4	Panda	English
2	7	Codul lui Davinci	English
3	9	Harry Potter and The Chamber of Secrets	English

	(No column name)
1	Transaction rolledback

	ErrorNumber	ErrorMessage
1	50000	Language must be Romanian, English or French

	id	title	language
1	4	Panda	English
2	7	Codul lui Davinci	English
3	9	Harry Potter an...	English

- Creați o procedură stocată ce inserează date pentru entități ce se află într-o relație m-n. Dacă o operație de inserare eșuează va trebui să se păstreze cât mai mult posibil din ceea ce s-a modificat până în acel moment. De exemplu, dacă se încearcă inserarea unei cărți și a autorilor acesteia, iar autorii au fost inserați cu succes însă apare o problemă la inserarea cărții, atunci să se facă *roll-back* la inserarea de carte însă autorii acesteia să rămână în baza de date. (nota: 5)

Here, the transaction will be split into 3 transactions in the same stored procedure:

- First for the table Authors - with the validation also
- Second for the table Books – with the validation also
- Third for the table BooksAuthors – with the id's taken from both of the previous tables

The idea is that one can insert separately in each of the table. If we can add in Books, we add, and in Authors we cannot add, but this won't affect the add from Books. Each table is treat separately and do not affect the add of the others tables.

The execution has to be done for a success case and also for an un-success case.

- Creați 4 scenarii ce reproduc următoarele situații generate de execuția concurentă: *dirty reads*, *non-repeatable reads*, *phantom reads* și un *deadlock*. Puteți implementa aceste scenarii atât ca proceduri stocate cât și ca interogări de sine stătătoare. De asemenea, pentru fiecare dintre scenariile create, găsiți soluții de rezolvare/evitare a acestor situații. (nota: 8)

You need to consider a table in which you will analyze the concurrency execution. Here I choose Books. You must prepare scenarios for each case: Transaction 1 with Transaction 2 and Transaction 1 with Transaction 2 'solved'. You have to create and save each of the transactions used. You can use one file for Transaction 1 and for Transaction 2 one file with both of the cases (unsolved and solved- also commented), or 2 files, saved suggestive. Or, you can organize the structure as you prefer, but to be clear. Also, prepare examples for each of the cases.

Try to run the transactions in the same time (or close). Start Transaction 1 first, introduce a delay there, so that Transaction 2 can be executed in that time. Immediately that Transaction 1 was started, start also Transaction 2. (If you run the transactions converse, the result will also be converse).

In table Books we have

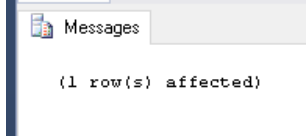
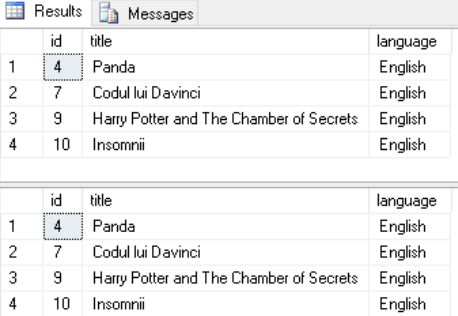
	id	title	language
1	4	Panda	English
2	7	Codul lui Davinci	English
3	9	Harry Potter and The Chamber of Secrets	English
4	10	Insomnii	English

For what follows: T1=Transaction 1 starts first. T2=Transaction start immediately after T1.

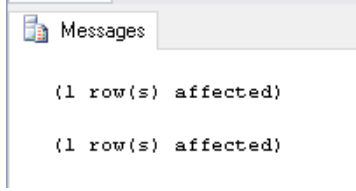
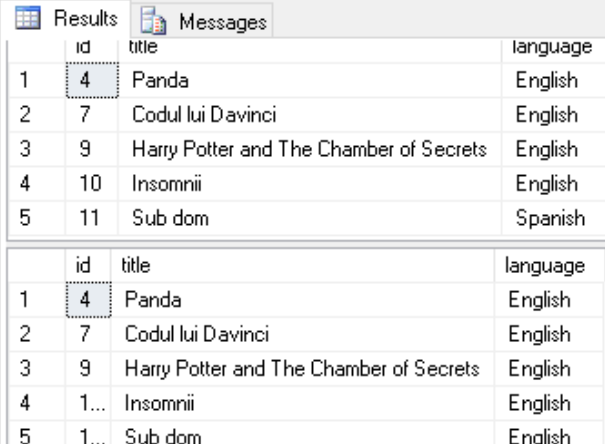
1. **DIRTY READS** – T1: 1 update + delay + rollback, T2: select + delay + select -> we see the update in the first select (T1 – finish first), even if it is rollback then
Isolation level: Read Uncommitted / Read Committed (solution)

<pre>--Dirty Reads Part 1 BEGIN TRANSACTION UPDATE Books SET language='Romanian' WHERE id = 7 WAITFOR DELAY '00:00:10' ROLLBACK TRANSACTION</pre>	<pre>--Dirty Reads Part 2 SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED BEGIN TRAN SELECT * FROM Books WAITFOR DELAY '00:00:15' SELECT * FROM Books COMMIT TRAN</pre>																																								
<div>Messages</div> <div>{1 row(s) affected}</div>	<div>Results Messages</div> <table><tr><th></th><th>id</th><th>title</th><th>language</th></tr><tr><td>1</td><td>4</td><td>Panda</td><td>English</td></tr><tr><td>2</td><td>7</td><td>Codul lui Davinci</td><td>Romanian</td></tr><tr><td>3</td><td>9</td><td>Harry Potter and The Chamber of Secrets</td><td>English</td></tr><tr><td>4</td><td>10</td><td>Insomnii</td><td>English</td></tr></table> <table><tr><th></th><th>id</th><th>title</th><th>language</th></tr><tr><td>1</td><td>4</td><td>Panda</td><td>English</td></tr><tr><td>2</td><td>7</td><td>Codul lui Davinci</td><td>English</td></tr><tr><td>3</td><td>9</td><td>Harry Potter and The Chamber of Secrets</td><td>English</td></tr><tr><td>4</td><td>10</td><td>Insomnii</td><td>English</td></tr></table>		id	title	language	1	4	Panda	English	2	7	Codul lui Davinci	Romanian	3	9	Harry Potter and The Chamber of Secrets	English	4	10	Insomnii	English		id	title	language	1	4	Panda	English	2	7	Codul lui Davinci	English	3	9	Harry Potter and The Chamber of Secrets	English	4	10	Insomnii	English
	id	title	language																																						
1	4	Panda	English																																						
2	7	Codul lui Davinci	Romanian																																						
3	9	Harry Potter and The Chamber of Secrets	English																																						
4	10	Insomnii	English																																						
	id	title	language																																						
1	4	Panda	English																																						
2	7	Codul lui Davinci	English																																						
3	9	Harry Potter and The Chamber of Secrets	English																																						
4	10	Insomnii	English																																						

Solution: T1: 1 update + delay + rollback, T2: select + delay + select -> we don't see the update (that is also rollback) – T1 finish first

<pre>--Dirty Reads Part 1 BEGIN TRANSACTION UPDATE Books SET language='Romanian' WHERE id = 7 WAITFOR DELAY '00:00:10' ROLLBACK TRANSACTION</pre>	<pre>--Solution: SET TRANSACTION ISOLATION LEVEL TO READ COMMITTED SET TRANSACTION ISOLATION LEVEL READ COMMITTED BEGIN TRAN SELECT * FROM Books WAITFOR DELAY '00:00:15' SELECT * FROM Books COMMIT TRAN</pre>
	

2. **NON-REPEATABLE READS** – T1: (insert +)delay + update + commit, T2: select + delay + select -> see the insert in first select of T2 + update in the second select of T2, T1 finish first
Isolation level: Read Committed / Repeatable Read (solution)

<pre>INSERT INTO Books(title, language) VALUES ('Sub dom', 'Spanish') BEGIN TRAN WAITFOR DELAY '00:00:05' UPDATE Books SET language='English' WHERE title = 'Sub dom' COMMIT TRAN</pre>	<pre>SET TRANSACTION ISOLATION LEVEL READ COMMITTED BEGIN TRAN SELECT * FROM Books WAITFOR DELAY '00:00:05' SELECT * FROM Books COMMIT TRAN</pre>
	

Solution: T1: insert + delay + update + commit, T2: select + delay + select -> see only the final result in both of the select of T2, T1 finish first

<pre>INSERT INTO Books(title, language) VALUES ('Sub dom','Spanish') BEGIN TRAN WAITFOR DELAY '00:00:05' UPDATE Books SET language='English' WHERE title = 'Sub dom' COMMIT TRAN</pre>	<pre>SET TRANSACTION ISOLATION LEVEL REPEATABLE READ BEGIN TRAN SELECT * FROM Books WAITFOR DELAY '00:00:05' SELECT * FROM Books COMMIT TRAN</pre>																																																								
<div>Messages</div> <div><pre>(1 row(s) affected) (2 row(s) affected)</pre></div>	<div>ResultsMessages</div> <table><tr><th></th><th>id</th><th>title</th><th>language</th></tr><tr><td>1</td><td>4</td><td>Panda</td><td>English</td></tr><tr><td>2</td><td>7</td><td>Codul lui Davinci</td><td>English</td></tr><tr><td>3</td><td>9</td><td>Harry Potter and The Chamber of Secrets</td><td>English</td></tr><tr><td>4</td><td>10</td><td>Insomnii</td><td>English</td></tr><tr><td>5</td><td>11</td><td>Sub dom</td><td>English</td></tr><tr><td>6</td><td>13</td><td>Sub dom</td><td>Spanish</td></tr></table> <table><tr><th></th><th>id</th><th>title</th><th>language</th></tr><tr><td>1</td><td>4</td><td>Panda</td><td>English</td></tr><tr><td>2</td><td>7</td><td>Codul lui Davinci</td><td>English</td></tr><tr><td>3</td><td>9</td><td>Harry Potter and The Chamber of Secrets</td><td>English</td></tr><tr><td>4</td><td>10</td><td>Insomnii</td><td>English</td></tr><tr><td>5</td><td>11</td><td>Sub dom</td><td>English</td></tr><tr><td>6</td><td>13</td><td>Sub dom</td><td>Spanish</td></tr></table>		id	title	language	1	4	Panda	English	2	7	Codul lui Davinci	English	3	9	Harry Potter and The Chamber of Secrets	English	4	10	Insomnii	English	5	11	Sub dom	English	6	13	Sub dom	Spanish		id	title	language	1	4	Panda	English	2	7	Codul lui Davinci	English	3	9	Harry Potter and The Chamber of Secrets	English	4	10	Insomnii	English	5	11	Sub dom	English	6	13	Sub dom	Spanish
	id	title	language																																																						
1	4	Panda	English																																																						
2	7	Codul lui Davinci	English																																																						
3	9	Harry Potter and The Chamber of Secrets	English																																																						
4	10	Insomnii	English																																																						
5	11	Sub dom	English																																																						
6	13	Sub dom	Spanish																																																						
	id	title	language																																																						
1	4	Panda	English																																																						
2	7	Codul lui Davinci	English																																																						
3	9	Harry Potter and The Chamber of Secrets	English																																																						
4	10	Insomnii	English																																																						
5	11	Sub dom	English																																																						
6	13	Sub dom	Spanish																																																						

3. **PHANTOM READS** – T1: delay + insert (/delete) + commit, T2: select + delay + select -> see the inserted value only at the second select from T2, T1 finish first
Isolation level: Repeatable Read / Serializable (solution)

<pre>--Phantom Reads Part 1 --DELETE FROM Books BEGIN TRAN WAITFOR DELAY '00:00:04' INSERT INTO Books(title,language) VALUES ('Morometii', 'Romanian') COMMIT TRAN</pre>	<pre>--Phantom Reads Part 2 SET TRANSACTION ISOLATION LEVEL REPEATABLE READ BEGIN TRAN SELECT * FROM Books WAITFOR DELAY '00:00:05' SELECT * FROM Books COMMIT TRAN</pre>																																																												
<div>Messages</div> <div>(1 row(s) affected)</div>	<div>Results Messages</div> <table><tr><th></th><th>id</th><th>title</th><th>language</th></tr><tr><td>1</td><td>4</td><td>Panda</td><td>English</td></tr><tr><td>2</td><td>7</td><td>Codul lui Davinci</td><td>English</td></tr><tr><td>3</td><td>9</td><td>Harry Potter and The Chamber of Secrets</td><td>English</td></tr><tr><td>4</td><td>10</td><td>Insomnii</td><td>English</td></tr><tr><td>5</td><td>11</td><td>Sub dom</td><td>English</td></tr><tr><td>6</td><td>13</td><td>Sub dom</td><td>English</td></tr></table> <div></div> <table><tr><th></th><th>id</th><th>title</th><th>language</th></tr><tr><td>1</td><td>4</td><td>Panda</td><td>English</td></tr><tr><td>2</td><td>7</td><td>Codul lui Davinci</td><td>English</td></tr><tr><td>3</td><td>9</td><td>Harry Potter and The Chamber of Secrets</td><td>English</td></tr><tr><td>4</td><td>1...</td><td>Insomnii</td><td>English</td></tr><tr><td>5</td><td>1...</td><td>Sub dom</td><td>English</td></tr><tr><td>6</td><td>1...</td><td>Sub dom</td><td>English</td></tr><tr><td>7</td><td>1...</td><td>Morometii</td><td>Romani...</td></tr></table>		id	title	language	1	4	Panda	English	2	7	Codul lui Davinci	English	3	9	Harry Potter and The Chamber of Secrets	English	4	10	Insomnii	English	5	11	Sub dom	English	6	13	Sub dom	English		id	title	language	1	4	Panda	English	2	7	Codul lui Davinci	English	3	9	Harry Potter and The Chamber of Secrets	English	4	1...	Insomnii	English	5	1...	Sub dom	English	6	1...	Sub dom	English	7	1...	Morometii	Romani...
	id	title	language																																																										
1	4	Panda	English																																																										
2	7	Codul lui Davinci	English																																																										
3	9	Harry Potter and The Chamber of Secrets	English																																																										
4	10	Insomnii	English																																																										
5	11	Sub dom	English																																																										
6	13	Sub dom	English																																																										
	id	title	language																																																										
1	4	Panda	English																																																										
2	7	Codul lui Davinci	English																																																										
3	9	Harry Potter and The Chamber of Secrets	English																																																										
4	1...	Insomnii	English																																																										
5	1...	Sub dom	English																																																										
6	1...	Sub dom	English																																																										
7	1...	Morometii	Romani...																																																										

Solution: T1: delay + insert + commit, T2: select + delay + select -> see the inserted value in both of the select from T2, T1 finish first

<pre> --Phantom Reads Part 1 --DELETE FROM Books BEGIN TRAN WAITFOR DELAY '00:00:04' INSERT INTO Books(title,language) VALUES ('Morometii', 'Romanian') </pre>	<pre> --Solution: Set transaction isolation level to SERIALIZABLE SET TRANSACTION ISOLATION LEVEL SERIALIZABLE BEGIN TRAN SELECT * FROM Books WAITFOR DELAY '00:00:05' </pre>
--	---

COMMIT TRAN

SELECT * FROM Books
COMMIT TRAN

Messages

(1 row(s) affected)

Results

Messages

	id	title	language
1	4	Panda	English
2	7	Codul lui Davinci	English
3	9	Harry Potter and The Chamber of Secrets	English
4	10	Insomnii	English
5	11	Sub dom	English
6	13	Sub dom	English
7	19	Morometii	Romanian

	id	title	language
1	4	Panda	English
2	7	Codul lui Davinci	English
3	9	Harry Potter an...	English
4	1...	Insomnii	English
5	1...	Sub dom	English
6	1...	Sub dom	English
7	1...	Morometii	Romani...

4. **DEADLOCK** – T1: update on table A + delay + update on table B, T2: update on table B + delay + update on table A

We update on table A (from T1 – that exclusively lock on table A), update on table B (from T2 – that exclusively lock on table B), try to update from T1 table B (but this transaction will be blocked because T2 has already been locked on table B), try to update from T2 table A (but this transaction will be blocked because T1 has already been locked on table A). So, both of the transactions are blocked. After some seconds T2 will be chosen as a deadlock victim and terminates with an error. After that, T1 will finish also. In table A and table B will be the values from T1.

Here we consider 2 tables: Books, Authors.

Books

	id	title	language
1	4	Panda	English
2	7	Codul lui Davinci	English
3	9	Harry Potter and The Chamber of Secrets	English
4	10	Insomnii	English
5	11	Sub dom	English
6	13	Sub dom	English
7	19	Morometii	Romanian
8	20	Morometii	Romanian

Authors

	id	name
1	4	J.K. Rowling
2	5	Irina Binder
3	6	2

```
-- transaction 1
begin tran
update Books set title='La cirese transaction 1' where id=20
-- this transaction has exclusively lock on table Books
waitfor delay '00:00:10'

update Authors set name='Petre Ispirescu transaction 1' where id=6
-- this transaction will be blocked because transaction 2 has already
blocked our lock on table Authors
-- so, transaction 1 is blocked on an exclusively block on table
Authors
commit tran
-- transaction 2
```

Messages

(1 row(s) affected)

(1 row(s) affected)

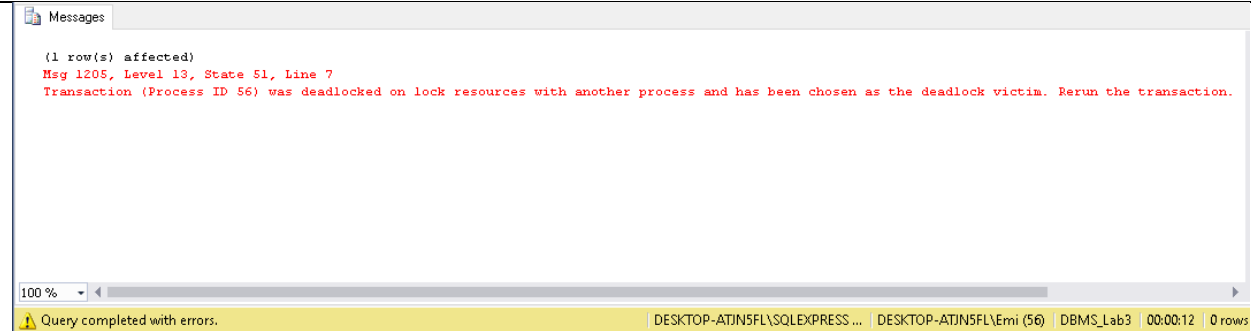
```

begin tran
update Authors set name='Petre Ispirescu transaction 2' where id=6
-- this transaction has exclusively lock on table Authors
waitfor delay '00:00:10'

update Books set title='La ci Reese transaction 2' where id=20
-- this transaction will be blocked because transaction 1 has already blocked our lock on
table Books, so, both of the transactions are blocked
commit tran

-- after some seconds transaction 2 will be chosen as a deadlock victim and terminates
with an error
-- in tables Books and Authors will be the values from transaction 1

```



	id	title	language
1	4	Panda	English
2	7	Codul lui Davinci	English
3	9	Harry Potter and The Chamber of Secrets	English
4	10	Insomnii	English
5	11	Sub dom	English
6	13	Sub dom	English
7	19	Morometii	Romanian
8	20	La ci Reese transaction 1	Romanian

	id	name
1	4	J.K. Rowling
2	5	Irina Binder
3	6	Petre Ispirescu transaction 1

Solution: For deadlock, the priority has to be set (LOW, NORMAL, HIGH, or from -10 to 10). Implicit is NORMAL (0).

For example, here we set the DEADLOCK_PRIORITY to HIGH for T2, so that T1 be chosen as a deadlock victim (T1 will have a lower priority than T2 and it will finish first).

```

-- transaction 1
begin tran
update Books set title='La ci Reese transaction 1' where id=20
-- this transaction has exclusively lock on table Books
waitfor delay '00:00:10'

update Authors set name='Petre Ispirescu transaction 1' where id=6
commit tran
-- this transaction is chose as a deadlock, because it has the lowest priority level here
(normal)

```

```

(1 row(s) affected)
Msg 1205, Level 13, State 51, Line 7
Transaction (Process ID 54) was deadlocked on lock resources with another process and has been chosen as the deadlock victim. Rerun the transaction.

```

```
-- transaction 2
SET DEADLOCK_PRIORITY HIGH
begin tran
update Authors set name='Petre Ispirescu transaction 2' where id=6
-- this transaction has exclusively lock on table Authors
waitfor delay '00:00:10'

update Books set title='La cirese transaction 2' where id=20
commit tran

-- this transaction has the higher priority level from here (set to HIGH)
-- transaction 1 finish with an error, and ans results are the ones
from this transaction (transaction 2)
```

Messages

```
(1 row(s) affected)
|
(1 row(s) affected)
```

	id	title	language
1	4	Panda	English
2	7	Codul lui Davinci	English
3	9	Harry Potter and The Chamber of Secrets	English
4	10	Insomnii	English
5	11	Sub dom	English
6	13	Sub dom	English
7	19	Morometii	Romanian
8	20	La cirese transaction 2	Romanian

	id	name
1	4	J.K. Rowling
2	5	Irina Binder
3	6	Petre Ispirescu transaction 2

- Creați un scenariu de *deadlock* prin intermediul unei aplicații .NET, folosind *multithreading*. Va trebui ca două proceduri stocate/interogări să fie executate în 2 fire de execuție diferite. Firul de execuție ce eșuează din cauza *deadlock*-ului va trebui să fie reluat (stabiliți un număr maxim de reluări până când procedura stocată/interogarea este considerată terminată fără succes - *aborted*). (nota: 10)

There are 2 possibilities: create the stored procedures in SQL Server and only use them in C# for 2 threads with locks or create everything in C#.