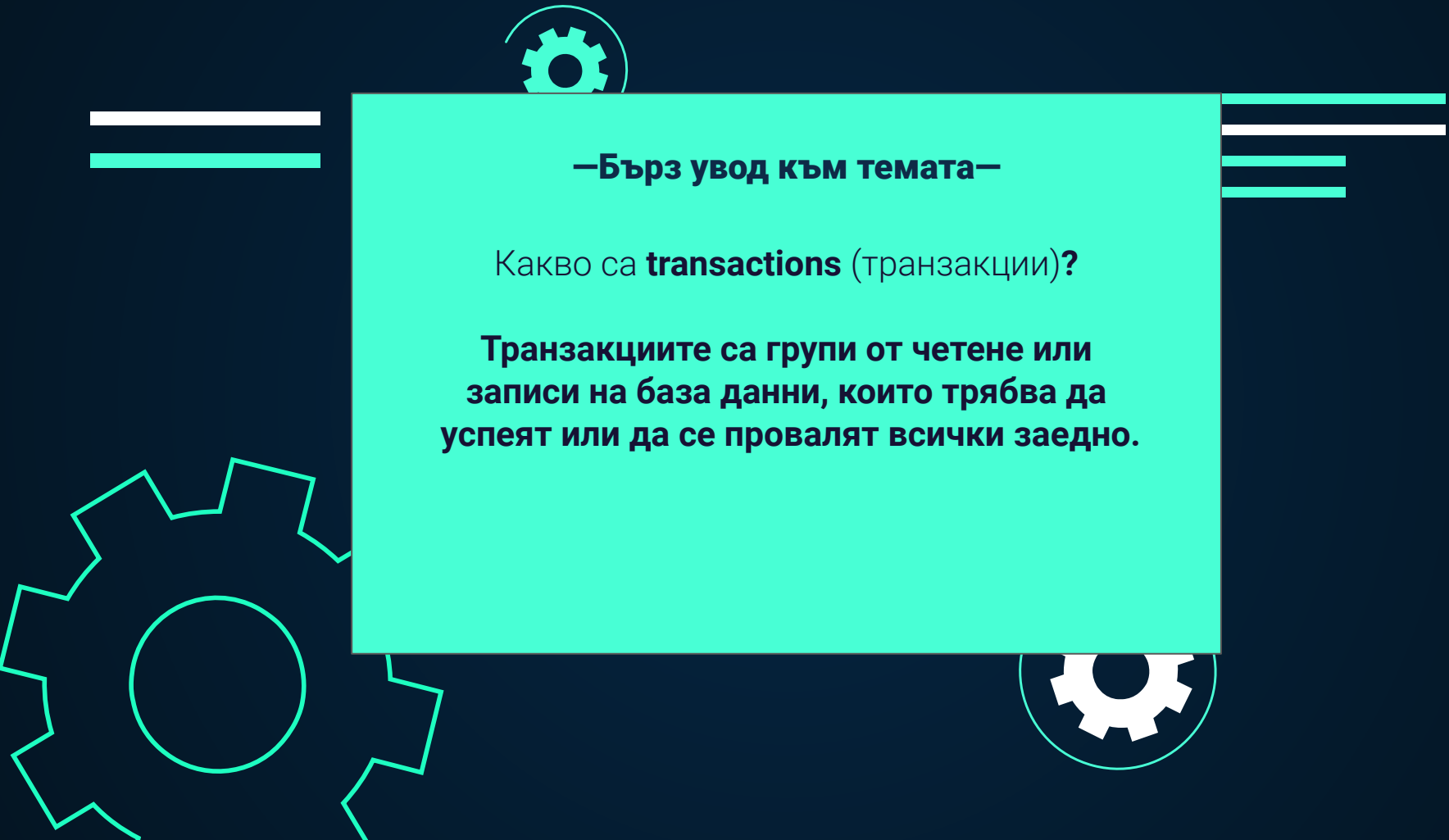
ACID в SQL

Тема по Web технологии
2020/2021 г.

Изготвена от:
Даниел Шахънски
Атанас Пашев
Йоана Стефанова



—Бърз увод към темата—

Какво са **transactions** (транзакции)**?**

Транзакциите са групи от четене или записи на база данни, които трябва да успеят или да се провалят всички заедно.

DATABASE ROLL BACK

Ако **връзката се прекъсне** преди транзакцията да завърши или ако някоя команда в транзакцията се провали, тогава базата данни се **връща назад** (отменя) **всички промени**, които е написала по време на транзакцията.

—Бърз увод към темата—

Какво точно е **ACID?**

ACID означава **Atomicity** (атомност), **Consistency** (консистентност), **Isolation** (изолация) и **Durability** (издръжливост). За да се гарантира официално валидността на дадена **транзакция**, е необходимо да се покажат тези четири характеристики.

ACID ОТБЛИЗО

Automicity

01 

Атомността гарантира, че всички команди, съставляващи транзакцията, се третират като една единица и успяват или се провалят заедно.

Consistency

02 

Последователността гарантира, че промените, направени в рамките на транзакция, са в съответствие с ограниченията, действащи върху базата данни (като уникалност на стойност за даден ключ в документи в колекция).

Isolation

03 

Изолирането гарантира, че едновременно изпълняваните транзакции не си влияят на резултатите взаимно.

Durability

04 

Устойчивостта гарантира, че след като базата данни е казала на клиента, че е записала данните, те всъщност са били записани в резервно хранилище и ще продължат да съществуват дори в случай на повреда на системата.

ACID в SQL

За демонстрация ще използваме следните две таблици (Sales, DimProduct) и техните дадени полета и записи.

```
FROM [Sales]
```

100 %

Results | Messages

	ProductKey	OrderQuantity	UnitPrice	SalesAmount
1	212	1	20.1865	20.1865
2	212	1	20.1865	20.1865
3	213	200	48.0673	9613.46

```
FROM [DimProduct]
```

100 %

Results | Messages

	ProductKey	EnglishProductName	Color	StandardCost	ListPrice	DealerPrice	StockLevel
1	212	Sport-100 Helmet, Red	Red	12.0278	33.6442	20.1865	10000
2	213	Long-Sleeve Logo Jersey, S	Multi	31.7244	48.0673	28.8404	5000
3	214	HL Road Frame - Red, 62	Red	747.9682	1263.4598	758.0759	3000
4	215	LL Road Frame - Black, 60	Black	204.6251	337.22	202.332	4000
5	216	Road-650 Black, 60	Black	486.7066	33.6442	20.1865	5000

ACID в SQL

Atomicity

```
USE [SQLTEST]
GO

BEGIN TRANSACTION
    UPDATE [DimProduct]
        SET [StockLevel] = 4700
        WHERE [ProductKey] = 213

    INSERT INTO [Sales] ([ProductKey], [OrderQuantity], [UnitPrice], [SalesAmount])
    VALUES (213, 300, 48.0673, 48.0673 * 300)
COMMIT TRANSACTION
```

100 %

Messages

```
(1 row(s) affected)

(1 row(s) affected)
```

Results | Messages

	ProductKey	EnglishProductName	Color	StandardCost	ListPrice	DealerPrice	StockLevel
1	212	Sport-100 Helmet, Red	Red	12.0278	33.6442	20.1865	10000
2	213	Long-Sleeve Logo Jersey, S	Multi	31.7244	48.0673	28.8404	4700
3	214	HL Road Frame - Red, 62	Red	747.9682	1263.4598	758.0759	3000
4	215	LL Road Frame - Black, 60	Black	204.6251	337.22	202.332	4000
5	216	Road-650 Black, 60	Black	486.7066	33.6442	20.1865	5000

	ProductKey	OrderQuantity	UnitPrice	SalesAmount
1	212	1	20.1865	20.1865
2	212	1	20.1865	20.1865
3	213	200	48.0673	9613.46
4	213	300	48.0673	14420.19

```
USE [SQLTEST]
GO

BEGIN TRANSACTION
    UPDATE [DimProduct]
        SET [StockLevel] = 4700
        WHERE [ProductKey] = 213

    INSERT INTO [Sales] ([ProductKey], [OrderQuantity], [UnitPrice], [SalesAmount])
    VALUES (213, 300, 48.0673, 'Hey! This is Wrong')
COMMIT TRANSACTION
```

100 %

Messages

```
(1 row(s) affected)
Msg 235, Level 16, State 0, Line 9
Cannot convert a char value to money. The char value has incorrect syntax.
```

Results | Messages

	ProductKey	EnglishProductName	Color	StandardCost	ListPrice	DealerPrice	StockLevel
1	212	Sport-100 Helmet, Red	Red	12.0278	33.6442	20.1865	10000
2	213	Long-Sleeve Logo Jersey, S	Multi	31.7244	48.0673	28.8404	4700
3	214	HL Road Frame - Red, 62	Red	747.9682	1263.4598	758.0759	3000
4	215	LL Road Frame - Black, 60	Black	204.6251	337.22	202.332	4000
5	216	Road-650 Black, 60	Black	486.7066	33.6442	20.1865	5000

	ProductKey	OrderQuantity	UnitPrice	SalesAmount
1	212	1	20.1865	20.1865
2	212	1	20.1865	20.1865
3	213	200	48.0673	9613.46
4	213	300	48.0673	14420.19

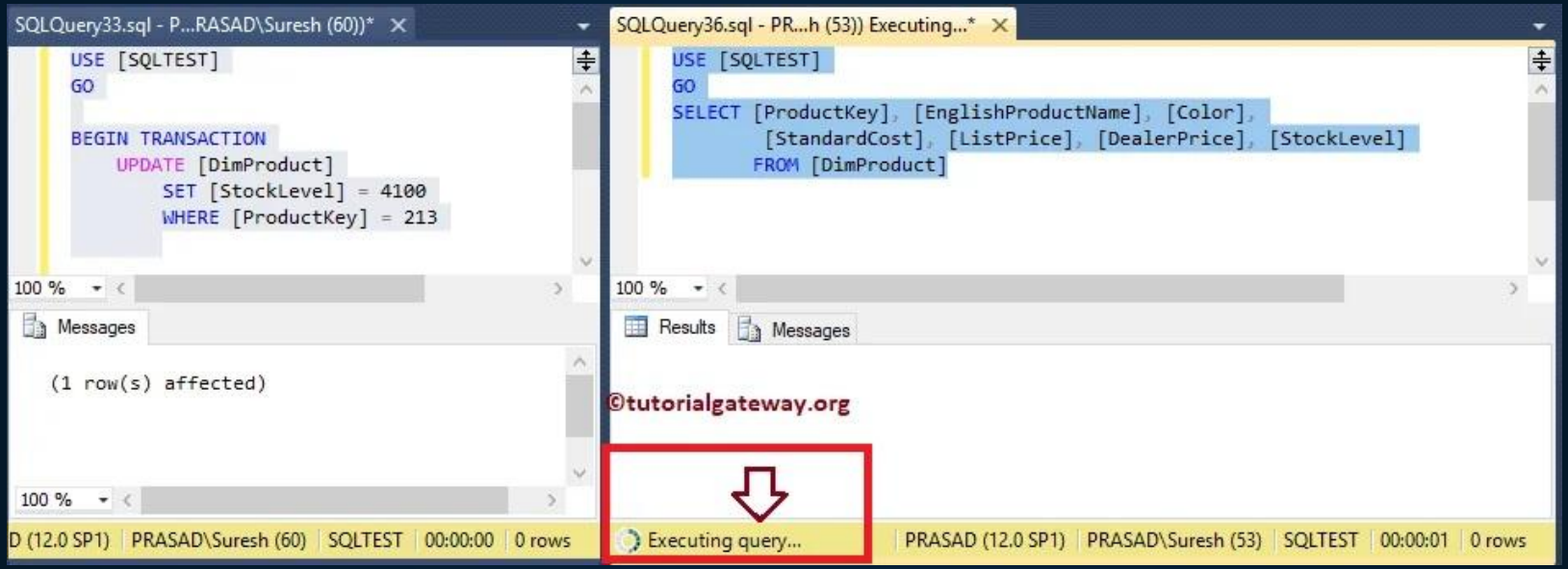
ACID в SQL

Consistency

Свойството на последователност на транзакция в SQL Server гарантира, че данните на базата данни са в постоянно състояние преди стартирането на транзакцията и също така оставя данните в постоянно състояние след завършване на транзакцията.

ACID в SQL

Isolation



ACID в SQL

Durability

Свойството на трайност на транзакцията в SQL Server гарантира, че след като транзакцията бъде успешно завършена, промените, направени в базата данни, ще бъдат постоянни.

Нарушаване на ACID в SQL

Пример:

```
CREATE TABLE acidtest (A INTEGER, B INTEGER,
CHECK (A + B = 100));
```

Нарушаване на последователността

От A изваждаме 10 без да променяме B:

- Валидация (A + B) = 90

  => отмяна на транзакцията

Нарушаване на изолацията

1) Прехвърляме 10 от A в B
2) Прехвърляме 20 от B в A

Комбинирано:

1. T1 изважда 10 от A
2. T1 добавя 10 към B
3. T2 изважда 20 от B
4. T2 добавя 20 към A

Преплетено:

1. T1 изважда 10 от A
2. T2 изважда 20 от B
3. T2 добавя 20 към A
4. T1 добавя 10 към B

Нарушаване на трайността

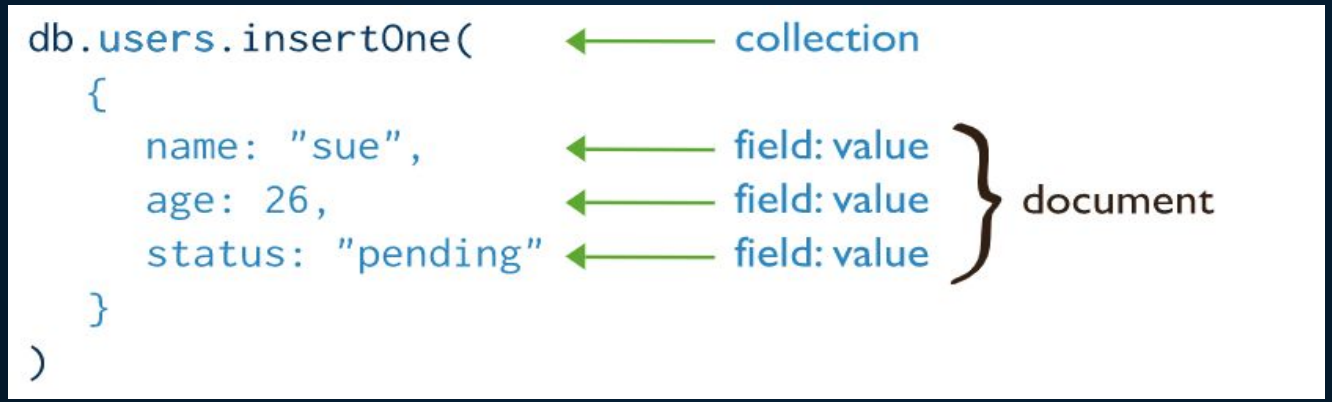
Прехвърляне на 10 от A в B

- срив в електрическото захранване

  => загуба на промените

NoSQL Database

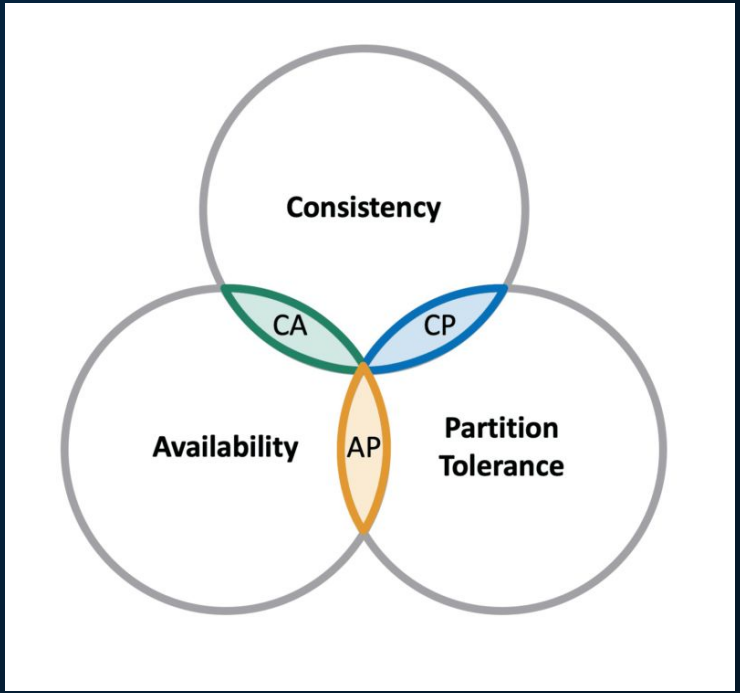
- Поява на NoSQL база данни
- MongoDB е документно ориентирана база от данни(разпределена)
  - + не е нужна схема
  - + поддържа чести промени и голям обем от данни
  - + не са нужни свързвания между отделните таблици
  - + лесно четим синтаксис
  - + хоризонтално мащабиране (scaling)



Теоремата CAP

Невъзможно е разпределена база от данни да отговаря едновременно на повече от 2 от следните изисквания:

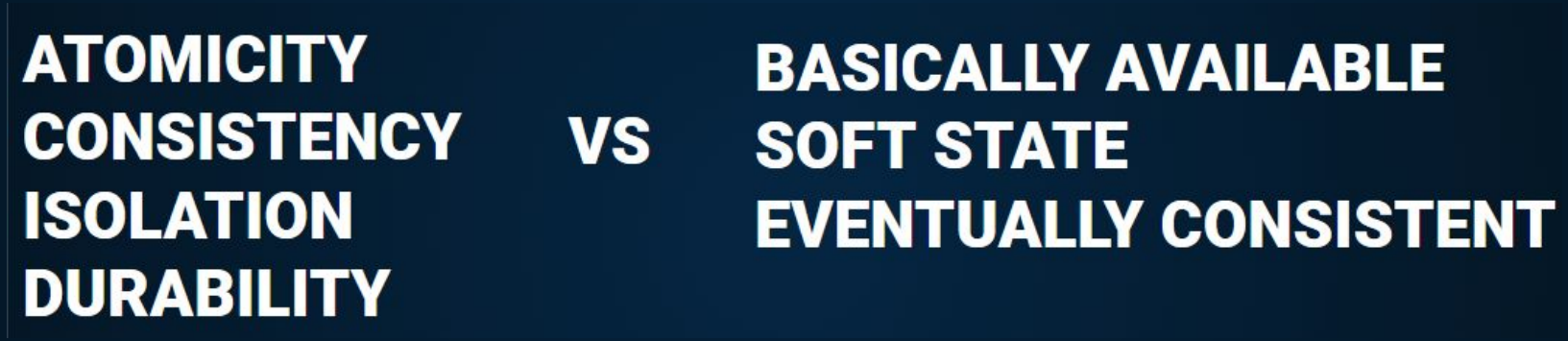
- *Консистентност*
- *Наличност*
- *Устойчивост на разпределението*



BASE свойства

Всяка NoSQL БД отговаря на следните функционалности:

- *Достъпност*
- *Променливо състояние*
- *Евентуална консистентност*



Защо да използваме MongoDB?

1. Лесна промяна на базата
2. JSON файловете могат да съдържат и структурирана, и неструктурирана информация
3. Може да се съхраняват комплексни обекти
4. Структурата зависи от разработчика
5. Архитектурата на Mongo позволява разпределение на повече и по-евтини машини, съответно е по-евтин за поддръжка
6. Поддържа голям обем от различни типове данни

MongoDB vs MySQL

selecting records

MySQL:

```
SELECT *
FROM customer
```

MongoDB:

```
db.customer.find()
```

inserting records

MySQL:

```
INSERT INTO customer (cust_id, branch, status)
VALUES ('appl01', 'main', 'A')
```

MongoDB:

```
db.customer.insert({
        cust_id: 'appl01',
        branch: 'main',
        status: 'A'
})
```

updating records

MySQL:

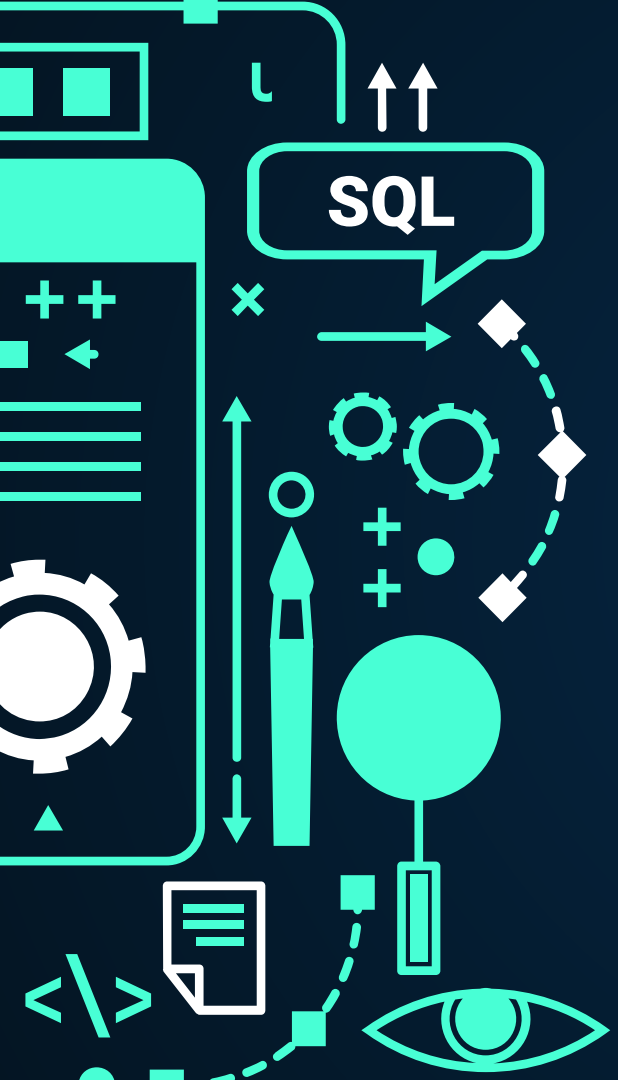
```
UPDATE customer
SET branch = 'main'
WHERE custage > 2
```

MongoDB:

```
db.customer.update({
        custage: { $gt: 2 }
  },
  {
        $set: { branch:'main' }
  },
  {
        multi: true
})
```

ИЗПОЛЗВАНА ЛИТЕРАТУРА
и полезни препратки..

- https://en.wikipedia.org/wiki/ACID
- https://www.tutorialgateway.org/acid-properties-in-sql-server
- https://dotnettutorials.net/lesson/acid-properties-in-sql-server/
- https://docs.microsoft.com/en-us/sql/relational-databases/logs/control-transaction-durability?view=sql-server-ver15
- https://www.mongodb.com/basics/transactions
- https://dzone.com/articles/how-acid-mongodb
- https://www.mongodb.com/blog/post/mongodb-multi-document-acid-transactions-general-availability
- https://cloudxlab.com/assessment/displayslide/342/nosql-acid-properties-and-rdbms-story
- https://neo4j.com/blog/acid-vs-base-consistency-models-explained/
- https://www.lifewire.com/abandoning-acid-in-favor-of-base-1019674



Благодарим за вниманието!

Изготвили:

Даниел Шахънски 71868
Атанас Пешев 71837
Йоана Стефанова 71854