

Исследование  
эффективности  
применения индексов  
ColumnStore при  
выполнении  
SQL - запросов в Microsoft  
SQL Server 2016

Выполнил: студент группы 14-При Шилина Алина

Научный руководитель: Коптенок Е.В.

# Проблема

## ИССЛЕДОВАНИЯ

Все более сложные приложения взаимодействуют с базами данных. Причем получение данных из базы является одним из узких мест приложения. Таким образом, оптимизация запросов к базе способна существенно повысить производительность приложения в целом.



Microsoft®  
**SQL Server**®

# Проблема

The screenshot shows Microsoft SQL Server Management Studio with a query execution error. The query in the query editor is:

```
select getdate()  
SELECT *  
FROM  
market3  
select getdate()
```

The error message in the Results pane is: (Отсутствует имя столбца). The status bar at the bottom indicates the execution was successful: "Выполнение запроса... LAPTOP-P4FN2SFJ (13.0 RTM) LAPTOP-P4FN2SFJ\Admin ... Conference 00:00:51 0 строк".

The Results pane shows a table with the following data:

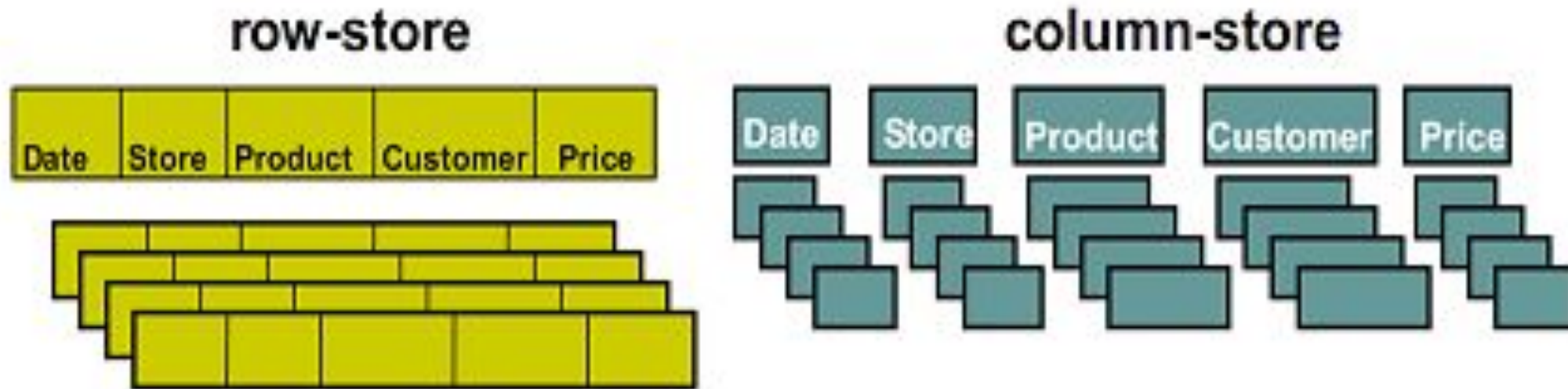
id	type	name	vendor	cost	col	comment	
1	103011	13	Serial102012	13	2112.00	22	это описание товара
2	103026	13	Serial102027	13	2127.00	37	это описание товара
3	103041	13	Serial102042	13	2142.00	12	это описание товара
4	103056	13	Serial102057	13	2157.00	27	это описание товара
5	103071	13	Serial102072	13	2172.00	42	это описание товара
6	103086	13	Serial102087	13	2187.00	17	это описание товара
7	103101	13	Serial102102	13	2102.00	32	это описание товара
8	103116	13	Serial102117	13	2117.00	47	это описание товара

The interface also shows the Object Explorer on the left with the 'Conference' database selected, and the Properties window on the right showing connection details for 'LAPTOP-P4FN2SFJ'.

# Индексы

## COLUMN-STORE

Отличительная особенность колоночных индексов в том, что они основаны на колоночном хранении данных.



*Рис. 1 Отличие column-store от row-store*

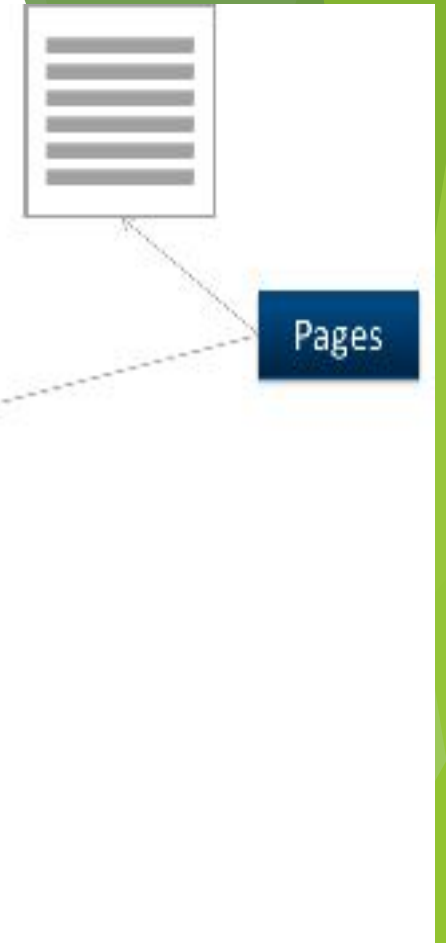
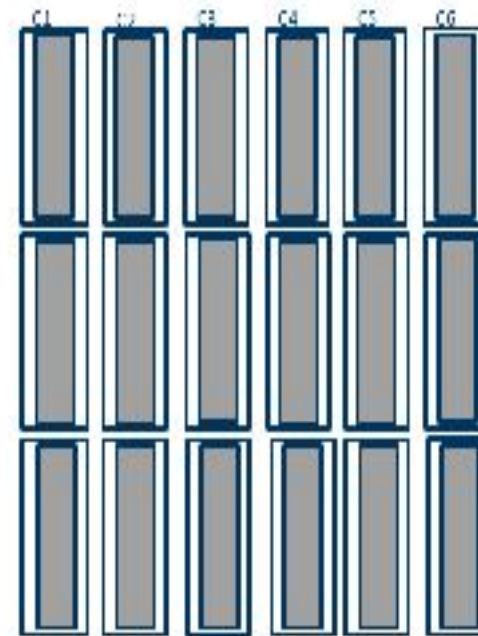
# Индексы COLUMN

Некластеризованный индекс columnstore и кластеризованный индекс columnstore — это одно и то же. Разница в том, что некластеризованный индекс вторичен и создается на основе таблицы индексов rowstore, а кластеризованный индекс columnstore является первичным для всей страницы.

Row store:



Column store:



*Рис. 2 Отличие column-store от row-store*

# Структура базы данных

База наполнена тестовыми данными:

Vendor — 1000 записей

Type — 1000 записей

Market1 — 1000 записей

Market2 — 500 000 записей.

Market3 — 3 млн. записей.

Market4 — 18 млн. записей

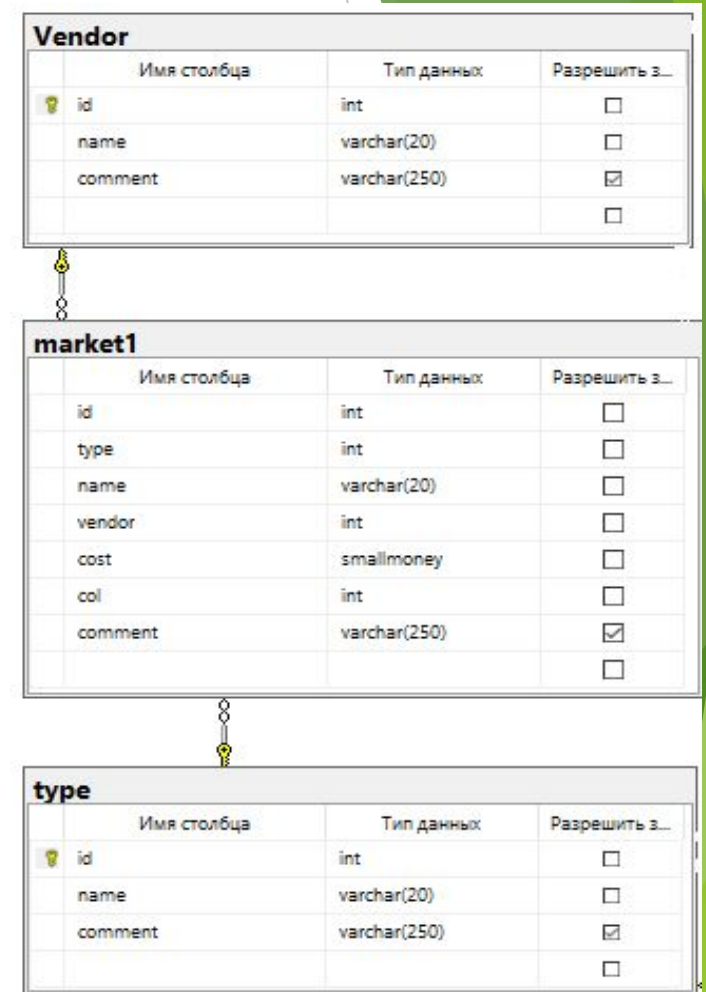


Рис. 4 Диаграмма базы данных

# Запросы к

## таблицам

```
SELECT DISTINCT t.name AS 'Название товара', v.name AS 'Название производителя'  
FROM market3 m  
JOIN Vendor v ON (m.vendor=v.id) JOIN type t ON (m.type=t.id)  
WHERE m.col IN  
(SELECT DISTINCT col FROM market1 WHERE col BETWEEN 11 AND 45);
```

*Рис. 5 Пример запроса*

Для того чтобы оценить эффективность использования некластеризованных индексов column-store, мной было составлено несколько запросов различной структуры. Пример одного из них приведен на рис.5.

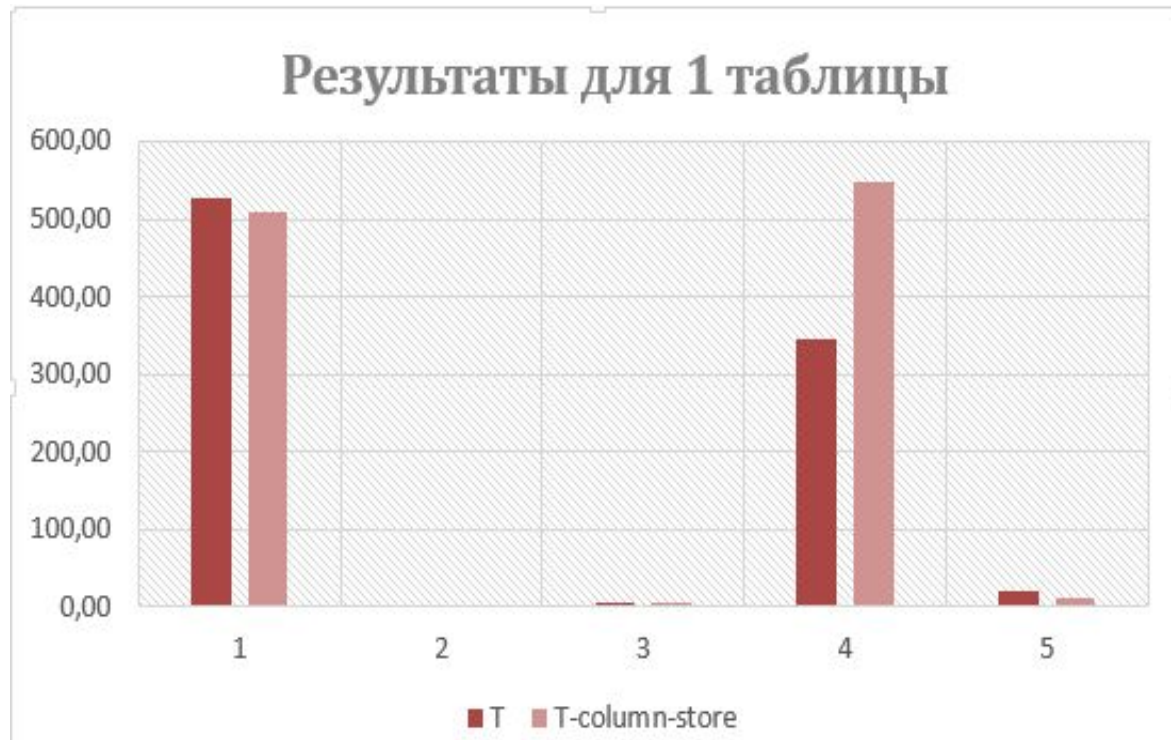
# Запросы к

Запрос №1	Запрос №2	Запрос №3
SELECT *	SELECT type AS 'Тип', COUNT(*) AS 'Кол-во товара данного типа'	SELECT t.name, AVG(m.cost) AS 'Средняя стоимость данного товара'
FROM market3	FROM market1	FROM type t
WHERE type<=6	GROUP BY type	CROSS JOIN market3 m
	ORDER BY type	where m.type=t.id
		GROUP BY t.name
		ORDER BY t.name
Запрос №4	Запрос №5	
SELECT *	SELECT DISTINCT t.name AS 'Название товара', v.name AS 'Название производителя'	
FROM	FROM market3 m	
market3	JOIN Vendor v ON (m.vendor=v.id) JOIN type t ON (m.type=t.id)	
	WHERE m.col IN	
	(SELECT DISTINCT col FROM market1 WHERE col BETWEEN 11 AND 45);	

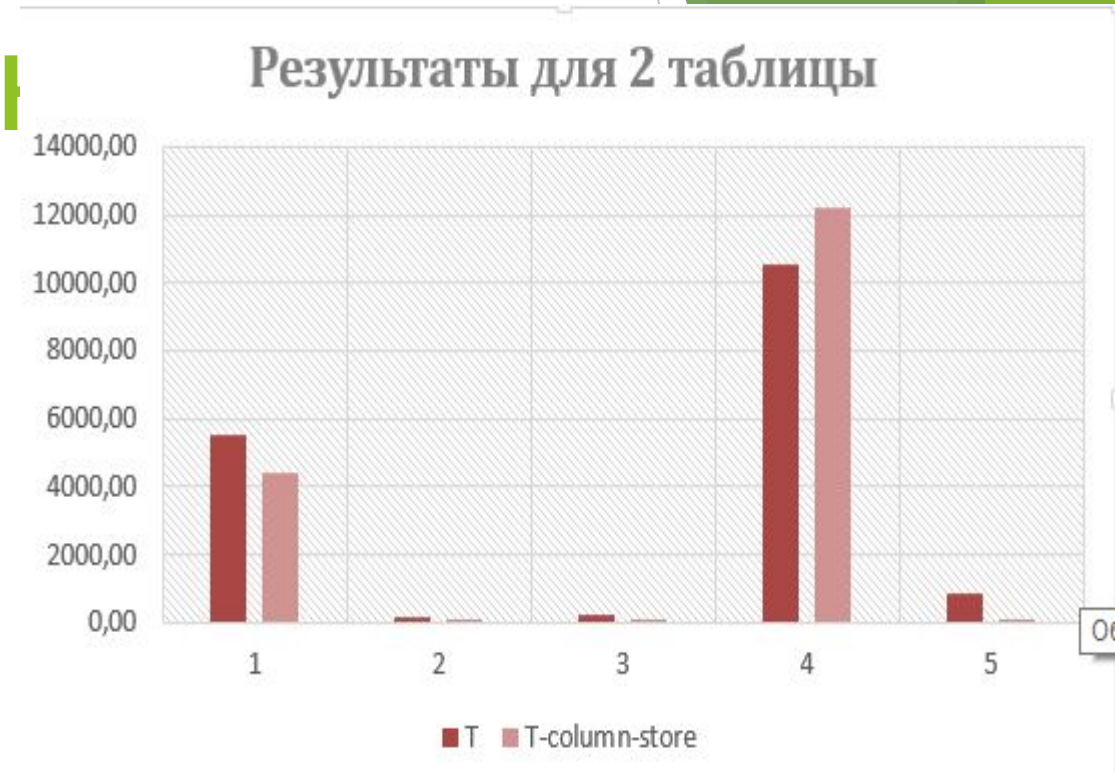
*Рис. 6 Запросы*



# Результаты



а



*Рис. 7 Результаты для таблицы с 1000 записями*

*Рис. 8 Результаты для таблицы с 500 000 записями*

По оси ОХ отображается номер запроса, а по ОУ – время выполнения запроса в мс. Красный столбец показывает время без использования индексов. Розовым – с использованием Column-Store.

# Результаты исследований

Результаты для 3 таблицы

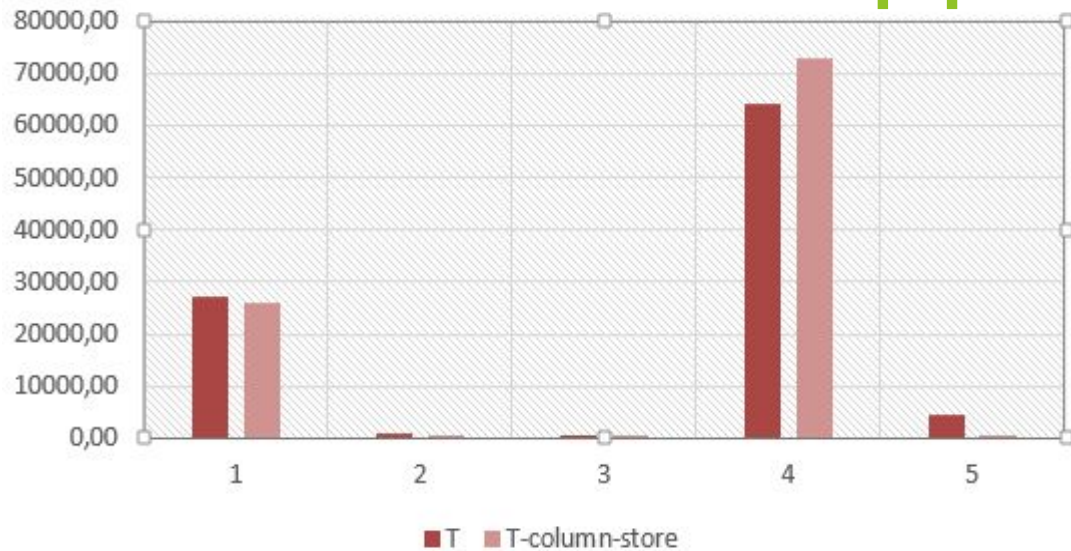


Рис.9 Результаты для таблицы с 3 млн. записями

Результаты для 4 таблицы

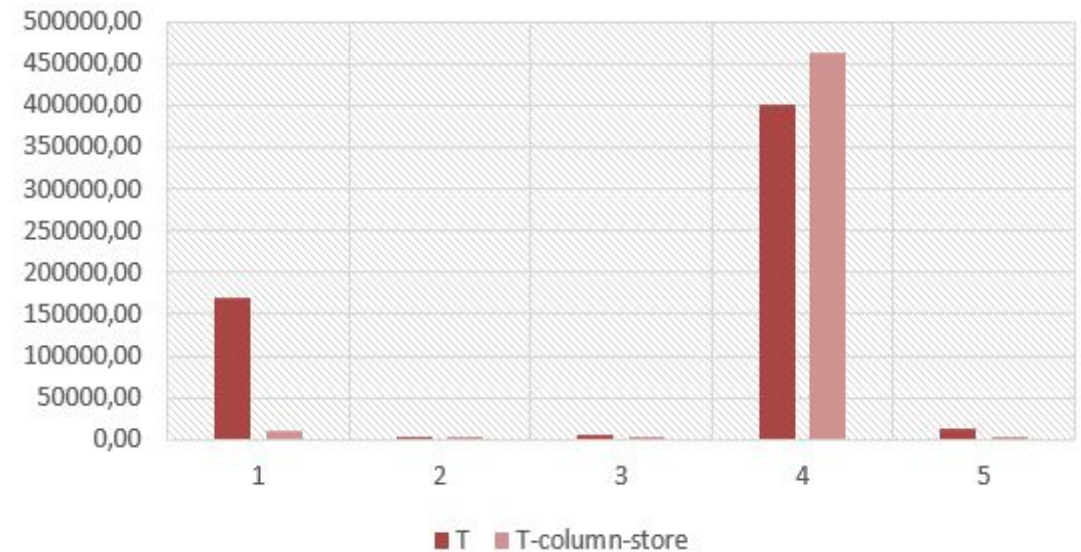


Рис. 10 Результаты для таблицы с 18 млн. записями

По оси ОХ отображается номер запроса, а по ОУ – время выполнения запроса в мс. Красный столбец показывает время без использования индексов. Розовым – с использованием Column-Store.

# Результаты исследований

Результаты для таблицы с 18 млн записями

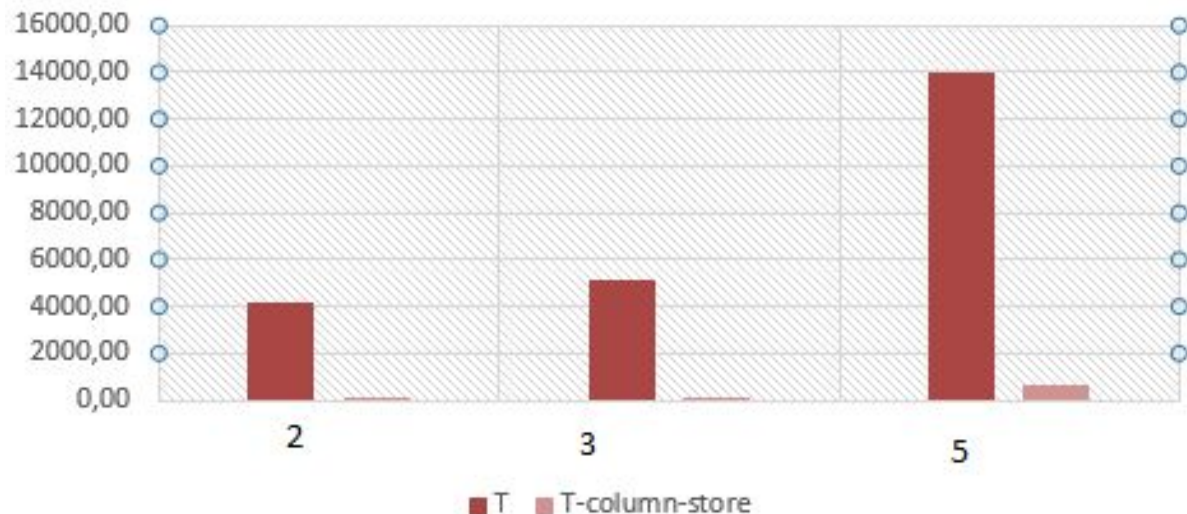


Рис. 11 Результаты для таблицы с 18 млн.записями

Результаты для таблицы с 1000 данными

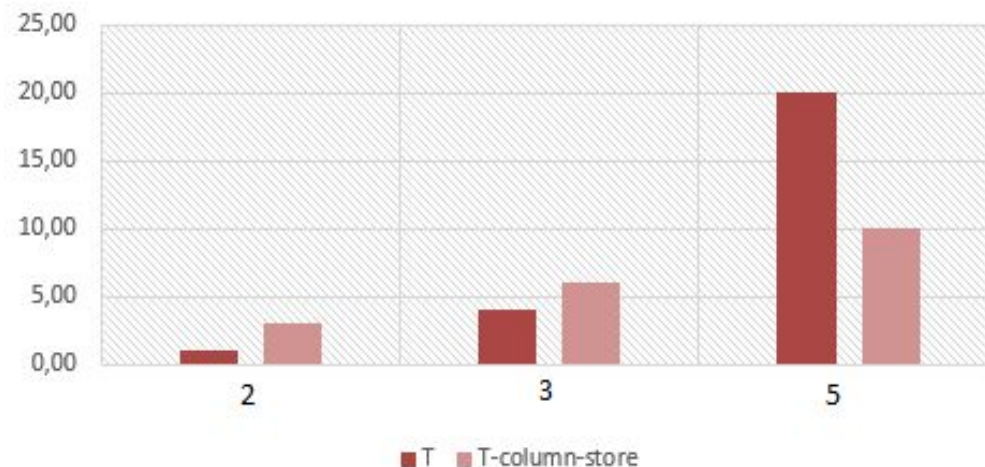


Рис. 12 Результаты для таблицы с 1000 записями

По оси ОХ отображается номер запроса, а по ОУ – время выполнения запроса в мс. Красный столбец показывает время без использования индексов. Розовым – с использованием Column-Store.

# Результаты исследований

Проанализировав полученные результаты можно сделать вывод, что:

- Column-Store лучше работают с таблицами, содержащими большое количество строк;
- Column-Store повышает производительность в 10 раз;
- Column-Store позволяет добиться высокого уровня сжатия данных;
- Не следует использовать Column-Store при малых объемах данных и простых запросах
- Column-Store следует использовать при тяжелых запросах.

Спасибо за  
внимание!



SQL

