

# 8. План выполнения запроса. Ограничения целостности

к.т.н., доцент кафедры ИиСП

Лучинин  
Захар Сергеевич

# План выполнения запроса

**План выполнения запроса** - последовательность операций, необходимых для получения результата SQL-запроса.

```
SELECT  
  FROM Продажа  
  ORDER BY Номер_продавца
```

1. Сканирование кластерного индекса по первичному ключу таблицы **Продажа**.
2. Сортировка результатов шага 1 по столбцу **Номер\_продавца**.
3. Возврат приложению результатов шага 2.

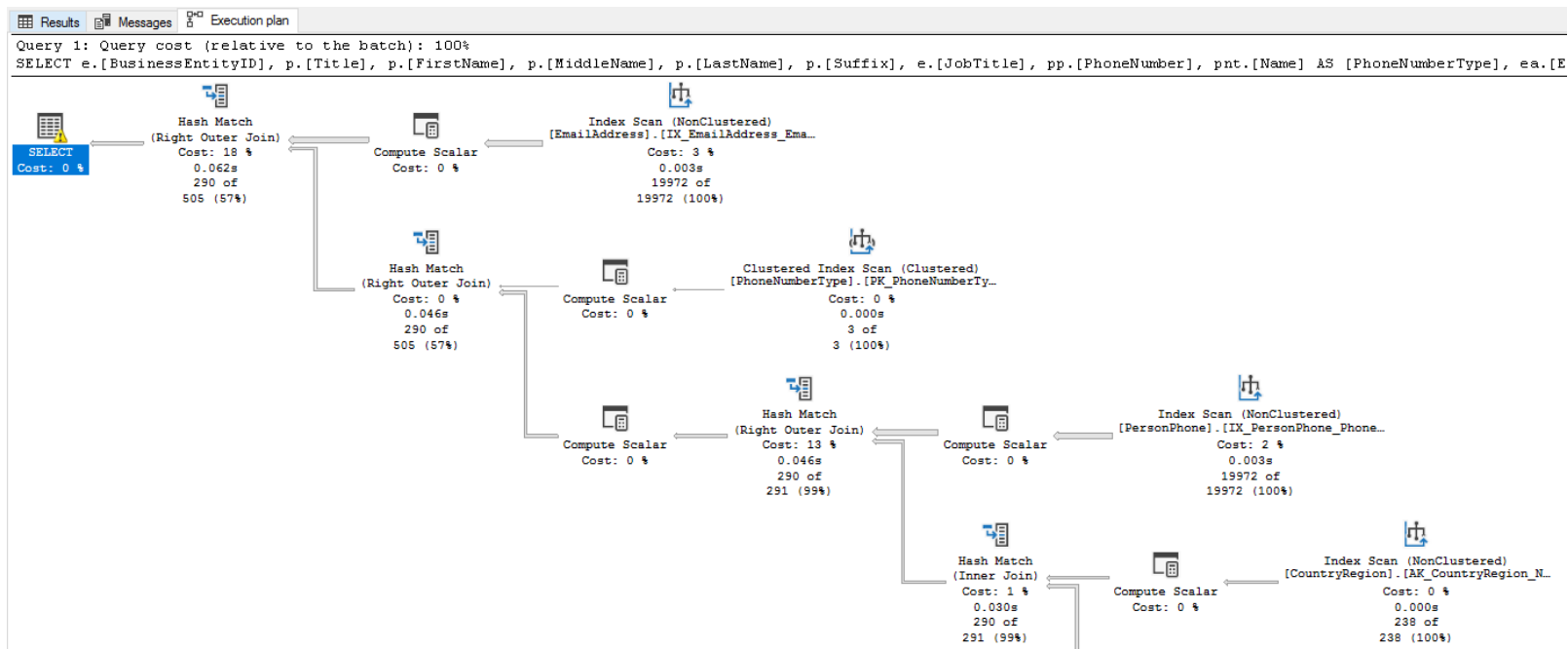
# План выполнения запроса

- План выполнения запроса представляет собой:
  - Последовательности, в которой происходит обращение к исходным таблицам.
  - Методы, используемые для извлечения данных из каждой таблицы.
  - Методы, используемые для вычислений, а также фильтрации, сортировки данных из каждой таблицы.

# План выполнения MSSQL

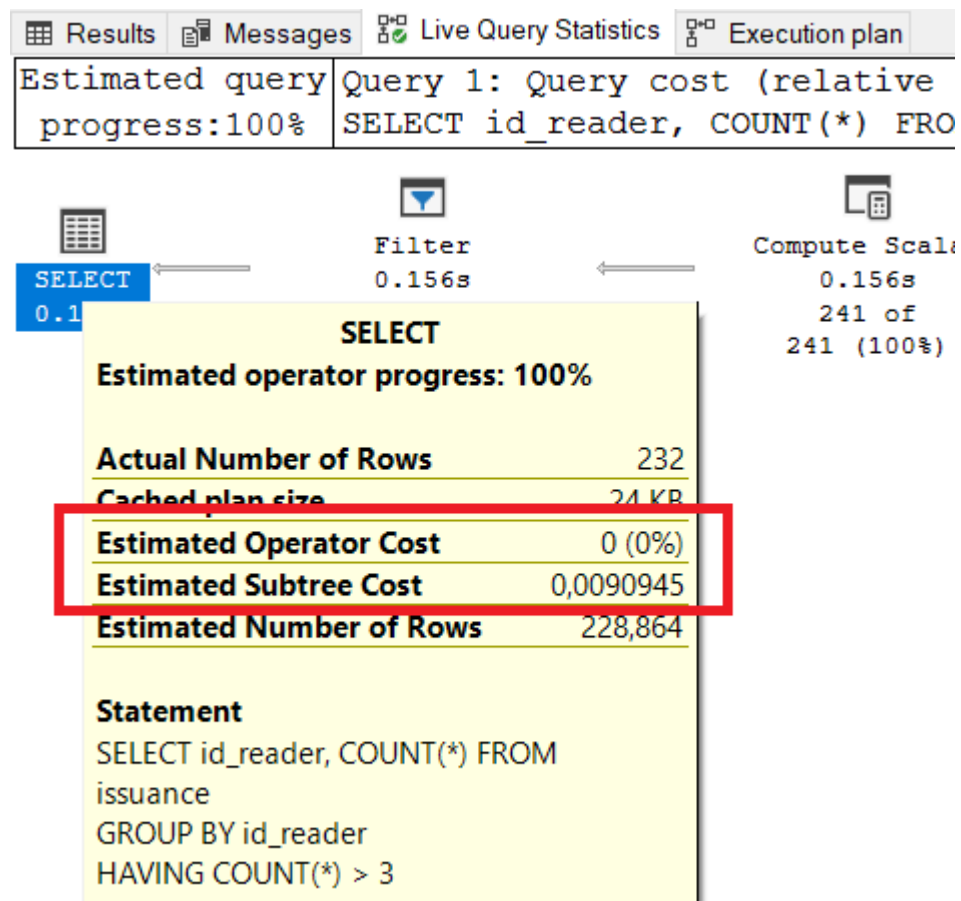
- Actual Execution Plan (Предполагаемый план выполнения)
- Estimated Execution Plan (Действительный план выполнения)
- Statistics (Статистика активных запросов)

# План выполнения MSSQL



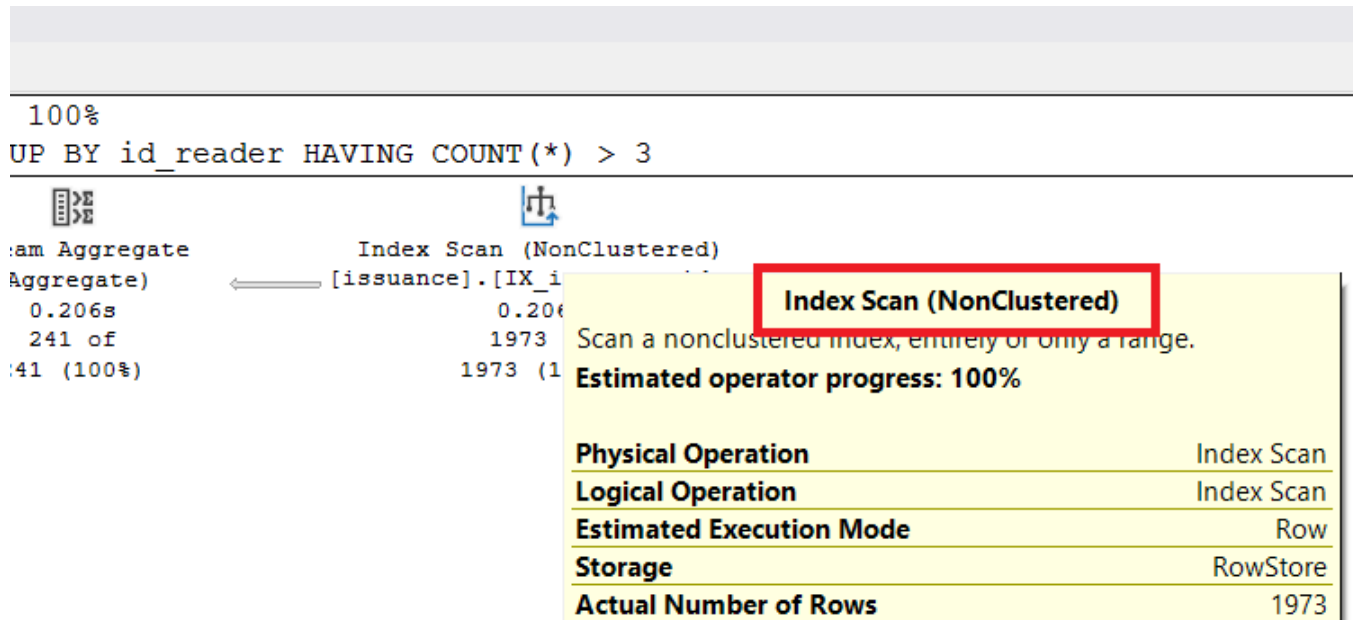
# Стоимость запроса

- **Operator cost** – стоимость запроса в процентном выражении относительно всего пакета запросов.
- **Subtree Cost** – суммарная стоимость всех операций внутри дерева



# Seek vs Scan

- **Scan** – Сканирование обычно худший вариант с точки зрения производительности, поскольку в поисках нужной информации просматривается весь индекс.
- **Seek** – Поиск в отличие от сканирования — лучший вариант использования индекса



Physical Operation	Index Scan
Logical Operation	Index Scan
Estimated Execution Mode	Row
Storage	RowStore
Actual Number of Rows	1973

# Оптимизация запросов СУБД

Планы выполнения запроса сравниваются исходя из множества факторов, например:

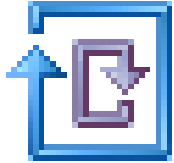
- потенциальное число строк, извлекаемое из каждой таблицы, получаемое из статистики;
- использование индексов;
- способ выполнения слияния таблиц;
- способ чтения записей/блоков таблиц/индексов.



# При оптимизации обрати внимание

1. Стоимость запроса
2. Последовательность операций
3. Детали дорогих операций
4. Толщина стрелок

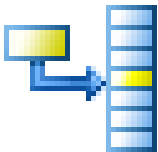
# Алгоритмы объединения двух таблиц



**Nested Loops** - базовый алгоритм объединения двух таблиц. Суть очень простая — для каждого элемента первой таблицы пройдемся по всем элементам второй таблицы; если ключи элементов оказались равны — запишем совпадение в результирующую таблицу.






**Merge Join** (объединение слиянием) теоретически является самым быстрым физическим оператором объединения, однако требуются, чтобы данные обоих входов были отсортированы.



**Hash Join** - объединение через просчет хеш значений ключей

# Справочник по логическим и физическим операторам

	<b>Если оператор</b>	Оператор <b>If</b> выполняет условную обработку в зависимости от значения выражения. <b>If</b> является элементом языка.
None	<b>Inner Join</b>	Логический оператор <b>Inner Join</b> возвращает каждую строку, которая удовлетворяет соединению первого (верхнего) входа со вторым (нижним).
	<b>Insert</b>	Логический оператор <b>Insert</b> вставляет строки из входного потока в объект, заданный в столбце <b>Argument</b> . Соответствующим физическим оператором является <b>Table Insert</b> , <b>Index Insert</b> или <b>Clustered Index Insert</b> .
	<b>Inserted Scan</b>	Оператор <b>Inserted Scan</b> просматривает таблицу <b>inserted</b> . <b>Inserted Scan</b> является логическим и физическим оператором.
	<b>Intrinsic</b>	Оператор <b>Intrinsic</b> вызывает внутреннюю функцию Transact-SQL . <b>Intrinsic</b> является элементом языка.

<https://docs.microsoft.com/ru-ru/sql/relational-databases/showplan-logical-and-physical-operators-reference?view=sql-server-ver15>

# Поиск медленных запросов

- Slow log – список медленных запросов. PostgreSQL и MySQL.
- Extended Events – упрощенная система мониторинга производительности в MSSQL.
- Сторонние системы мониторинга. ELK
- MSSQL Query Store

# Slow log в MySQL

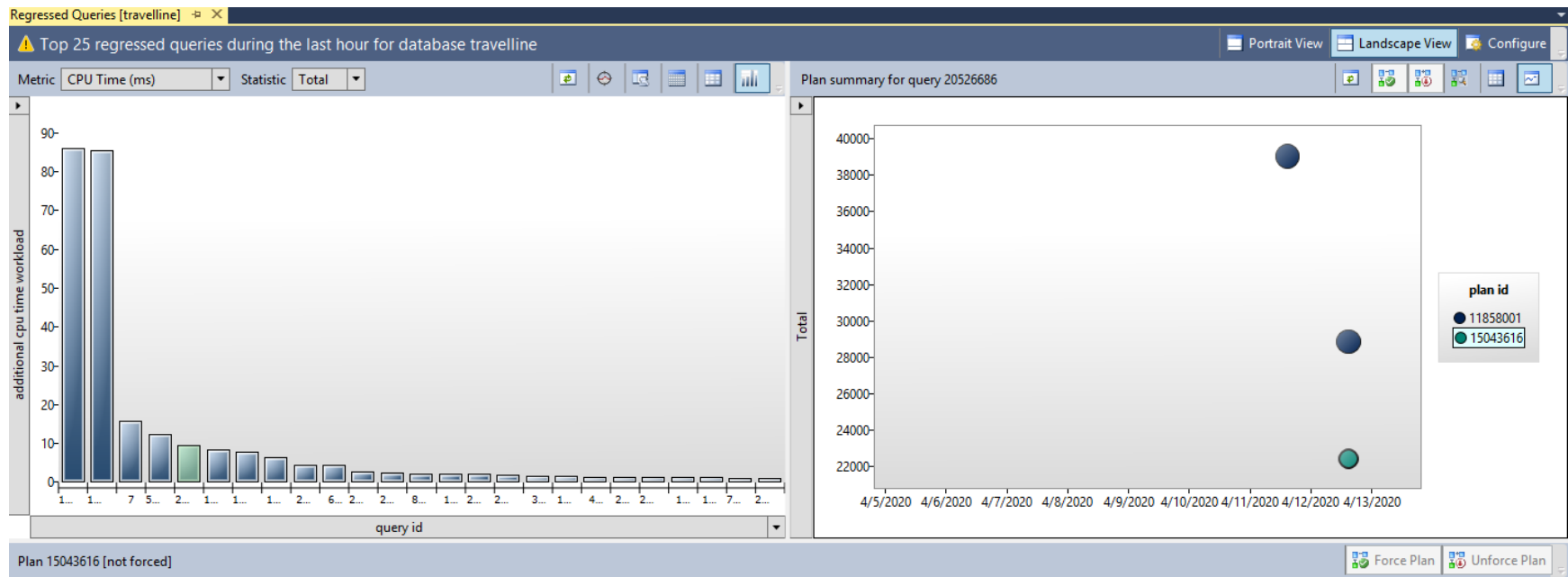
- В **MySQL** есть встроенный функционал для ведения логов медленных запросов.

*log-slow-queries=/tmp/slow\_queries.log*  
*long\_query\_time=10*

## Ложные срабатывания

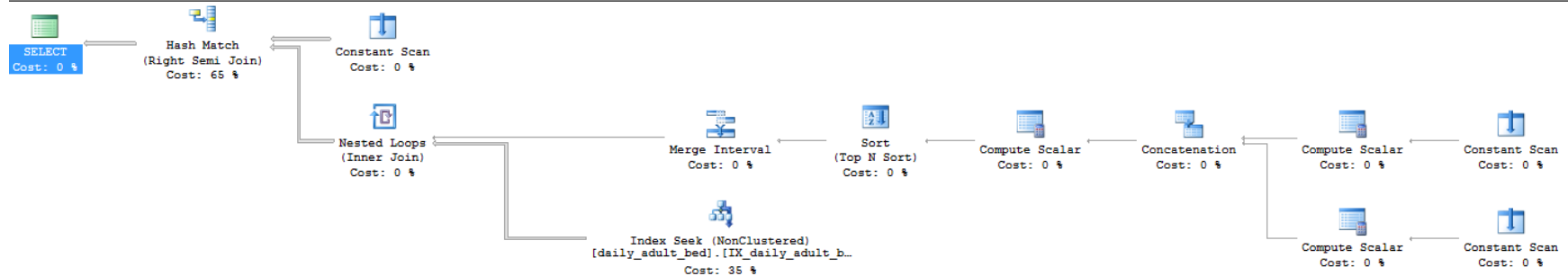
- Таблица может быть заблокирована (locked)
- Данные и индексы не были занесены в кэш память.
- Был запущен сторонний процесс, замедляющий работу диска.
- Сервер перегружен другими запросами в это время, и не хватает ресурсов CPU для эффективной работы.

# MSSQL Query Store



Query 1: Query cost (relative to the batch): 100%

```
SELECT * FROM [daily_adult_bed] WHERE [id_season_calendar] = @season_calendar AND [id_channel] IN (@id_channel0,@id_channel1,@id_channel2,@id_channel3,@id_channel4,@id_ch
```



# Extended Events

**Session Properties**

Ready

Select a page: General, Events, Data Storage, Advanced

Script | Help

Select

Configure the events to collect additional fields and specify filters.

Selected events:

Name	Count	✓
rpc_completed	5	✓
sp_statement_completed	5	✓
sql_batch_completed	5	✓
sql_statement_completed	5	✓

Event configuration options:

Global Fields (Actions) | Filter (Predicate) | Event Fields

And/Or	Field	Operator	Value
	sqlserver.is_system	equal_boolean	0
And	physical_reads	greater_than_uint...	2000
Or	writes	greater_than_uint...	500
Or	cpu_time	greater_than_uint...	10000000
Or	duration	greater_than_uint...	5000000
And	connection_reset_option	=	

Click here to add a clause:

- cpu\_time
- data\_stream
- duration
- logical\_reads
- object\_name
- output\_parameters
- physical\_reads
- result
- row\_count
- statement
- writes
- package0.counter
- package0.current\_thread\_id
- package0.last\_error
- package0.partitioned\_counter
- sqlqs.cpu\_id
- sqlqs.numa\_node\_id
- sqlqs.scheduler\_address
- sqlqs.scheduler\_id
- sqlqs.system\_thread\_id
- sqlqs.task\_address
- sqlqs.task\_elapsed\_quantum
- sqlqs.task\_execution\_time
- sqlqs.task\_resource\_group\_id
- sqlqs.task\_resource\_pool\_id
- sqlqs.worker\_address
- sqlserver.client\_app\_name
- sqlserver.client\_connection\_id
- sqlserver.client\_hostname

**Connection**

spbsql2  
(TRAVELLINE\zakhar.luchinin)

[View connection properties](#)

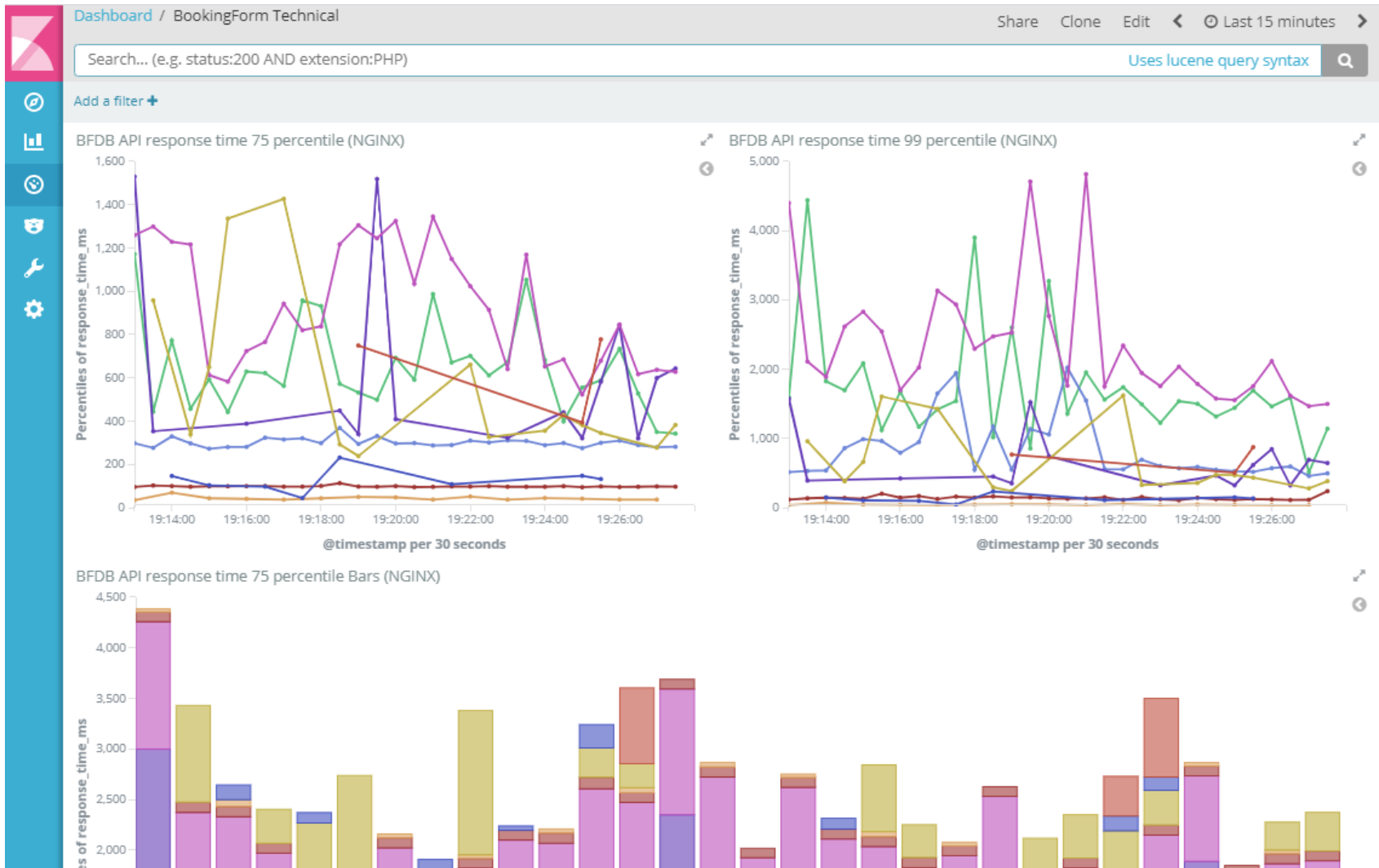
**Progress**

Ready

**rpc\_completed**

Occurs when a remote procedure call has completed.

# ELK. Kibana





# EXPLAIN

- Для анализа запросов в MySQL используется инструкция EXPLAIN

EXPLAIN [запрос]

- Результатом выполнения является таблица (план выполнения запроса)

EXPLAIN **SELECT \* FROM `users` WHERE** email = 'hello@gmail.com';

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	users	ALL	NULL	NULL	NULL	NULL	336	Using where

# Использование EXPLAIN для анализа использования индексов

EXPLAIN **SELECT** \* **FROM** `users` **WHERE** email = 'hello@gmail.com';

**До создания индекса:**

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	users	ALL	NULL	NULL	NULL	NULL	336	Using where

**После создания индекса:**

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	users	const	email	email	386	const	1	

# Статистика

- Статистика – это информация о распределении данных в столбце или столбцах
- Используется для поиска наилучшего плана выполнения
- Данные статистики измеряются двумя различными способами:
  - Плотность
  - Распределение данных

# Обновление статистики

- Производите первый поиск по столбцу
  - Изменилось ~20% таблицы
  - Создаете индекс
- 
- Требуется обновление статистики администратором БД

# Целостность и истинность

- **Целостность БД не гарантирует** достоверности (истинности) содержащейся в ней информации, но обеспечивает по крайней мере правдоподобность этой информации, отвергая заведомо невероятные, невозможные значения.
- **Достоверность (или истинность)** есть соответствие фактов, хранящихся в базе данных, реальному миру.

# Категории целостности данных

## Domain integrity / Доменная целостность

Домен – множество допустимых значений столбца

- Типы данных
- CHECK
- NOT NULL
- FOREIGN KEY
- DEFAULT

## Entity integrity / Целостность сущностей

Определяет строку как уникальную сущность в конкретной таблице

- PRIMARY KEY
- UNIQUE

## Referential integrity / Ссылочная целостность

- Сохраняет определенные связи между таблицами при добавлении или удалении строк.

- Гарантирует согласованность значений ключей во всех таблицах

- FOREIGN KEY

## User-defined integrity / Пользовательские ограничения

# Нормальные формы

Нормальные Формы позволяют:

- Бороться с избыточностью данных;
- Избегать создания заведомо ложных кортежей путем уничтожение транзитивных зависимостей;
- Упрощать поддержание целостности базы данных.

# Ограничение NOT NULL

Ограничение NOT NULL используется для тех столбцов таблицы, которые требуют, чтобы значение было всегда.

ID	NAME	JOB	HIREDATE	SAL	ID_DEPT
7329	SMITH	CEO	16.12.85	9000	20
7499	ALLEN	VP-SALES	02.04.90	7500	30
7521	WARD	MANAGER	23.08.91	5000	30
7566	JONES	SALEMAN	12.01.90	3700	NULL

NOT NULL



HE NOT NULL





# Индекс UNIQUE

Ограничения с помощью уникального индекса UNIQUE проверяет столбец на уникальность атрибута или набора атрибутов.

ID	NAME	EMAIL
1	IVAN I	<a href="#">IVAN@M.RU</a>
2	PETR I	<a href="#">PETR@M.RU</a>
3	ALEX	<a href="#">ALEX@M.RU</a>
	<del>IVAN III</del>	<del><a href="#">IVAN@M.RU</a></del>

Значение [IVAN@M.RU](#) уже существует



# Ограничение PRIMARY KEY

- Ограничение PRIMARY KEY для таблицы или столбца означает, что группа из одного или нескольких столбцов образуют потенциальный ключ таблицы.
- Для одной таблицы может быть определено единственное ограничение PRIMARY KEY.

# Атрибут и тип данных

- Тип данных как ограничение
- Длина строковых типов данных
- Атрибут unsigned
- Название атрибута

# Ссылочная целостность

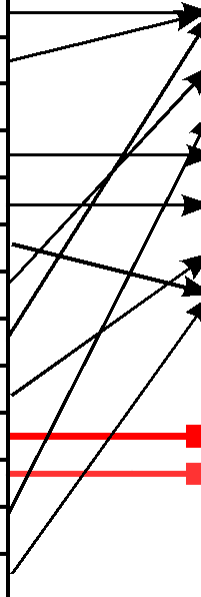
**Ссылочная целостность (referential integrity)** — необходимое качество реляционной базы данных, заключающееся в отсутствии в любом её отношении внешних ключей, ссылающихся на несуществующие кортежи.

**Address (адрес)**

Key	House	Apart	Street
12	52	1	15
15	12-а	2	15
85	9	3	NULL
152	3	4	120
254	85-6	5	122
374	132	6	150
495	2	7	35
514	52	8	15
632	75	9	130
887	56	10	155
994	47	11	12
1021	32	12	84
4511	14	13	150

**Street (улица)**

Key	Prefix	Name
15	ул	Ленина
35	ул	Энгельса
84	ул	Дзержинского
120	ул	Клары Цеткин
122	ул	Розы Люксембург
130	пр	Московский
150	пр	Победы



# FOREIGN KEY

**Внешний ключ (FK)** — это столбец или сочетание столбцов, которое применяется для принудительного установления связи между данными в двух таблицах с целью контроля данных, которые могут храниться в таблице внешнего ключа.

```
ALTER TABLE [product]
ADD CONSTRAINT FK_product_product_category_id FOREIGN KEY
(id_product_category)
REFERENCES [dbo].[product_category] (id_product_category)
ON DELETE CASCADE
ON UPDATE CASCADE
```

# Действия внешнего ключа

- **NO ACTION (RESTRICT)**

Операция формирует ошибку, после чего выполняется откат операции удаления или обновления строки в родительской таблице.

- **SET NULL**

Всем значениям, составляющим внешний ключ, присваивается значение NULL, когда обновляется или удаляется соответствующая строка в родительской таблице. Для выполнения этого ограничения внешние ключевые столбцы должны допускать значения NULL.

- **CASCADE**

Соответствующие строки обновляются или удаляются из ссылающейся таблицы, если данная строка обновляется или удаляется из родительской таблицы.

# Индексы в ограничениях внешнего ключа

При создании ограничения внешнего ключа необходимо:

- **Создать индекс** на атрибут в зависимой таблице
  - Столбцы внешнего ключа часто используются в критериях соединения при совместном применении в запросах данных из связанных таблиц.
  - С помощью ограничений внешнего ключа в связанных таблицах проверяются изменения ограничений первичного ключа.
- Убедиться в **полном соответствии** типов данных