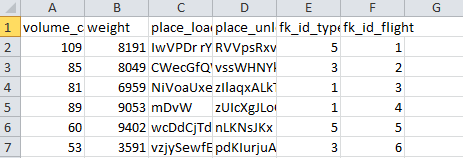
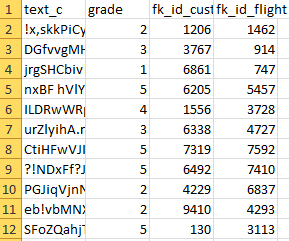
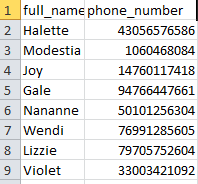
|  |  |  |
| --- | --- | --- |
| Лабораторная работа №5  Индексы, хранимые процедуры, представления  Тема: грузоперевозки (В-13) | Ф.И.О. | Козякин А.Д. |
| Группа | ИВТ-263 |
| Преподаватель | Соколов А.А. |
| Дата сдачи |  |

**Цель работы:** изучить принцип работы индексов, процедур и представлений в SQL.

**Ход работы:**

Сначала создадим генератором данных 10000 строк данных для большинства таблиц в формате csv. Примеры csv:



Теперь пропишем скрипты для добавления этих данных:

/\*добавлены 10000 городов V\*/

load data infile 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\cities.csv' INTO TABLE city FIELDS TERMINATED BY ',' IGNORE 1 LINES (name\_city);

/\*добавлены через csv 10000 рабочих V\*/

load data infile 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\workers.csv' INTO TABLE worker FIELDS TERMINATED BY ';' IGNORE 1 LINES (full\_name, phone\_number);

insert into type\_cargo (factor\_cargo, name\_type\_cargo)

values

(1.4, 'Стройматериалы (твёрдые)'),

(1.75, 'Наливной (топливо)'),

(1.34, 'Негабаритный груз (запчасти)'),

(1.27, 'Наливной (продукты)'),

(1.42, 'Насыпной'),

(1.88, 'Штучный');/\*update ego v factore\*/

/\*через csv 10000 водителей V\*/

load data infile 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\drivers.csv' INTO TABLE driver FIELDS TERMINATED BY ',' IGNORE 1 LINES (full\_name\_driver);

insert into category (name\_category)

values

('С1E'),

('D1E'),

('DE'),

('C'),

('E');/\*udalit etu categoriu\*/

insert into type\_customer (name\_type)

values

('ИП'),

('Физ.лицо'),/\*zdelat update\*/

('Компания'),

('Несуществующий тип');/\*udalit tip\*/

/\*csv 10000 цепочек водителя и категории V\*/

load data infile 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\DRIVER\_TO\_CATEGORY.csv' INTO TABLE driver\_to\_category FIELDS TERMINATED BY ',' IGNORE 1 LINES (fk\_id\_driver, fk\_id\_category);

/\*добавили в customer 10000 строк csv V\*/

load data infile 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\customers.csv' INTO TABLE customer FIELDS TERMINATED BY ';' IGNORE 1 LINES (title, phone\_number\_c, fk\_id\_type\_customer);

/\*csv, 10000 грузовиков V\*/

load data infile 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\TRUCK.csv' INTO TABLE truck FIELDS TERMINATED BY ',' IGNORE 1 LINES (id\_truck, name\_truck, number\_sign, factor, volume, fk\_id\_category);

/\*многие ко многим 10000 строк грузовика и типа груза V\*/

load data infile 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\TRUCK\_TO\_TYPE\_CARGO.csv' INTO TABLE truck\_to\_type\_cargo FIELDS TERMINATED BY ',' IGNORE 1 LINES (fk\_id\_truck, fk\_id\_type\_cargo);

/\*в рейсы 10000 строк из csv V\*/

load data infile 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\flights\_and\_ids.csv' INTO TABLE flight FIELDS TERMINATED BY ',' IGNORE 1 LINES (id\_flight, status\_f, director\_full\_name, date\_time\_start, date\_time\_end, number\_f, base\_price, fk\_id\_truck, fk\_id\_worker, fk\_id\_city1, fk\_id\_city2, fk\_id\_driver1, fk\_id\_driver2, fk\_id\_customer);

/\*добавлены 10000 строк грузов V\*/

load data infile 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\cargos.csv' INTO TABLE cargo FIELDS TERMINATED BY ';' IGNORE 1 LINES (volume\_cargo, weight, place\_loading, place\_unloading, fk\_id\_type\_cargo, fk\_id\_flight);

/\*добавлены 10000 строк комментариев к рейсам V\*/

load data infile 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\comments.csv' INTO TABLE comment\_c FIELDS TERMINATED BY ';' IGNORE 1 LINES (text\_c, grade, fk\_id\_customer, fk\_id\_flight);

**1. Индексы.**

Для 100 запросов из Л/Р 4 выбран следующий топ 10 запросов (которые возможно оптимизировать за счёт индексов):

**1)**

/\*5 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.00506580, СТОИМОСТЬ 1120.71\*/

/\*5 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00314000, СТОИМОСТЬ 540.71\*/

/\*Суперзапрос SELECT, DISTINCT, WHERE, AND/OR/NOT, IN, BETWEEN\*/

select status\_f as 'СТАТУС',

number\_f as 'НОМЕР',

base\_price as 'СТОИМОСТЬ БЕЗ КОЭФФИЦИЕНТОВ'

from flight

where ((base\_price between 39000 and 47000)

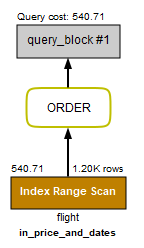
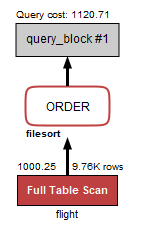
and ((date\_time\_start >= '2022-03-01 00:00'))

and (date\_time\_end <= '2022-05-25 23:00'))

order by base\_price asc;/\*вывести столбцы с номером, статусом и стоимостью рейсов, где ограничения стоимости и промежутки дат начала и конца\*/

create index in\_price\_and\_dates on flight (base\_price);/\*индекс на стоимость(составной в данном случае не нужен, если сделать составной и цену не на первое место, то стоимость только вырастет, а так снизится в 2 раза)\*/

drop index in\_price\_and\_dates on flight;



**2)**

/\*11 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.00510150, СТОИМОСТЬ 1000.25\*/

/\*11 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00478700, СТОИМОСТЬ 414.25\*/

select count(1) as 'Количество отменённых рейсов за апрель 2022 года'

from flight

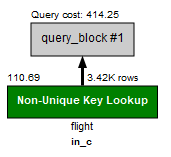
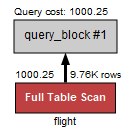
where ((status\_f = 'Canceled')

and (date\_time\_start >= '2022-04-01 00:00')

and (date\_time\_end <= '2022-04-30 23:59'));/\*1. Показать количество отмененных рейсов за последний месяц\*/

create index in\_canceled\_flights on flight (status\_f);/\*индекс на статус, только этот индекс оптимизирует запрос\*/

drop index in\_canceled\_flights on flight;



**3)**

/\*3 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.00645680, СТОИМОСТЬ 1000.25\*/

/\*3 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00705830, СТОИМОСТЬ 407.95\*/

SELECT

number\_f, fk\_id\_driver1, fk\_id\_driver2, status\_f

FROM

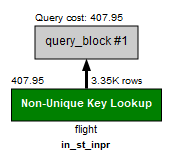
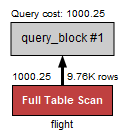
flight

WHERE

(status\_f IN ('In progress'));/\*вывод столбцов строк рейсов, которые в процессе\*/

create index in\_status\_in\_progress\_flights on flight (status\_f);/\*индекс только на статус, т.к выборка различна только в статусах, оптимизировали стоимость запроса с индексом\*/

drop index in\_status\_in\_progress\_flights on flight;



**4)**

/\*12 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.01365150, СТОИМОСТЬ 1000.25\*/

/\*12 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00684820, СТОИМОСТЬ 396.05\*/

SELECT

COUNT(fk\_id\_truck), fk\_id\_truck

FROM

flight

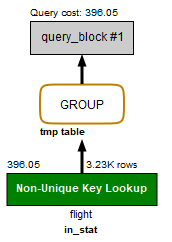
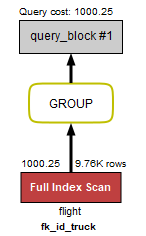
WHERE

(status\_f = 'Done')

GROUP BY fk\_id\_truck;/\*вывод того, сколько каждый грузовик совершил рейсов, считаются только выполненные\*/

create index in\_status\_f on flight (status\_f);/\*спад скорости есть, но индексы не работают на запросах с like\*/

drop index in\_status\_f on flight;



**5)**

/\*1 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.00398, СТОИМОСТЬ 1000.25\*/

/\*1 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00164, СТОИМОСТЬ 249.11\*/

select base\_price as 'Базовая стоимость',

date\_time\_start as 'Начало рейса',

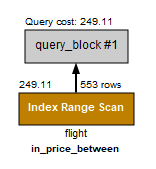
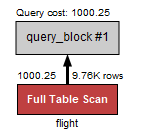
date\_time\_end as 'Конец рейса'

from flight

where (base\_price between 33000 and 37000);/\*вывести столбцы рейсов с датами и ценой, у которых цена от 33000 до 37000, могут быть рейсы, равные левой или правой границе\*/

create index in\_price\_between on flight (base\_price);/\*индекс по 1 полю цены, первый индекс, где стоимость запроса заметно меньше в цифрах(на глаз неощутимо) и сканируется гораздо меньше строк\*/

drop index in\_price\_between on flight;



**6)**

/\*2 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.00028040, СТОИМОСТЬ 1000.25\*/

/\*2 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00033430, СТОИМОСТЬ 5.15\*/

SELECT DISTINCT

status\_f AS 'Статус рейса'

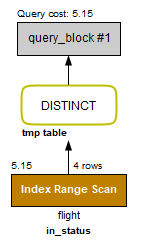
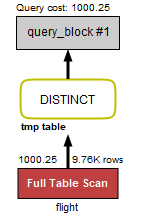
FROM

flight

LIMIT 2;/\*вывести статусы из всех рейсов (только 2 и без повторений)\*/

create index in\_status on flight (status\_f);/\*индекс на статус (уменьшает стоимость запроса, в плане показано)\*/

drop index in\_status on flight;



**7)**

/\*7 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.00397680, СТОИМОСТЬ 1559.97\*/

/\*7 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00190120, СТОИМОСТЬ 1.59\*/

SELECT

\*

FROM

flight

WHERE

fk\_id\_worker = ANY (SELECT

id\_worker

FROM

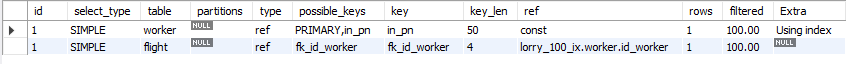
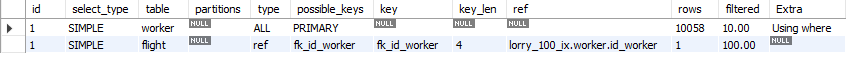
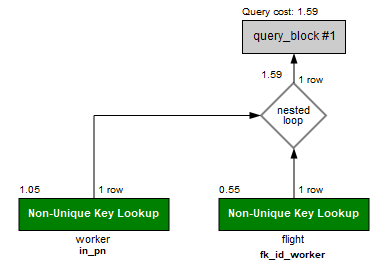
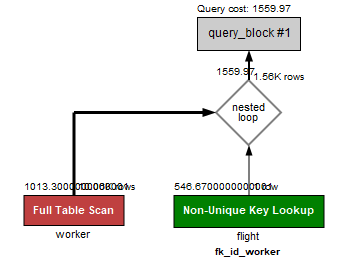
worker

WHERE

phone\_number = '89613338345');/\*вывести рейсы, у которых работник имеет указанный номер\*/

create index in\_phone\_number\_worker on worker (phone\_number);/\*индекс очень сильно снизил стоимость, время в 2 раза меньше, но это доли секунды\*/

drop index in\_phone\_number\_worker on worker;



**8)**

/\*9 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.00574690, СТОИМОСТЬ 1000.25\*/

/\*9 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00036030, СТОИМОСТЬ 0.71\*/

SELECT

\*

FROM

flight

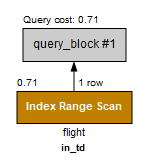
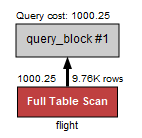
WHERE

((date\_time\_start >= '2022-04-27 00:00')

AND (date\_time\_start <= '2022-04-27 23:59'));/\*1. Показать расписание на текущие сутки. (27.04.2022)\*/

create index in\_1\_day on flight (date\_time\_start);

drop index in\_1\_day on flight;



**9)**

/\*10 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.00401060, СТОИМОСТЬ 1000.25\*/

/\*10 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00036080, СТОИМОСТЬ 0.35\*/

select sum(base\_price) as 'Общая выручка с рейсов за 1 год'

from flight

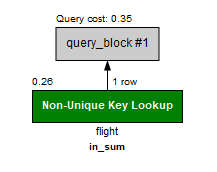
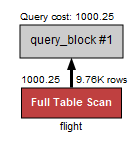
where ((date\_time\_start >= '2022-01-01 00:00')

and (date\_time\_end <= '2022-12-31 23:59')

and (status\_f = 'Выполнен'));/\*2. Показать общую выручку с рейсов за 1 год. Без учёта коэффициентов типов груза и грузовиков\*/

create index in\_sum\_done on flight (status\_f);/\*индекс на статус, только этот индекс оптимизирует запрос\*/

drop index in\_sum\_done on flight;



**10)**

/\*4 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.00565880, СТОИМОСТЬ 1000.25\*/

/\*4 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00032190, СТОИМОСТЬ 0.35\*/

SELECT

\*

FROM

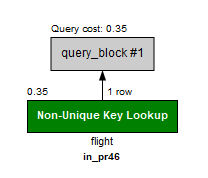
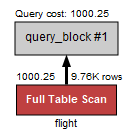
flight

WHERE

(base\_price IN (46000));/\*выводим только те рейсы, где базовая стоимость 46000\*/

create index in\_price\_46000 on flight (base\_price);/\*индекс на цену, обычный(очень большой выигрыш в стоимости запроса, но время не меняется почти)\*/

drop index in\_price\_46000 on flight;



Запросы, заслуживающие отдельного внимания:

**1)**

/\*6 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.03272300, СТОИМОСТЬ 1024.35(ИТОГОВАЯ), ПОДЗАПРОСА - 0.65\*/

/\*6 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.02559830, СТОИМОСТЬ 1024.35(ИТОГОВАЯ), ПОДЗАПРОСА - 0.35(НЕЗНАЧИТЕЛЬНО)\*/

SELECT

name\_truck, number\_sign

FROM

truck

WHERE

fk\_id\_category = ALL (SELECT

id\_category

FROM

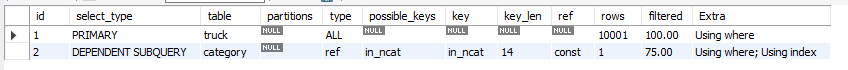
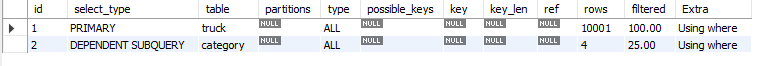
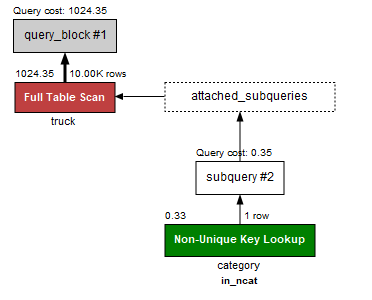
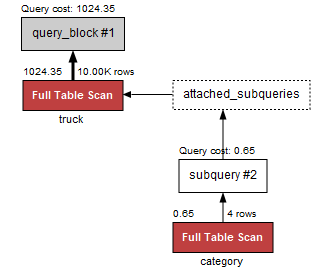
category

WHERE

name\_category = 'С1E');/\*искать номер и грузовик, у которого категория C1E\*/

create index in\_name\_category on category (name\_category);/\*время и конечная стоимость не меняются, но есть различия в explain и стоимость подзапроса мала, но меньше чуть\*/

drop index in\_name\_category on category;



Особенность запроса в том, что несмотря на неизменный показатель конечной стоимости и уменьшенной при этом стоимости подзапроса, время выполнение стало меньше.

**2)**

/\*13 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.01367010, СТОИМОСТЬ 1024.35\*/

/\*13 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00022500, СТОИМОСТЬ -(ПИШЕТ, ЧТО ВЫБРАНЫ ОПТИМИЗИРОВАННЫЕ ТАБЛИЦЫ), ОЧЕНЬ ОПТИМИЗИРОВАНО\*/

SELECT

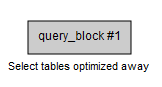
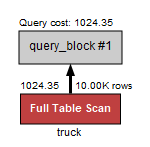
MIN(volume)

FROM

truck;/\*вывод наименьшего объёма у грузовиков\*/

create index in\_volume\_truck on truck (volume);/\*оптимизация появилась, но нельзя сказать по плану, насколько улучшилась (по времени в 10 раз)\*/

drop index in\_volume\_truck on truck;



Особенность: запрос хорошо оптимизировался, однако мы не можем узнать итоговую стоимость запроса (но время выполнения намного меньше).

**3)**

/\*8 ДО ИНДЕКСА: ВРЕМЯ = 0:00:0.01216680, СТОИМОСТЬ 1000.25\*/

/\*8 ПОСЛЕ ИНДЕКСА: ВРЕМЯ = 0:00:0.00908040, СТОИМОСТЬ 1028.06(СТОИМОСТЬ ЧУТЬ ВЫШЕ, НО ВРЕМЯ МЕНЬШЕ)\*/

SELECT

\*

FROM

flight

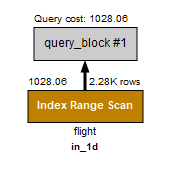
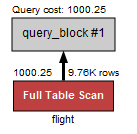
WHERE

((date\_time\_start >= '2022-04-01 00:00')

AND (date\_time\_start <= '2022-04-03 23:59'));/\*2. Показать расписание на ближайшие 3 дня.\*/

create index in\_day\_flight on flight (date\_time\_start);/\*уменьшение стоимости запроса\*/

drop index in\_day\_flight on flight;



Конечная стоимость запроса увеличилась, но при этом время выполнения уменьшилось (в несколько раз).

**2. Процедуры.**

**1)**

/\*1. Процедура без параметров, просто вывести строки с именами работников на A\*/

DELIMITER $$

CREATE PROCEDURE pr\_all\_workers\_like\_A ()/\*нет параметров\*/

BEGIN

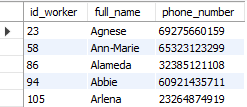
SELECT \* FROM worker where full\_name like 'A%';

END $$

DELIMITER ;

call pr\_all\_workers\_like\_A;

drop procedure pr\_all\_workers\_like\_A;



**2)**

/\*2. Процедура с 1 входным параметром in, вывести поля с id, директором и датами, где заданная в процедура стоимость рейса < базовой из таблицы\*/

delimiter $$

create procedure pr\_prices\_more (in price int)/\*1 параметр\*/

begin

select id\_flight, director\_full\_name, date\_time\_start, date\_time\_end, base\_price

from flight

where price < base\_price

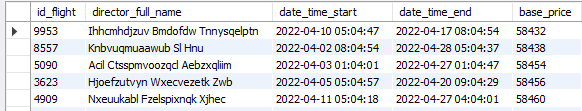
order by base\_price asc;

end $$

delimiter ;

call pr\_prices\_more(58424);/\*вывести рейсы, стоимость которых больше указанной\*/

drop procedure pr\_prices\_more;



**3)**

/\*3. Процедура с несколькими(3) входными in, вывести строку груза, где вес меньше указанного и места погрузки и выгрузки равны заданным в процедуре\*/

delimiter $$

create procedure pr\_search\_comment (in weight\_pr int, in place1 varchar(50), in place2 varchar(50))/\*3 параметра\*/

begin

select \* from cargo

where weight\_pr > weight

and place1 = place\_loading

and place2 = place\_unloading;

end $$

delimiter ;

call pr\_search\_comment(5000, 'LLk', 'VVyJwd');

drop procedure pr\_search\_comment;



**4)**

delimiter $$

create procedure pr\_avg\_grade\_drivers ()/\*без параметров\*/

begin

SELECT

AVG(ROUND(comment\_c.grade, 3)), d.full\_name\_driver

FROM comment\_c

LEFT JOIN

flight ON (comment\_c.fk\_id\_flight = flight.id\_flight)

LEFT JOIN

driver d ON d.id\_driver = flight.fk\_id\_driver1 OR d.id\_driver = flight.fk\_id\_driver2

GROUP BY d.id\_driver

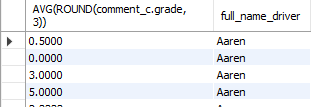
ORDER BY d.full\_name\_driver;

end $$

delimiter ;

call pr\_avg\_grade\_drivers;

drop procedure pr\_avg\_grade\_drivers;



**5)**

/\*5. Процедура на 4 параметра, вывести столбцы с номером, статусом и стоимостью рейсов, где ограничения стоимости и промежутки дат начала и конца\*/

delimiter $$

create procedure pr\_flights\_With\_4\_properties (price1 int, price2 int, date\_time1 datetime, date\_time2 datetime)/\*4 параметра как ограничения\*/

begin

select status\_f as 'СТАТУС',

number\_f as 'НОМЕР',

base\_price as 'СТОИМОСТЬ БЕЗ КОЭФФИЦИЕНТОВ'

from flight

where ((base\_price between price1 and price2)

and ((date\_time\_start >= date\_time1))

and (date\_time\_end <= date\_time2))

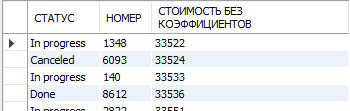
order by base\_price asc;

end $$

delimiter ;

call pr\_flights\_With\_4\_properties(33520, 63040, '2022-03-01 00:00', '2022-05-25 23:00');

drop procedure pr\_flights\_With\_4\_properties;



**6)**

/\*6. \*/

delimiter $$

create procedure pr\_statused\_flights (in status\_f\_pr varchar(45), out outer\_var int)/\*1 вх, 1 вых\*/

begin

select count(1) as 'Количество отменённых рейсов за апрель 2022 года'

into outer\_var

from flight

where ((status\_f\_pr = status\_f)

and (date\_time\_start >= '2022-04-01 00:00')

and (date\_time\_end <= '2022-04-30 23:59'));/\*1. Показать количество отмененных рейсов за последний месяц\*/

end $$

delimiter ;

call pr\_statused\_flights ('In progress', @t1);/\*результат в этой временной переменной\*/

select @t1;/\*вывод того, что получила процедура в данную переменную\*/

drop procedure pr\_statused\_flights;



**3. Функции.**

**1)**

/\*1. Пробная функция\*/

DELIMITER $$

CREATE FUNCTION FunctCalc (starting\_value INT)

RETURNS INT

DETERMINISTIC

BEGIN

DECLARE income INT;

SET income = 0;

label1: WHILE income <= 10 DO

SET income = income + starting\_value;

END WHILE label1;

RETURN income;

END $$

DELIMITER ;

SELECT FunctCalc (1);

drop function FunctCalc;/\*в результате получим на 10 большее число, чем указанное в параметре\*/



**2)**

/\*2.\*/

delimiter $$

create function func\_min\_volume\_truck (volume\_func double)/\*1 вх.параметр\*/

returns double

deterministic

begin

declare out\_var double default 0;

SELECT

MIN(volume)

into out\_var

FROM

truck

where volume\_func < volume;/\*вывод наименьшего объёма у грузовиков, начиная с указанного\*/

RETURN out\_var;

end $$

delimiter ;

select func\_min\_volume\_truck(60);

drop function func\_min\_volume\_truck;



**3)**

/\*3.\*/

delimiter $$

create function func\_sum\_prices (date1 datetime, date2 datetime, status\_fl\_func varchar(25))/\*3 вх.параметра\*/

returns int

deterministic

begin

declare summ double default 0;

select sum(base\_price) as 'Общая выручка с рейсов за 1 год'

into summ

from flight

where ((date\_time\_start >= date1)

and (date\_time\_end <= date2)

and (status\_f = status\_fl\_func));/\*2. Показать общую выручку с рейсов за 1 год. Без учёта коэффициентов типов груза и грузовиков\*/

return summ;

end $$

delimiter ;

select func\_sum\_prices ('2022-01-01 00:00', '2022-12-31 23:59', 'Done');

drop function func\_sum\_prices;



**4)**

/\*4.\*/

delimiter $$

create function func\_flights\_6\_properties (st varchar(25), price1 int, price2 int, num int, date1 datetime, date2 datetime)/\*6 вх.параметров\*/

returns int

deterministic

begin

declare count\_flights double default 0;

select count(\*)

into count\_flights

from flight

where status\_f = st

and base\_price between price1 and price2

and number\_f > num

and date\_time\_start > date1

and date\_time\_end < date2;/\*кол-во рейсов со определённым статусом, ценой, номером > указанного и датами начала и конца\*/

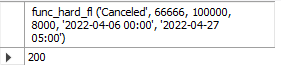
return count\_flights;

end $$

delimiter ;

select func\_flights\_6\_properties ('Canceled', 66666, 100000, 8000, '2022-04-06 00:00', '2022-04-27 05:00');

drop function func\_flights\_6\_properties;



**5)**

/\*5.\*/

delimiter $$

create function func\_avg\_factor (v float, categ tinyint)/\*2 параметра\*/

returns int

deterministic

begin

declare coun double default 0;

select avg(factor)

into coun

from truck

where volume > v

and fk\_id\_category = categ;/\*средний коэффициент грузовика с объёмом более указанного и с определённой категорией\*/

return `coun`;

end $$

delimiter ;

select func\_avg\_factor(50.00, 3);/\*округляет до целого значения\*/

drop function func\_avg\_factor;



**4. Представления.**

**1)**

/\*1. \*/

create view v\_rating\_drivers /\*стоимость стала выше!\*/

as

/\*3) Для каждого водителя вывести среднюю оценку из комментариев к рейсам, где они участвовали. Where нет, просто присоединение по ключам.\*/

SELECT

AVG(ROUND(comment\_c.grade, 3)), d.full\_name\_driver

FROM comment\_c

LEFT JOIN

flight ON (comment\_c.fk\_id\_flight = flight.id\_flight)

LEFT JOIN

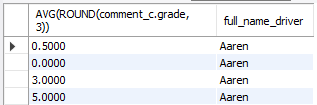
driver d ON d.id\_driver = flight.fk\_id\_driver1 OR d.id\_driver = flight.fk\_id\_driver2

GROUP BY d.id\_driver

ORDER BY d.full\_name\_driver;

select \* from v\_rating\_drivers;

drop view v\_rating\_drivers;



**2)**

/\*2. \*/

create view v\_sum\_prices/\*очень сильно снизилась стоимость запроса с 1000.25 до 1!!! По времени поменялось чуть-чуть\*/

as

SELECT

SUM(base\_price)

FROM

flight;/\*вывод суммы всей выручки за всё время рейсов\*/

select \* from v\_sum\_prices;

drop view v\_sum\_prices;



**3)**

/\*3. \*/

create view v\_cargo\_with\_weight /\*стоимость запроса осталась неизменной(на плане)\*/

as

SELECT

\*

FROM

cargo

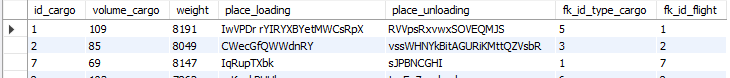
WHERE

((weight BETWEEN 5000 AND 6700)

OR (weight BETWEEN 7400 AND 9040));/\*вывести строки с грузами, вес которых принадлежит одному из промежутков\*/

select \* from v\_cargo\_with\_weight;

drop view v\_cargo\_with\_weight;



**4)**

/\*4. \*/

create view v\_top3\_type\_cargo

as

/\*ТОП 3 типов груза за последние 3 месяца и их кол-во (1 - Насыпной (4 раза), 2 - Наливной (2 раза) -> пример.\*/

SELECT

type\_cargo.name\_type\_cargo, COUNT(fk\_id\_flight) AS raz

FROM

cargo

JOIN

flight ON cargo.fk\_id\_flight = flight.id\_flight

RIGHT JOIN

type\_cargo ON cargo.fk\_id\_type\_cargo = type\_cargo.id\_type\_cargo

WHERE

flight.date\_time\_start IS NULL

OR (flight.date\_time\_start >= '2020-03-01 00:00'

AND flight.date\_time\_end <= '2022-05-15 00:00')

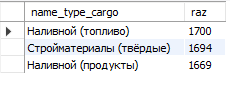
GROUP BY cargo.fk\_id\_type\_cargo

ORDER BY raz DESC

limit 3;

select \* from v\_top3\_type\_cargo;/\*снова сильное снижение стоимости запроса через представление\*/

drop view v\_top3\_type\_cargo;



**5)**

/\*5. \*/

create view v\_some\_trucks/\*представление не меняет стоимоть запроса(это делает только индекс)\*/

as

select

name\_truck,

number\_sign

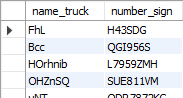
from truck

where factor > 1.89

and volume between 55 and 60;/\*вывести номер и название грузовика, у которого опроеделённый объём и коэффициент\*/

select \* from v\_some\_trucks;

drop view v\_some\_trucks;



**4. Триггеры.**

/\*1. Триггер на удаление строки города с названием менее 3 символов\*/

DELIMITER $$

CREATE TRIGGER tr\_on\_delete\_wrong\_city

after insert

ON city

FOR EACH ROW

delete FROM city

WHERE length(name\_city) < 3 $$

DELIMITER ;

drop trigger tr\_on\_delete\_wrong\_city;

/\*2. Триггер на обновление строки грузовика с коэффициентом менее 1\*/

DELIMITER $$

CREATE TRIGGER tr\_on\_update\_wrong\_trucks

after insert

ON truck

FOR EACH ROW

update truck set factor = 1.1

WHERE factor < 1 $$

DELIMITER ;

drop trigger tr\_on\_update\_wrong\_trucks;

Модификация:

1) Запрос на 6 секунд и запрос где запрос на ТОП 3 типов груза – план (по возможности, оптимизировать)

/\*6.362 sec -> 6.527 sec, запрос стал дольше\*/

/\*3) Для каждого водителя вывести среднюю оценку из комментариев к рейсам, где они участвовали. Where нет, просто присоединение по ключам.\*/

SELECT

AVG(ROUND(comment\_c.grade, 3)), d.full\_name\_driver

FROM comment\_c

LEFT JOIN

flight ON (comment\_c.fk\_id\_flight = flight.id\_flight)

LEFT JOIN

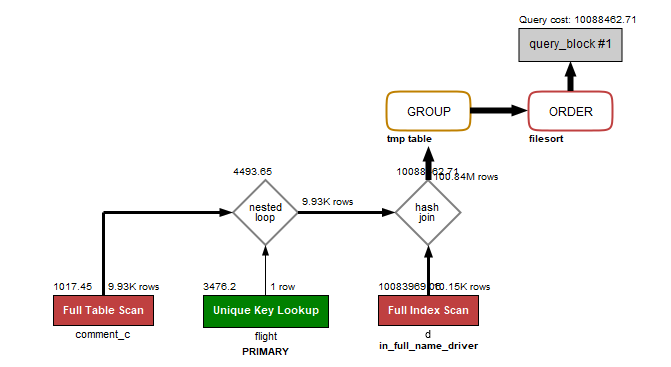
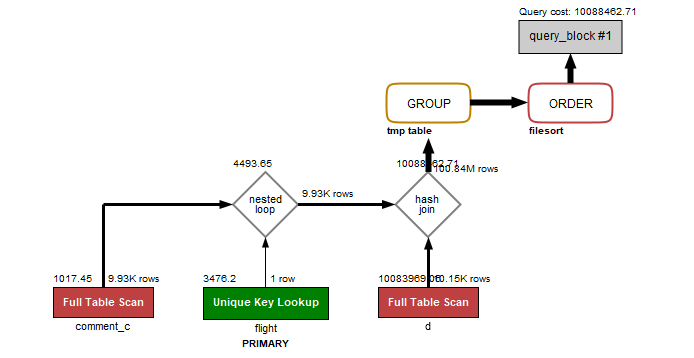
driver d ON d.id\_driver = flight.fk\_id\_driver1 OR d.id\_driver = flight.fk\_id\_driver2

GROUP BY d.id\_driver

ORDER BY d.full\_name\_driver;

create index in\_full\_name\_driver on driver (full\_name\_driver);

drop index in\_full\_name\_driver on driver;



К сожалению, этот запрос нельзя оптимизировать, поскольку он не содержит where. По полю с order by оптимизация также не наблюдается.

Но можно сделать следующий запрос:

select/\*сам запрос модификации(вместо предыдущего) 0.098 сек(время)\*/

avg(avg\_rating), driver.full\_name\_driver

from

(

select

fk\_id\_driver1 as driver\_id, ifnull(avg(comment\_c.grade), 0) as avg\_rating

from flight

left join comment\_c on comment\_c.fk\_id\_flight = flight.id\_flight

group by fk\_id\_driver1

union all

select

fk\_id\_driver2, ifnull(avg(comment\_c.grade), 0)

from flight

left join comment\_c on comment\_c.fk\_id\_flight = flight.id\_flight

group by fk\_id\_driver2

)

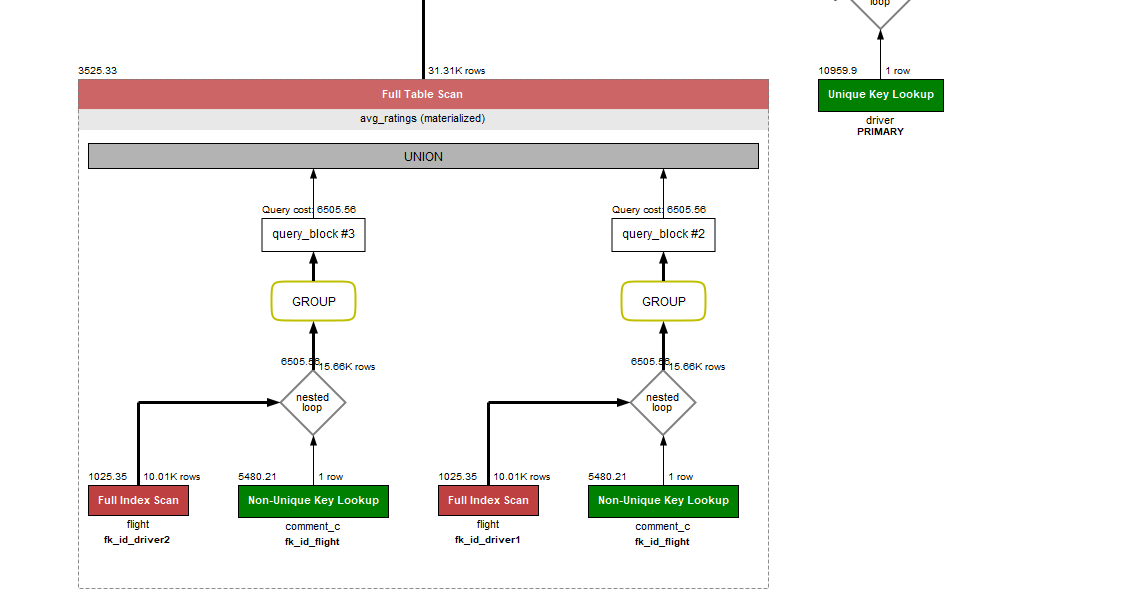
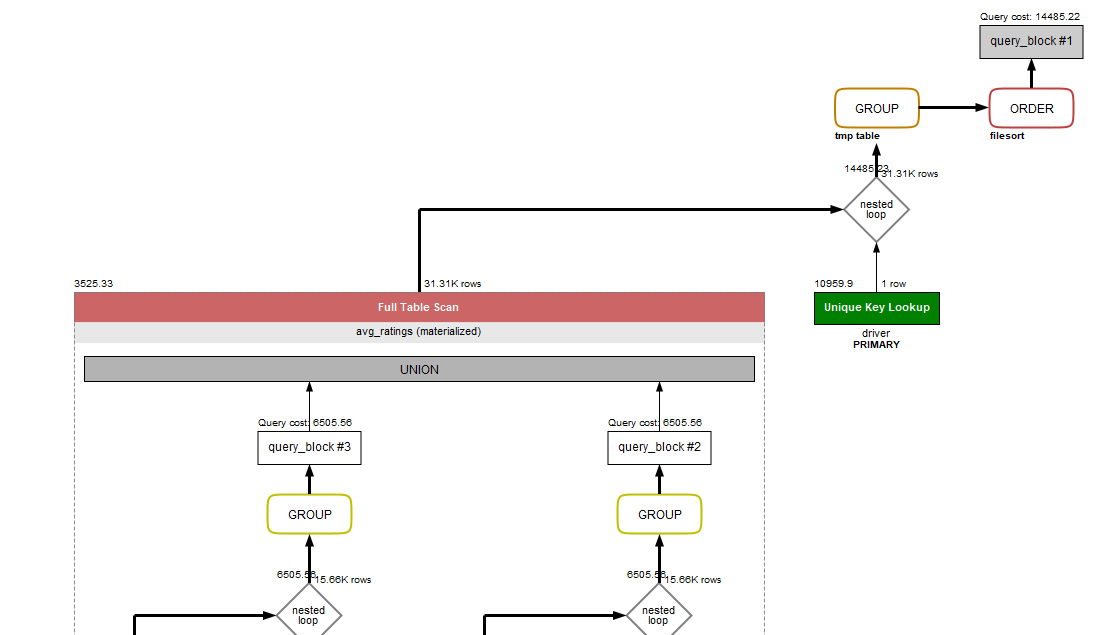
as avg\_ratings

join driver on driver\_id = driver.id\_driver

group by driver\_id

order by avg(avg\_rating) desc, driver.full\_name\_driver asc;

План нового запроса:



Вывод: в join не следует создавать условие or!!!

2) Процедура из 4 частей:

1. Получить id заказчика по входному параметру телефона заказчика процедуры;

2. Получить id рейса по входному параметру номера рейса процедуры;

3. Добавить комментарий по параметрам (предыдущие 2 id и текст с оценкой(inout) как конкретные значения процедуры);

4. Вывести среднюю оценку рейса из всех его комментариев, т.е комментарий добавился и вывести обновлённое среднее из оценки.

delimiter $$

create procedure pr\_from\_four\_parts (in number\_phone varchar(12), in number\_flight int, inout pr\_grade tinyint, in txt text)

begin

declare var\_id\_c int default 0;

declare var\_id\_f int default 0;

/\*1ч\*/

select id\_customer

into var\_id\_c

from customer

where number\_phone = phone\_number\_c;

/\*2ч\*/

select id\_flight

into var\_id\_f

from flight

where number\_flight = number\_f;

/\*3ч\*/

insert into comment\_c (text\_c, grade, fk\_id\_customer, fk\_id\_flight) values

(txt, pr\_grade, var\_id\_c, var\_id\_f);

/\*4ч\*/

select avg(grade)

into pr\_grade

from comment\_c

where fk\_id\_flight = var\_id\_f;

end $$

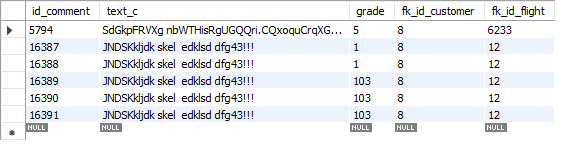
delimiter ;

set @grade = 103;

call pr\_from\_four\_parts ('886838', 12, @grade, 'JNDSKkljdk skel edklsd dfg43!!!');

drop procedure pr\_from\_four\_parts;

select @grade;



3) Представление, в котором выводится вся информация о рейсе, но без id внешних ключей (полная таблица).

create view v\_all\_info\_flight as

SELECT

f.id\_flight,

f.status\_f,

f.director\_full\_name,

f.date\_time\_start,

f.date\_time\_end,

f.number\_f,

f.base\_price,

t.name\_truck,

w.full\_name,

y.name\_city as first\_city,

n.name\_city as second\_city,

i.full\_name\_driver as first\_driver,

v.full\_name\_driver as second\_driver,

c.title

FROM

flight f

LEFT JOIN

truck t ON (f.fk\_id\_truck = t.id\_truck)

LEFT JOIN

worker w ON (f.fk\_id\_worker = w.id\_worker)

LEFT JOIN

city y ON (f.fk\_id\_city1 = y.id\_city)

LEFT JOIN

city n ON (f.fk\_id\_city2 = n.id\_city)

LEFT JOIN

driver i ON (f.fk\_id\_driver1 = i.id\_driver)

LEFT JOIN

driver v ON (f.fk\_id\_driver2 = v.id\_driver)

LEFT JOIN

customer c ON (f.fk\_id\_customer = c.id\_customer);/\*вывести полностью таблицу рейсов, заменив внешние ключи конкретными соответствующими данными (не fk\_id)\*/

select \* from v\_all\_info\_flight;

drop view v\_all\_info\_flight;

