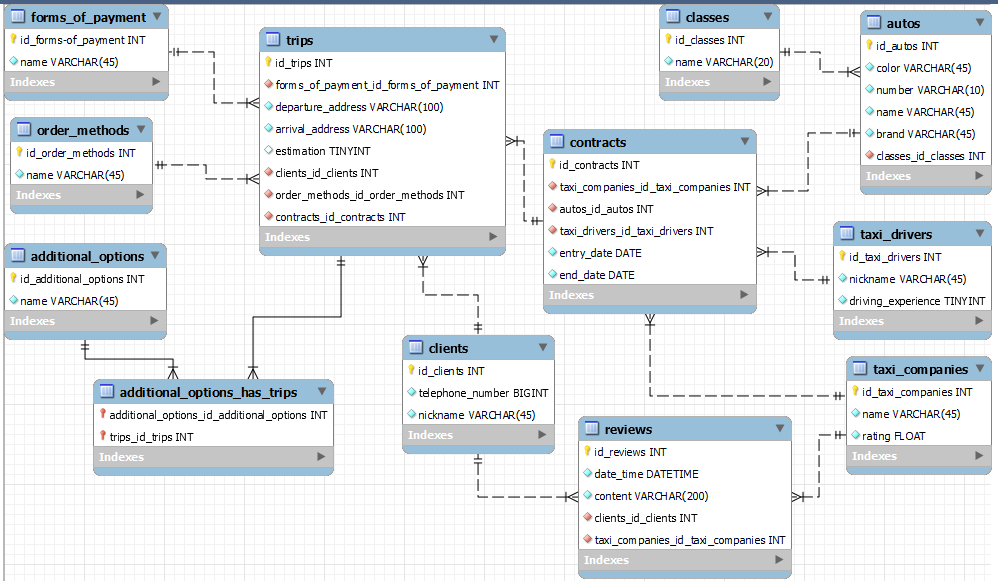
|  |  |  |
| --- | --- | --- |
| Лабораторная работа № 5  «Такси» | Ф.И.О. | Самороков Н. Н. |
| Группа | ИВТ-263 |
| Преподаватель | Соколов А. А. |
| Дата сдачи |  |

**Реляционная схема по теме «Такси»:**



**Код загрузки данных:**

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\autos.csv' INTO TABLE autos -- добавить 10к записей в авто

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_autos, color, number, name, brand, classes\_id\_classes);

SELECT \* FROM autos;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\additional\_opt.csv' INTO TABLE additional\_options -- добавить 10к записей в доп. опции

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_additional\_options, name);

SELECT \* FROM additional\_options;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\classes.csv' INTO TABLE classes -- добавить 10к записей в классы

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_classes, name);

SELECT \* FROM classes;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\forms\_of\_payment.csv' INTO TABLE forms\_of\_payment -- добавить 10к записей в формы оплаты

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_forms\_of\_payment, name);

SELECT \* FROM forms\_of\_payment;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\clients.csv' INTO TABLE clients -- добавить 10к записей в клиентов

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_clients, telephone\_number, nickname);

SELECT \* FROM clients;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\order\_methods.csv' INTO TABLE order\_methods -- добавить 10к записей в способ заказа

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_order\_methods, name);

SELECT \* FROM order\_methods;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\taxi\_drivers.csv' INTO TABLE taxi\_drivers -- добавить 10к записей в таксистов

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_taxi\_drivers, nickname, driving\_experience);

SELECT \* FROM taxi\_drivers;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\taxi\_companies.csv' INTO TABLE taxi\_companies -- добавить 10к записей в такси компании

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_taxi\_companies, name, rating);

SELECT \* FROM taxi\_companies;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\reviews.csv' INTO TABLE reviews -- добавить 10к записей в отзывы

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_reviews, date\_time, content, clients\_id\_clients, taxi\_companies\_id\_taxi\_companies);

SELECT \* FROM reviews;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\contracts.csv' INTO TABLE contracts -- добавить 10к записей в контракты

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_contracts, taxi\_companies\_id\_taxi\_companies, autos\_id\_autos, taxi\_drivers\_id\_taxi\_drivers, entry\_date, end\_date);

SELECT \* FROM contracts;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\trips.csv' INTO TABLE trips -- добавить 10к записей в поездки

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(id\_trips, forms\_of\_payment\_id\_forms\_of\_payment, departure\_address, arrival\_address, estimation, clients\_id\_clients, order\_methods\_id\_order\_methods, contracts\_id\_contracts);

SELECT \* FROM trips;

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\addit\_to\_trips.csv' INTO TABLE additional\_options\_has\_trips -- добавить 10к записей в связь доп. опции и поездки

FIELDS TERMINATED BY ','

ENCLOSED BY ','

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES

(additional\_options\_id\_additional\_options, trips\_id\_trips);

SELECT \* FROM additional\_options\_has\_trips;

**Результаты загрузки данных** (на примере некоторых таблиц):

Таблица «контракты»

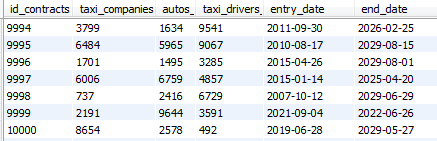


Таблица «клиенты»



Таблица «автомобили»

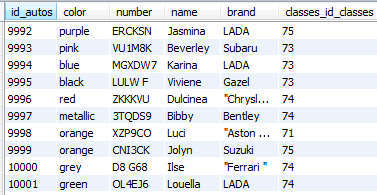


Таблица «таксисты»

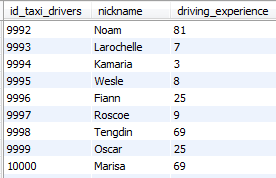
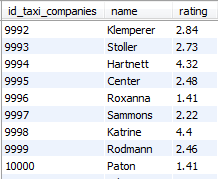


Таблица «такси компании»

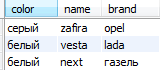


**«Долгие или частые» запросы:**

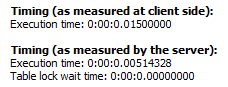
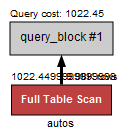
**1)**

select color, name, brand from autos

where color in ("белый", "серый"); -- отобразить все авто белого и серого цвета



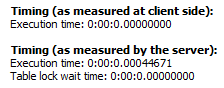
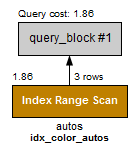
Получаем полный перебор таблицы autos:



-- создадим индекс по цвету, т.к. поиск осуществляется по нему

create index idx\_color\_autos on autos (color);

Результат оптимизированного запроса:



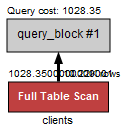
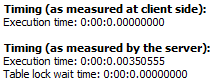
**2)**

select telephone\_number, nickname from clients

where telephone\_number = "79021527509"; -- отобразить кдиента с номером телефона "79021527509"



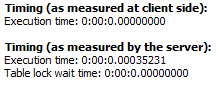
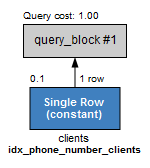
Получаем полный перебор таблицы clients:

-- создадим УНИКАЛЬНЫЙ индекс по номеру телефона, т.к. он не повторяется

create unique index idx\_phone\_number\_clients on clients (telephone\_number);

Результат оптимизированного запроса:



**3)**

select trips.departure\_address, trips.arrival\_address, forms\_of\_payment.name as forms\_of\_payment, order\_methods.name as order\_methods,

clients.nickname, estimation, taxi\_companies.name as taxi\_companies, taxi\_drivers.nickname as taxi\_drivers, autos.number, autos.color, autos.name as autos\_name, autos.brand

from trips

join forms\_of\_payment on forms\_of\_payment.id\_forms\_of\_payment = trips.forms\_of\_payment\_id\_forms\_of\_payment

join order\_methods on order\_methods.id\_order\_methods = trips.order\_methods\_id\_order\_methods

join clients on clients.id\_clients = trips.clients\_id\_clients

join contracts on contracts.id\_contracts = trips.contracts\_id\_contracts

join taxi\_companies on taxi\_companies.id\_taxi\_companies = contracts.taxi\_companies\_id\_taxi\_companies

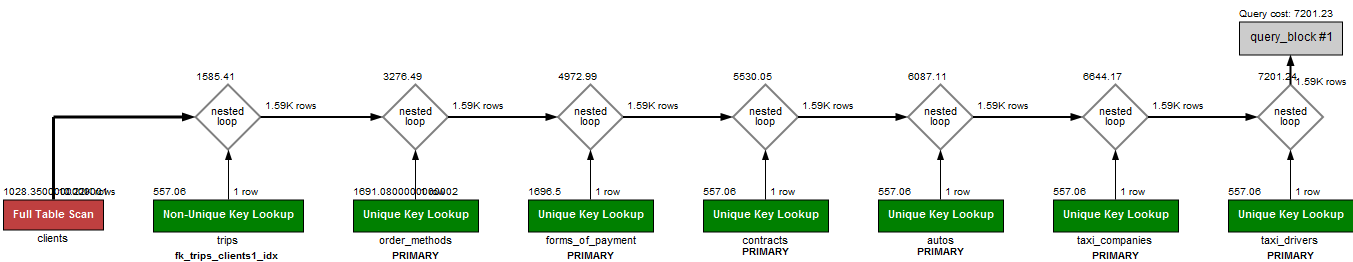
join taxi\_drivers on taxi\_drivers.id\_taxi\_drivers = contracts.taxi\_drivers\_id\_taxi\_drivers

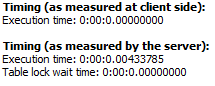
join autos on autos.id\_autos = contracts.autos\_id\_autos

where clients.nickname = "QuanRy"; -- вывести всю информацию о поздке для клиента QuanRy (1)



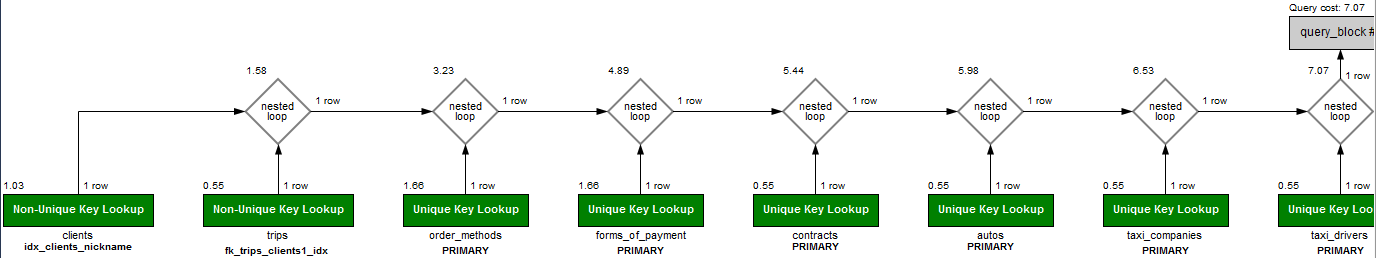
Получаем полный перебор таблицы clients:

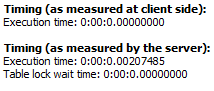




-- создадим индекс по никнейму в таблице клиент, потому что именно вней перебираются все никнеймы

create index idx\_clients\_nickname on clients (nickname);





**4)**

select taxi\_companies.name, autos.number, autos.color, autos.name, autos.brand, classes.name from contracts

join taxi\_companies on taxi\_companies.id\_taxi\_companies = contracts.taxi\_companies\_id\_taxi\_companies

join autos on autos.id\_autos = contracts.autos\_id\_autos

join classes on classes.id\_classes = autos.classes\_id\_classes

where classes.name = "эконом"

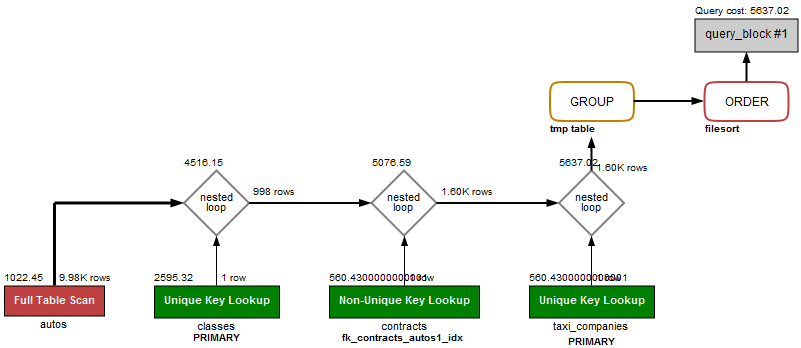
group by color

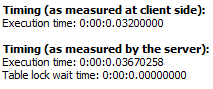
order by autos.name desc

limit 5; -- вывести автомобили, такси компании, в которых они находятся, их класс, который = "эконом", вывести максимум 5 штук



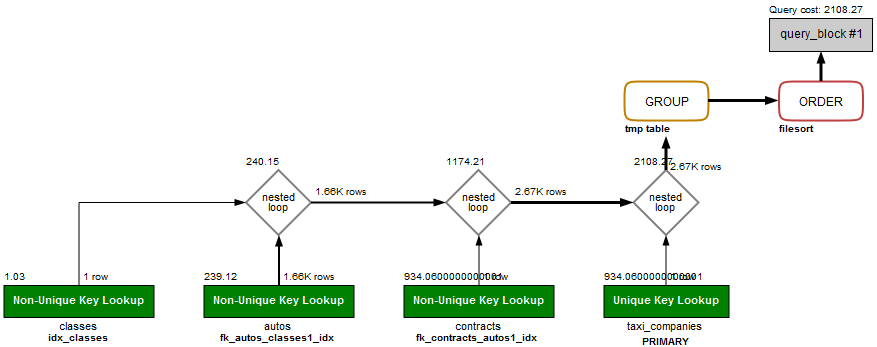
Получаем полный перебор таблицы:

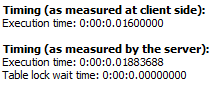




-- создадим индекс по названиям классов, т.к. по ним идет проверка

create index idx\_classes on classes (name);





**5)**

select trips.departure\_address, trips.arrival\_address, forms\_of\_payment.name, order\_methods.name

from trips

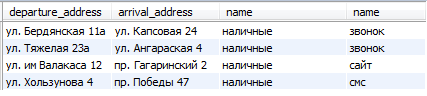
join forms\_of\_payment on forms\_of\_payment.id\_forms\_of\_payment = trips.forms\_of\_payment\_id\_forms\_of\_payment

join order\_methods on order\_methods.id\_order\_methods = trips.order\_methods\_id\_order\_methods

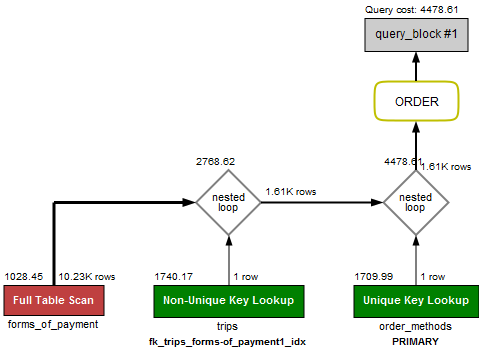
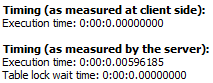
where forms\_of\_payment.name = "наличные"

order by forms\_of\_payment.name

limit 5; -- отобразить адрес назначения, адрес отбытия, тип оплаты - "наличные" вывести максимум 5 штук

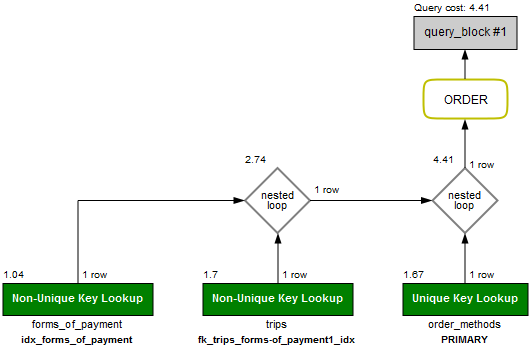
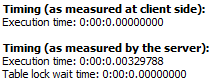


Получаем полный перебор таблицы forms\_of\_payment:

-- создадим индекс по типу оплаты, т.к. поиск осуществляется по нему

create index idx\_forms\_of\_payment on forms\_of\_payment (name);

**Процедуры**

**1)**

-- процедура без параметров (возвращающая марку, название, цвет, номер авто белого цвета)

DELIMITER $$

CREATE PROCEDURE pr\_auto\_white()

BEGIN

select name, brand, color, number from autos

where color = "белый";

END $$

DELIMITER ;

call pr\_auto\_white; -- вызов процедуры для отображения белых авто



**2)**

-- процедура с одним входным параметром - IN (возвращающая никнейм и стаж вождения таксиста, у которого стаж больше входного значения)

DELIMITER $$

CREATE PROCEDURE pr\_driving\_exp\_more (in in\_driving\_experience int)

BEGIN

SELECT nickname, driving\_experience FROM taxi\_drivers

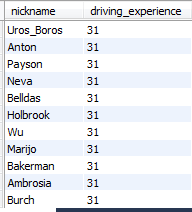
WHERE driving\_experience > in\_driving\_experience

ORDER BY driving\_experience ASC;

END $$

DELIMITER ;

CALL pr\_driving\_exp\_more (30); -- вызов процедуры для отображения более опытных таксистов по стажу входному



**3)**

-- процедура с одним входным параметром - IN (возвращающая никнейм и стаж вождения таксиста, если введен не "клиент", если наоборот, то вывести никнейм и телефон клиента)

DELIMITER $$

CREATE PROCEDURE pr\_client\_or\_driver (in in\_name varchar(45))

BEGIN

IF (in\_name = "клиент") THEN

SELECT nickname, telephone\_number FROM clients;

ELSE

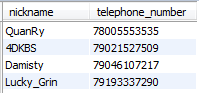
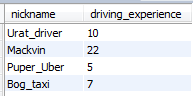
SELECT nickname, driving\_experience FROM taxi\_drivers;

END IF;

END $$

DELIMITER ;

CALL pr\_client\_or\_driver ("клиент"); -- вызов процедуры для отображения клиентов или таксистов

Введен «клиент»: Введено не «клиент»: 

**4)**

-- процедура с одним входным параметром - IN и join (возвращает информацию о контракте, стаж вождения водителя которого равен введенному)

DELIMITER $$

CREATE PROCEDURE pr\_driv\_exp\_ (in in\_driv\_exp int)

BEGIN

select taxi\_companies.name as taxi\_company, taxi\_drivers.nickname, taxi\_drivers.driving\_experience, autos.number, autos.color, autos.name, autos.brand, entry\_date, end\_date from contracts

join taxi\_companies on taxi\_companies.id\_taxi\_companies = contracts.taxi\_companies\_id\_taxi\_companies

join taxi\_drivers on taxi\_drivers.id\_taxi\_drivers = contracts.taxi\_drivers\_id\_taxi\_drivers

join autos on autos.id\_autos = contracts.autos\_id\_autos

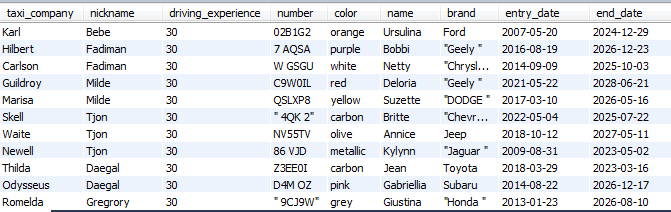
where in\_driv\_exp = taxi\_drivers.driving\_experience

order by taxi\_drivers.driving\_experience desc;

END $$

DELIMITER ;

CALL pr\_driv\_exp\_ (30); -- вызов процедуры для отображения контрактов, с таким же стажом вождения



**Функции**

**1)**

-- Функция для вывода максимального стажа вождения в введенной такси компании

delimiter $$

create function func\_max\_dr\_exp (in\_taxi\_comp varchar(100))

returns int

deterministic

begin

declare s int default 0;

select max(taxi\_drivers.driving\_experience) from contracts

join taxi\_drivers on contracts.taxi\_drivers\_id\_taxi\_drivers = taxi\_drivers.id\_taxi\_drivers

join taxi\_companies on taxi\_companies.id\_taxi\_companies = contracts.taxi\_companies\_id\_taxi\_companies

where taxi\_companies.name = in\_taxi\_comp into s;

return s;

end $$

delimiter ;

select func\_max\_dr\_exp('Yandex'); -- вызов функции для нахождения максимального стажа в такси компании Yandex



**2)**

-- функция для вывода или кол-ва таксистов, у которых стаж в рамках допустимого значения [2;82] и равен введенному

delimiter $$

create function func\_driv\_exp (in\_driv\_exp int)

returns int

deterministic

begin

declare s int default 0;

if (in\_driv\_exp >= 2 and in\_driv\_exp <= 82) then

SELECT count(id\_taxi\_drivers) FROM taxi\_drivers

WHERE driving\_experience = in\_driv\_exp into s;

end if;

return s;

end $$

delimiter ;

select func\_driv\_exp (81); -- вызов функции для вывода кол-ва таксистов с таким стажем



**3)**

-- функция для вывода позиции в рейтинге среди такси компаний

delimiter $$

create function func\_rat\_pos (in\_taxi\_rat int)

returns int

deterministic

begin

declare s int default 0;

if (in\_taxi\_rat > 1 and in\_taxi\_rat <= 5 ) then

SELECT count(id\_taxi\_companies) FROM taxi\_companies

WHERE rating >= in\_taxi\_rat into s;

end if;

return s;

end $$

delimiter ;

select func\_rat\_pos (4.1); -- вызов функции для вывода позиции такси компании в рейтинге



**Представления**

**1)**

-- представление для вывода топ 3 таксиста по сумме отзывов за поездки

create view top\_3\_drivers

as

select taxi\_companies.name as taxi\_company, taxi\_drivers.nickname as taxi\_driver, taxi\_drivers.driving\_experience, autos.number as auto\_number, autos.color, autos.name as auto\_name, autos.brand as auto\_brand, SUM(estimation)

from trips

join contracts on contracts.id\_contracts = trips.contracts\_id\_contracts

join taxi\_companies on taxi\_companies.id\_taxi\_companies = contracts.taxi\_companies\_id\_taxi\_companies

join taxi\_drivers on taxi\_drivers.id\_taxi\_drivers = contracts.taxi\_drivers\_id\_taxi\_drivers

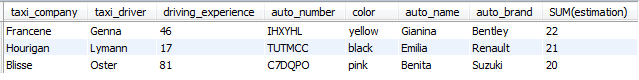
join autos on autos.id\_autos = contracts.autos\_id\_autos

group by contracts\_id\_contracts

order by sum(estimation) desc

limit 3; -- показать топ 3 самых лучших таксистов за оценки по поездкам

select \* from top\_3\_drivers; -- вызов представления



**2)**

-- представление о отзывах, в которых говорится про водителей

create view review\_about\_driver

as

select date\_time, content, clients.nickname, taxi\_companies.name, taxi\_companies.rating from reviews

join clients on clients.id\_clients = reviews.clients\_id\_clients

join taxi\_companies on taxi\_companies.id\_taxi\_companies = reviews.taxi\_companies\_id\_taxi\_companies

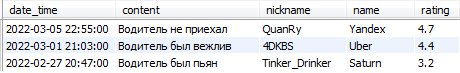
where content like '%водитель%'

group by date\_time

order by rating desc

limit 3; -- отобразить отзывы, в которых говорится про водителей, сгруппировать по дате и вывести в порядке убывания по рейтингу такси компаний, вывести 3 штуки

select \* from review\_about\_driver; -- вызов представления



**3)**

-- представление о топ 10-ти контрактах по дате вхождения (самых новеньких)

create view top\_10\_new\_contr

as

select taxi\_companies.name as taxi\_company, taxi\_drivers.nickname as taxi\_driver, taxi\_drivers.driving\_experience as driving\_experience, autos.number, autos.color, autos.name, autos.brand, entry\_date, end\_date from contracts

join taxi\_companies on taxi\_companies.id\_taxi\_companies = contracts.taxi\_companies\_id\_taxi\_companies

join taxi\_drivers on taxi\_drivers.id\_taxi\_drivers = contracts.taxi\_drivers\_id\_taxi\_drivers

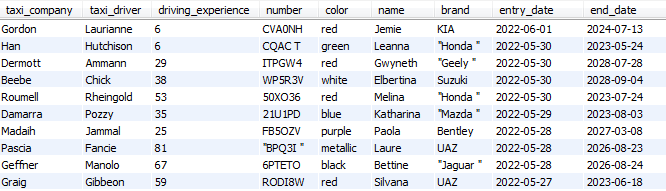
join autos on autos.id\_autos = contracts.autos\_id\_autos

where contracts.end\_date >= "2022-04-29"

order by entry\_date desc

limit 10; -- вывести контракты, которые еще актуальны в убывающем порядке по дате вхождения и максимум 10 штук

select \* from top\_10\_new\_contr; -- вызов представления



**Триггер**

**1)**

-- триггер для замены букв номера автомобиля в нижнем регистре. на буквы в верхнем регистре

Delimiter $$

create trigger auto\_number

before insert on autos

for each row

begin

SET new.number = concat ( upper(left (new.number, 7 ) ), lower (substring(new.number, 7) ) );

end $$

delimiter ;

-- drop trigger auto\_number; -- дропнуть триггер

-- добавляем номер с буквами в нижнем регистре

insert into autos values (null, 'зеленый', 'h274hр', 'SHUMA\_III', 'KIA', '73');

-- выолняем проверку и получаем номер с верхним регистром:

select \* from autos where name = 'SHUMA\_III';



**МОДА**

**1) ИНДЕКС**

select trips.departure\_address, trips.arrival\_address, forms\_of\_payment.name as forms\_of\_payment, order\_methods.name as order\_methods,

clients.nickname, estimation, taxi\_companies.name as taxi\_companies, taxi\_drivers.nickname as taxi\_drivers, autos.number, autos.color, autos.name as autos\_name, autos.brand

from trips

join forms\_of\_payment on forms\_of\_payment.id\_forms\_of\_payment = trips.forms\_of\_payment\_id\_forms\_of\_payment

join order\_methods on order\_methods.id\_order\_methods = trips.order\_methods\_id\_order\_methods

join clients on clients.id\_clients = trips.clients\_id\_clients

join contracts on contracts.id\_contracts = trips.contracts\_id\_contracts

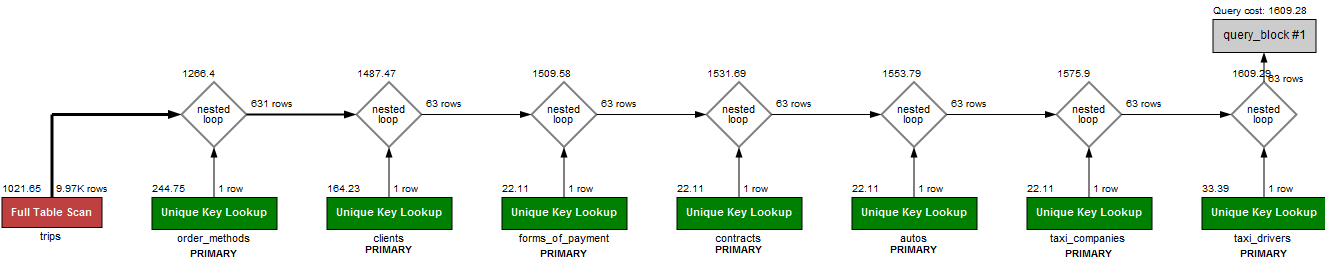
join taxi\_companies on taxi\_companies.id\_taxi\_companies = contracts.taxi\_companies\_id\_taxi\_companies

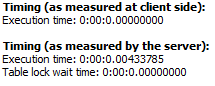
join taxi\_drivers on taxi\_drivers.id\_taxi\_drivers = contracts.taxi\_drivers\_id\_taxi\_drivers

join autos on autos.id\_autos = contracts.autos\_id\_autos

where clients.nickname = "Ana" and (trips.departure\_address or trips.arrival\_address = "Darwin") and trips.estimation >= 4; -- вывести всю информацию о поздке для клиента Ana, по улице "Darwin" с оценкой больше или равной 4



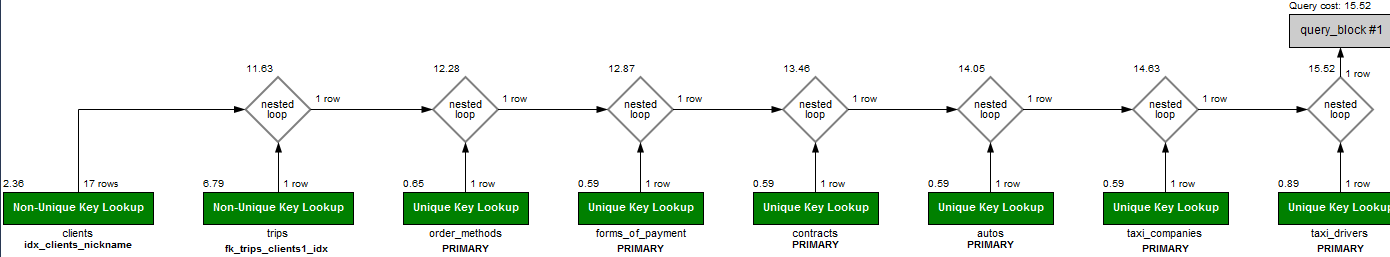


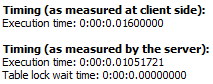


-- создадим индекс по никнейму в таблице клиент, затем создадим индекс по улицам прибытия и отбытия и оценке поездки в поездках, потому что выборка происходит именно по этим полям

create index idx\_clients\_nickname on clients (nickname);

create index idx\_trips\_address on trips (departure\_address, arrival\_address, estimation);





**2) ПРОЦЕДУРА**

-- процедура с одним входным параметром - IN (id\_driver) и двумя выходными OUT (кол-во поездок (num\_trips) и средняя оценка за поездки (avg\_estimation))

DELIMITER $$

CREATE PROCEDURE info\_driver\_trips (in id\_driver int, out num\_trips int, out avg\_estimation float)

BEGIN

select taxi\_drivers.nickname as taxi\_driver, AVG(estimation), count(id\_trips) as num\_trips from trips

join contracts on contracts.id\_contracts = trips.contracts\_id\_contracts

join taxi\_drivers on taxi\_drivers.id\_taxi\_drivers = contracts.taxi\_drivers\_id\_taxi\_drivers

where taxi\_drivers.id\_taxi\_drivers = id\_driver

group by taxi\_drivers.nickname

order by num\_trips desc; -- вывести таксиста, средний рейтинг по оценкам за поездки и кол-во поездок

select (select AVG(estimation) from trips

join contracts on contracts.id\_contracts = trips.contracts\_id\_contracts

join taxi\_drivers on taxi\_drivers.id\_taxi\_drivers = contracts.taxi\_drivers\_id\_taxi\_drivers

where taxi\_drivers.id\_taxi\_drivers = id\_driver) into avg\_estimation;

select (select count(id\_trips) from trips

join contracts on contracts.id\_contracts = trips.contracts\_id\_contracts

join taxi\_drivers on taxi\_drivers.id\_taxi\_drivers = contracts.taxi\_drivers\_id\_taxi\_drivers

where taxi\_drivers.id\_taxi\_drivers = id\_driver) into num\_trips;

END $$

DELIMITER ;

CALL info\_driver\_trips (62, @num\_trips, @avg\_estimation); -- вызов процедуры для отображения более опытных таксистов по стажу входному

При id\_driver = 62 -- > 

При id\_driver = 4116 -- > 

select @avg\_estimation;

При id\_driver = 62 -- > 

При id\_driver = 4116 -- > 

select @num\_trips;

При id\_driver = 62 -- >  При id\_driver = 4116 -- > 

**3) ТРИГГЕР**

-- триггер для добавление доп. опции "Wi-Fi" при заказе поездки через приложение

Delimiter $$

create trigger add\_opt\_for\_mob\_app

before insert on additional\_options\_has\_trips

for each row

begin

if ((select order\_methods\_id\_order\_methods from trips where new.trips\_id\_trips = id\_trips) = 12) then -- если метод заказа "приложение"

set new.additional\_options\_id\_additional\_options = 5; -- доп. опция - Wi-Fi

end if;

end $$

delimiter ;

drop trigger add\_opt\_for\_mob\_app; -- дропнуть триггер

**Пример 1**: (когда способ заказа - приложение)

insert into additional\_options\_has\_trips values (4, 93); -- добавить значение

Проверим сработал ли триггер с помощью кода (*доп. опция с id = 4 – детское кресло, доп. опция с id = 5 – Wi-Fi*):

select trips.departure\_address, trips.arrival\_address, forms\_of\_payment.name as forms\_of\_payment, order\_methods.name as order\_methods,

clients.nickname, estimation, taxi\_companies.name as taxi\_companies, taxi\_drivers.nickname as taxi\_drivers, autos.number, autos.color,

autos.name as autos\_name, autos.brand, group\_concat(additional\_options.name separator ", ") as additional\_options

from trips

join forms\_of\_payment on forms\_of\_payment.id\_forms\_of\_payment = trips.forms\_of\_payment\_id\_forms\_of\_payment

join order\_methods on order\_methods.id\_order\_methods = trips.order\_methods\_id\_order\_methods

join clients on clients.id\_clients = trips.clients\_id\_clients

join contracts on contracts.id\_contracts = trips.contracts\_id\_contracts

join taxi\_companies on taxi\_companies.id\_taxi\_companies = contracts.taxi\_companies\_id\_taxi\_companies

join taxi\_drivers on taxi\_drivers.id\_taxi\_drivers = contracts.taxi\_drivers\_id\_taxi\_drivers

join autos on autos.id\_autos = contracts.autos\_id\_autos

join additional\_options\_has\_trips on additional\_options\_has\_trips.trips\_id\_trips = trips.id\_trips

join additional\_options on additional\_options.id\_additional\_options = additional\_options\_has\_trips.additional\_options\_id\_additional\_options

where trips.id\_trips = 93

group by trips.id\_trips; -- вывести всю информацию о поездке и вывести через запятую все доп. оцпии этой поездки (поездка с id 93)

Это результат следующего вызова, когда способ заказа –приложение:



**Пример 2: (**когдаспособ заказа – не приложение**)**

insert into additional\_options\_has\_trips values (9, 90); -- добавить значение (9 - светомузыка)

select trips.departure\_address, trips.arrival\_address, forms\_of\_payment.name as forms\_of\_payment, order\_methods.name as order\_methods,

clients.nickname, estimation, taxi\_companies.name as taxi\_companies, taxi\_drivers.nickname as taxi\_drivers, autos.number, autos.color,

autos.name as autos\_name, autos.brand, group\_concat(additional\_options.name separator ", ") as additional\_options

from trips

join forms\_of\_payment on forms\_of\_payment.id\_forms\_of\_payment = trips.forms\_of\_payment\_id\_forms\_of\_payment

join order\_methods on order\_methods.id\_order\_methods = trips.order\_methods\_id\_order\_methods

join clients on clients.id\_clients = trips.clients\_id\_clients

join contracts on contracts.id\_contracts = trips.contracts\_id\_contracts

join taxi\_companies on taxi\_companies.id\_taxi\_companies = contracts.taxi\_companies\_id\_taxi\_companies

join taxi\_drivers on taxi\_drivers.id\_taxi\_drivers = contracts.taxi\_drivers\_id\_taxi\_drivers

join autos on autos.id\_autos = contracts.autos\_id\_autos

join additional\_options\_has\_trips on additional\_options\_has\_trips.trips\_id\_trips = trips.id\_trips

join additional\_options on additional\_options.id\_additional\_options = additional\_options\_has\_trips.additional\_options\_id\_additional\_options

where trips.id\_trips = 90

group by trips.id\_trips; -- вывести всю информацию о поездке и вывести через запятую все доп. оцпии этой поездки (поездка с id 90)

Это результат следующего вызова, когда способ заказа – НЕ приложение:



\*В этом случае клиент уже выбрал кондиционер и массажное кресло, но решил добавить светомузыку