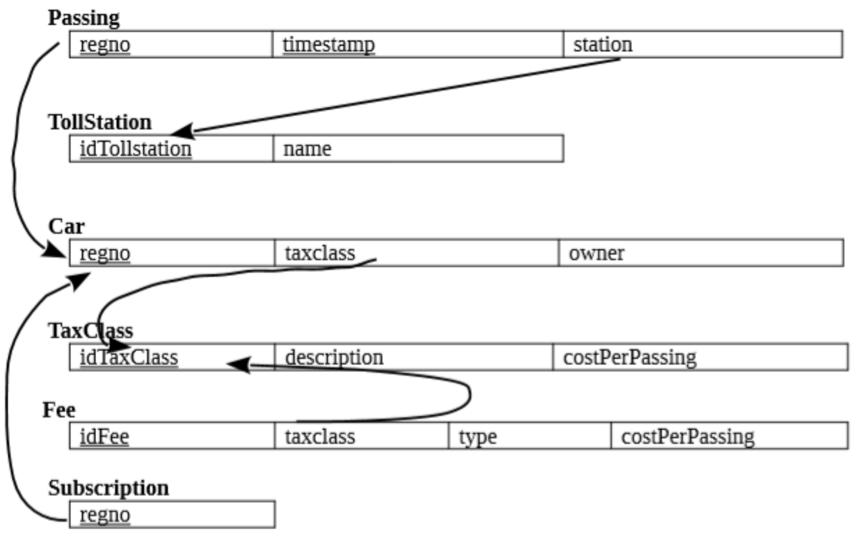
DAT151 – OBLIG5

Optimalisering og implementasjon av større database.

Adrian R.J. Mortensen

Innhold

Oppgave 1: Normal form



1. Er skjemaet 1NF?

Et skjema er 1NF hvis og bare hvis alle underliggende domener har atomiske verdier.

All data er atomære om vi antar «timestamp» er det.

2. Er det 2NF?

Et skjema er 2NF hvis og bare hvis det er 1NF og alle ikke nøkkel attributter er knyttet til en kandidat nøkkel eller en annen ikke nøkkel attributt.

Her kan man diskutere at Passing ikke følger denne regelen. Ettersom tollstasjon ikke er avhengig av hverken «regno» eller «timestamp».

3. er den 3nf

Om man mener station er knyttet til timestamp og regno er den det. Ellers er den ikke det siden den ikke fyller 2nf.

Oppgave 2: Implementasjon av fysisk skjema og test miljø.

Jeg bruker skjemaet fra forrige oppgave uten endringer.

SQL filer kan finnes på https://github.com/H571531/DAT151/tree/master/Oblig5

```
> Source /home/admo/git/DAT151/Oblig5/SQL/CreateTable.sql

CREATE TABLE IF NOT EXISTS  ImportTable  (
   regno  VARCHAR(45) NULL,
   tid  DATETIME NULL,
   idTollstation  VARCHAR(45) NULL,
   tollname  VARCHAR(45) NULL,
   OwnerName  VARCHAR(45) NULL,
   taxId varchar(3),
   taxDesc  VARCHAR(45) NULL,
   Subscription  VARCHAR(10),
   TollSFee  INT(11) NULL,
   SubFee  INT(11) NULL
   )
ENGINE = MyISAM;

CREATE TABLE IF NOT EXISTS  Tollstation  (
   idTollstation  SMALLINT UNSIGNED NOT NULL,
   name  VARCHAR(85) NULL,
  PRIMARY KEY  ( idTollstation ))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS  TaxClass  (
    idTaxClass  SMALLINT UNSIGNED NOT NULL,
    description  TEXT NULL,
    PRIMARY KEY ( idTaxClass )
  )
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS Fee  (
   idFee  SMALLINT NOT NULL AUTO_INCREMENT,
   taxclass  SMALLINT UNSIGNED NOT NULL,
   type  ENUM('regular', 'withsubscription') NOT NULL,
   costPerPassing  DECIMAL(5,2) NOT NULL,
  PRIMARY KEY ( idFee ),
  CONSTRAINT TeacherFK FOREIGN KEY (taxclass) REFERENCES
TaxClass(idTaxClass)
)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS  Car  (
   regno  CHAR(7) NOT NULL,
   owner  VARCHAR(85) NULL,
   taxclass  SMALLINT UNSIGNED NOT NULL,
  PRIMARY KEY ( regno ),
  CONSTRAINT  fk_Car_TaxClass1
    FOREIGN KEY ( taxclass )
    REFERENCES TaxClass (idTaxClass)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
CREATE TABLE IF NOT EXISTS  Passing  (
   regno  CHAR(7) NOT NULL,
   timestamp  TIMESTAMP NOT NULL,
   tollstation  SMALLINT UNSIGNED NOT NULL,
  PRIMARY KEY ( regno ,  timestamp ),
  CONSTRAINT  fk_Passing_1
    FOREIGN KEY ( tollstation )
    REFERENCES  Tollstation  ( idTollstation )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT  fk_Passing_Subscription1
    FOREIGN KEY ( regno )
    REFERENCES Car (regno)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS  Subscription  (
   regno  CHAR(7) NOT NULL,
  PRIMARY KEY ( regno ),
  CONSTRAINT  fk_Subscription_Car1
    FOREIGN KEY (regno)
    REFERENCES Car (regno)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Get data

```
#!/bin/bash
wget https://eple.hib.no/fag/dat151/v2020/carpassingdb.txt
```

(reason for making a script is due to Github limitations (Can’t push files over 100MB))

```
> LOAD DATA LOCAL INFILE '/home/admo/git/DAT151/Oblig5/SQL/carpassingdb.txt' INTO TABLE
ImportTable FIELDS TERMINATED BY ';';
```

Move data to tables

```
-- Tollstation
INSERT INTO Tollstation
SELECT DISTINCT idTollstation,tollname
FROM ImportTable;

-- TaxClass
INSERT INTO TaxClass
SELECT DISTINCT taxId,taxDesc
FROM ImportTable;

-- Car
INSERT INTO Car
SELECT DISTINCT regno,OwnerName,taxId
FROM ImportTable;

-- Subscription
INSERT INTO Subscription
SELECT DISTINCT regno
FROM ImportTable
WHERE Subscription='yes';

-- Passing
INSERT INTO Passing
SELECT DISTINCT regno,tid,idTollstation
FROM ImportTable;

--  Fee
INSERT INTO Fee (taxclass,type,costPerPassing)
SELECT DISTINCT taxId,'regular',TollSFee
FROM ImportTable
WHERE Subscription='no';

INSERT INTO Fee (taxclass,type,costPerPassing)
SELECT DISTINCT taxId,'withsubscription',SubFee
FROM ImportTable
WHERE Subscription='yes';
```

Oppgave 3: Optimalisering av database

a) Query 1

Finner navnet og tiden til noen som har kjørt igjennom en bomstasjon en spesifikk dag.

```
MariaDB [Oblig5]> SELECT SQL_NO_CACHE C.owner, P.timestamp
    -> FROM Car C JOIN Passing P USING(regno)
    -> WHERE YEAR(P.timestamp)=2018 AND MONTH(P.timestamp)=3
    -> AND DAYOFWEEK(P.timestamp)=1;
+--------------------+---------------------+
| owner              | timestamp           |
+--------------------+---------------------+
| Urfan SandÃ¸y      | 2018-03-04 04:30:00 |
| Urfan SandÃ¸y      | 2018-03-04 05:42:00 |
| Birte Fossum       | 2018-03-04 02:18:00 |
| Birte Fossum       | 2018-03-04 23:31:00 |
| Stanley Ingvaldsen | 2018-03-04 02:18:00 |
| Stanley Ingvaldsen | 2018-03-04 23:31:00 |
| Amar Wiig          | 2018-03-04 05:53:00 |
| Minda Larssen      | 2018-03-04 14:24:00 |
| Storm Nordstrand   | 2018-03-04 03:09:00 |
| Marcus Hafstad     | 2018-03-04 14:24:00 |
| Mathilde Lillevik  | 2018-03-04 04:30:00 |
| Mathilde Lillevik  | 2018-03-04 05:42:00 |
| Zilan Solbakken    | 2018-03-04 05:53:00 |
| Kurt Aslaksen      | 2018-03-04 19:02:00 |
| Dilara Skar        | 2018-03-04 03:09:00 |
| Annette Sara       | 2018-03-04 19:02:00 |
+--------------------+---------------------+
16 rows in set (3.085 sec)
```

Profiling before optimization:

Ran the query 10 times before profiling start.

```
MariaDB [Oblig5]> SHOW PROFILES;
+----------+------------+---------------------------------------------------------+
| Query_ID | Duration   | Query                                                   |
+----------+------------+---------------------------------------------------------+
|        1 | 3.08046248 | SELECT SQL_NO_CACHE C.owner, P.timestamp                |
|          |            | FROM Car C JOIN Passing P USING(regno)                  |
|          |            | WHERE YEAR(P.timestamp)=2018 AND MONTH(P.timestamp)=3   |
|          |            | AND DAYOFWEEK(P.timestamp)=1                            |
|        2 | 3.14813736 | SELECT SQL_NO_CACHE C.owner, P.timestamp                |
|          |            | FROM Car C JOIN Passing P USING(regno)                  |
|          |            | WHERE YEAR(P.timestamp)=2018 AND MONTH(P.timestamp)=3   |
|          |            | AND DAYOFWEEK(P.timestamp)=1                            |
|        3 | 3.15549203 | SELECT SQL_NO_CACHE C.owner, P.timestamp                |
|          |            | FROM Car C JOIN Passing P USING(regno)                  |
|          |            | WHERE YEAR(P.timestamp)=2018 AND MONTH(P.timestamp)=3   |
|          |            | AND DAYOFWEEK(P.timestamp)=1                            |
|        4 | 3.16799906 | SELECT SQL_NO_CACHE C.owner, P.timestamp                |
|          |            | FROM Car C JOIN Passing P USING(regno)                  |
|          |            | WHERE YEAR(P.timestamp)=2018 AND MONTH(P.timestamp)=3   |
|          |            | AND DAYOFWEEK(P.timestamp)=1                            |
|        5 | 3.16238553 | SELECT SQL_NO_CACHE C.owner, P.timestamp                |
|          |            | FROM Car C JOIN Passing P USING(regno)                  |
|          |            | WHERE YEAR(P.timestamp)=2018 AND MONTH(P.timestamp)=3   |
|          |            | AND DAYOFWEEK(P.timestamp)=1                            |
+----------+------------+---------------------------------------------------------+
```

Picking Query 3 for furter info:

```
MariaDB [Oblig5]> SHOW PROFILE for QUERY 3;
+------------------------+----------+
| Status                 | Duration |
+------------------------+----------+
| Starting               | 0.000083 |
| Checking permissions   | 0.000004 |
| Opening tables         | 0.000032 |
| After opening tables   | 0.000006 |
| System lock            | 0.000002 |
| Table lock             | 0.000005 |
| Init                   | 0.000029 |
| Optimizing             | 0.000018 |
| Statistics             | 0.000021 |
| Preparing              | 0.000018 |
| Executing              | 0.000002 |
| Sending data           | 3.155202 |
| End of update loop     | 0.000017 |
| Query end              | 0.000003 |
| Commit                 | 0.000004 |
| Closing tables         | 0.000003 |
| Unlocking tables       | 0.000001 |
| Closing tables         | 0.000010 |
| Starting cleanup       | 0.000002 |
| Freeing items          | 0.000008 |
| Updating status        | 0.000021 |
| Reset for next command | 0.000003 |
+------------------------+----------+
```

Most of the time in this query is spent between Executing and Sending data. (The Duration label is abit misleading…) Each time means the time elapsed between the previous event and the new event.

EXPLAIN before Optimization

```
MariaDB [Oblig5]> EXPLAIN SELECT SQL_NO_CACHE C.owner, P.timestamp FROM Car C JOIN Passing P USING(regno) WHERE YEAR
(P.timestamp)=2018 AND MONTH (P.timestamp)=3 AND DAYOFWEEK(P.timestamp)=1;
+------+-------------+-------+------+---------------+---------+---------+----------------+--------+-------------+
| id   | select_type | table | type | possible_keys | key     | key_len | ref            | rows   | Extra       |
+------+-------------+-------+------+---------------+---------+---------+----------------+--------+-------------+
|    1 | SIMPLE      | C     | ALL  | PRIMARY       | NULL    | NULL    | NULL           | 203998 |             |
|    1 | SIMPLE      | P     | ref  | PRIMARY       | PRIMARY | 7       | Oblig5.C.regno |      1 | Using where |
+------+-------------+-------+------+---------------+---------+---------+----------------+--------+-------------+
2 rows in set (0.000 sec)
```

Optimization of query and table.

By breaking normalization I might get a better result.

The easiest change I can see right now is remove the need to join the car table with Passing.

The only thing we need from the Car table is the name. Therefore I add name to the Passing.

```
CREATE TABLE IF NOT EXISTS  Passing2  (
   regno  CHAR(7) NOT NULL,
   timestamp  TIMESTAMP NOT NULL,
   tollstation  SMALLINT UNSIGNED NOT NULL,
   owner VARCHAR(85),
  PRIMARY KEY ( regno ,  timestamp ),
  CONSTRAINT  fk_Passing_2
    FOREIGN KEY ( tollstation )
    REFERENCES  Tollstation  ( idTollstation )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT  fk_Passing_Subscription2
    FOREIGN KEY ( regno )
    REFERENCES Car (regno)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- Passing2
INSERT INTO Passing2
SELECT DISTINCT regno,tid,idTollstation,OwnerName
FROM ImportTable;


-- Query
SELECT SQL_NO_CACHE owner,timestamp
FROM Passing2 WHERE YEAR(timestamp)=2018 AND MONTH(timestamp)=3
AND DAYOFWEEK(timestamp)=1;


+--------------------+---------------------+
| owner              | timestamp           |
+--------------------+---------------------+
| Urfan SandÃ¸y      | 2018-03-04 04:30:00 |
| Urfan SandÃ¸y      | 2018-03-04 05:42:00 |
| Birte Fossum       | 2018-03-04 02:18:00 |
| Birte Fossum       | 2018-03-04 23:31:00 |
| Stanley Ingvaldsen | 2018-03-04 02:18:00 |
| Stanley Ingvaldsen | 2018-03-04 23:31:00 |
| Amar Wiig          | 2018-03-04 05:53:00 |
| Minda Larssen      | 2018-03-04 14:24:00 |
| Storm Nordstrand   | 2018-03-04 03:09:00 |
| Marcus Hafstad     | 2018-03-04 14:24:00 |
| Mathilde Lillevik  | 2018-03-04 04:30:00 |
| Mathilde Lillevik  | 2018-03-04 05:42:00 |
| Zilan Solbakken    | 2018-03-04 05:53:00 |
| Kurt Aslaksen      | 2018-03-04 19:02:00 |
| Dilara Skar        | 2018-03-04 03:09:00 |
| Annette Sara       | 2018-03-04 19:02:00 |
+--------------------+---------------------+
16 rows in set (2.555 sec)
```

Can already see some time saved.

Profiling after optimization

Query has been ran 10 times before profiling was turned on.

```
MariaDB [Oblig5]> show profiles;
+----------+------------+--------------------------------------------------------------------------+
| Query_ID | Duration   | Query                                                                    |
+----------+------------+--------------------------------------------------------------------------+
|        1 | 2.46529822 | SELECT SQL_NO_CACHE owner,timestamp                                      |
|          |            | FROM Passing2 WHERE YEAR(timestamp)=2018 AND MONTH(timestamp)=3          |
|          |            | AND DAYOFWEEK(timestamp)=1                                               |
|        2 | 2.51719002 | SELECT SQL_NO_CACHE owner,timestamp                                      |
|          |            | FROM Passing2 WHERE YEAR(timestamp)=2018 AND MONTH(timestamp)=3          |
|          |            | AND DAYOFWEEK(timestamp)=1                                               |
|        3 | 2.52976462 | SELECT SQL_NO_CACHE owner,timestamp                                      |
|          |            | FROM Passing2 WHERE YEAR(timestamp)=2018 AND MONTH(timestamp)=3          |
|          |            | AND DAYOFWEEK(timestamp)=1                                               |
|        4 | 2.51797654 | SELECT SQL_NO_CACHE owner,timestamp                                      |
|          |            | FROM Passing2 WHERE YEAR(timestamp)=2018 AND MONTH(timestamp)=3          |
|          |            | AND DAYOFWEEK(timestamp)=1                                               |
|        5 | 2.47903186 | SELECT SQL_NO_CACHE owner,timestamp                                      |
|          |            | FROM Passing2 WHERE YEAR(timestamp)=2018 AND MONTH(timestamp)=3          |
|          |            | AND DAYOFWEEK(timestamp)=1                                               |
+----------+------------+--------------------------------------------------------------------------+
5 rows in set (0.000 sec)
```

Showing for query 3

```
MariaDB [Oblig5]> show profile for query 3;
+------------------------+----------+
| Status                 | Duration |
+------------------------+----------+
| Starting               | 0.000077 |
| Checking permissions   | 0.000005 |
| Opening tables         | 0.000019 |
| After opening tables   | 0.000004 |
| System lock            | 0.000003 |
| Table lock             | 0.000006 |
| Init                   | 0.000025 |
| Optimizing             | 0.000014 |
| Statistics             | 0.000014 |
| Preparing              | 0.000019 |
| Executing              | 0.000002 |
| Sending data           | 2.529506 |
| End of update loop     | 0.000018 |
| Query end              | 0.000002 |
| Commit                 | 0.000005 |
| Closing tables         | 0.000003 |
| Unlocking tables       | 0.000001 |
| Closing tables         | 0.000009 |
| Starting cleanup       | 0.000002 |
| Freeing items          | 0.000006 |
| Updating status        | 0.000021 |
| Reset for next command | 0.000003 |
+------------------------+----------+
22 rows in set (0.000 sec)
```

Explain optimized query

```
MariaDB [Oblig5]> EXPLAIN SELECT SQL_NO_CACHE owner,timestamp
    -> FROM Passing2 WHERE YEAR(timestamp)=2018 AND MONTH(timestamp)=3
    -> AND DAYOFWEEK(timestamp)=1;
+------+-------------+----------+------+---------------+------+---------+------+---------+-------------+
| id   | select_type | table    | type | possible_keys | key  | key_len | ref  | rows    | Extra       |
+------+-------------+----------+------+---------------+------+---------+------+---------+-------------+
|    1 | SIMPLE      | Passing2 | ALL  | NULL          | NULL | NULL    | NULL | 5035359 | Using where |
+------+-------------+----------+------+---------------+------+---------+------+---------+-------------+
1 row in set (0.001 sec)
```

Indexes

Now attempting to add a index on the timestamp collumn to perhaps make the query abit faster.

```
CREATE INDEX Timestamp ON Passing2(timestamp);
```

This adds very little to making the query any faster.

This also counts for Passing (the original table) the optimizer does not even want to use the index.

My thoughts aret hat this is because there is no need for it.

Note:

It is worth noting that on pre cache queries The un optimized query used 7s and the optimized query used 4s.

b) Query 2

Finner de som har abonnement med total kostnad over 4000

```
CREATE INDEX Timestamp ON Passing(timestamp);
-- b
MariaDB [Oblig5]> SELECT SQL_NO_CACHE C.owner AS carowner,
Sum(F.costPerPassing) AS totalfee
    -> FROM Car C JOIN Passing P USING(regno)
    -> JOIN TaxClass T ON C.taxclass=T.idTaxClass
    -> JOIN Fee F ON F.taxclass=T.idTaxClass
    -> JOIN Subscription S USING(regno)
    -> WHERE F.type='withsubscription'
    -> GROUP BY C.owner HAVING totalfee > 4000;
+----------------+----------+
| carowner       | totalfee |
+----------------+----------+
| Ansgar Oftedal |  4210.00 |
| Rina Kvalheim  |  4605.00 |
+----------------+----------+
2 rows in set (2.475 sec)

MariaDB [Oblig5]> EXPLAIN SELECT SQL_NO_CACHE C.owner AS carowner,
Sum(F.costPerPassing) AS totalfee
    -> FROM Car C JOIN Passing P USING(regno)
    -> JOIN TaxClass T ON C.taxclass=T.idTaxClass
    -> JOIN Fee F ON F.taxclass=T.idTaxClass
    -> JOIN Subscription S USING(regno)
    -> WHERE F.type='withsubscription'
    -> GROUP BY C.owner HAVING totalfee > 4000;
+------+-------------+-------+--------+---------------------------+------------------+---------+---------------------+------+--------------------------
--------------------+
| id   | select_type | table | type   | possible_keys             | key              | key_len | ref                 | rows | Extra
|
+------+-------------+-------+--------+---------------------------+------------------+---------+---------------------+------+--------------------------
--------------------+
|    1 | SIMPLE      | T     | index  | PRIMARY                   | PRIMARY          | 2       | NULL                |   10 | Using index; Using
temporary; Using filesort |
|    1 | SIMPLE      | F     | ref    | TeacherFK                 | TeacherFK        | 2       | Oblig5.T.idTaxClass |    1 | Using where
|
|    1 | SIMPLE      | C     | ref    | PRIMARY,fk_Car_TaxClass1 | fk_Car_TaxClass1 | 2       | Oblig5.T.idTaxClass |    1 |
|
|    1 | SIMPLE      | S     | eq_ref | PRIMARY                   | PRIMARY          | 7       | Oblig5.C.regno      |    1 | Using index
|
|    1 | SIMPLE      | P     | ref    | PRIMARY,reg               | PRIMARY          | 7       | Oblig5.C.regno      |   13 |
|
+------+-------------+-------+--------+---------------------------+------------------+---------+---------------------+------+--------------------------
--------------------+
5 rows in set (0.001 sec)
```

The optimizer already uses indexes and I cant see a better way to do the indexes. However I want to attempt denormalization to see if it can get better by denormalization.

I see that Taxclass is joined with the cars taxclass. Only to get fee. Now if we get the fee into passing we wont have to do those two joins. And if we also add owner to the table we might just have to find the cars with subscription. Adding who has subscription aswell might be faster. Butt his will increase the collumn count with 3 so the negative effects of this could be negative to the outcome.

Trying this either way.

Optimizing

Sql: Table

```
CREATE TABLE IF NOT EXISTS  Passing3  (
   regno  CHAR(7) NOT NULL,
   timestamp  TIMESTAMP NOT NULL,
   tollstation  SMALLINT UNSIGNED NOT NULL,
   owner VARCHAR(85),
   type  ENUM('regular', 'withsubscription') NOT NULL,
   costPerPassing  DECIMAL(5,2) NOT NULL,
  PRIMARY KEY ( regno ,  timestamp ),
  CONSTRAINT  fk_Passing_3
    FOREIGN KEY ( tollstation )
    REFERENCES  Tollstation  ( idTollstation )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT  fk_Passing_Subscription3
    FOREIGN KEY ( regno )
    REFERENCES Car (regno)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Adding data:

```
-- Passing3
INSERT INTO Passing3
SELECT DISTINCT regno,tid,idTollstation,OwnerName,'withsubscription',SubFee
FROM ImportTable
WHERE Subscription='yes';


INSERT INTO Passing3
SELECT DISTINCT regno,tid,idTollstation,OwnerName,'regular',TollSFee
FROM ImportTable
WHERE Subscription='no';
```

Query:

```
SELECT SQL_NO_CACHE owner AS carowner, Sum(costPerPassing) AS totalfee
from Passing3
WHERE type='withsubscription'
GROUP BY owner HAVING totalfee > 4000;
```

We can see the Query being alot shorter with no joins and maybe arguably easier to read.

However did it increase performance?

Profiling

Pre denormalization

```
MariaDB [Oblig5]> show profiles;
+----------+------------+--------------------------------------------------------------------------------+
| Query_ID | Duration   | Query                                                                          |
|
+----------+------------+--------------------------------------------------------------------------------+
|        1 | 3.51759643 | SELECT SQL_NO_CACHE C.owner AS carowner, Sum(F.costPerPassing) AS totalfee     |
|          |            | FROM Car C JOIN Passing P USING(regno)                                         |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass                                     |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass                                          |
|          |            | JOIN Subscription S USING(regno)                                               |
|          |            | WHERE F.type='withsubscription'                                                |
|        2 | 3.53729194 | SELECT SQL_NO_CACHE C.owner AS carowner, Sum(F.costPerPassing) AS totalfee     |
|          |            | FROM Car C JOIN Passing P USING(regno)                                         |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass                                     |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass                                          |
|          |            | JOIN Subscription S USING(regno)                                               |
|          |            | WHERE F.type='withsubscription'                                                |
|        3 | 3.51002205 | SELECT SQL_NO_CACHE C.owner AS carowner, Sum(F.costPerPassing) AS totalfee     |
|          |            | FROM Car C JOIN Passing P USING(regno)                                         |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass                                     |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass                                          |
|          |            | JOIN Subscription S USING(regno)                                               |
|          |            | WHERE F.type='withsubscription'                                                |
|        4 | 3.47434126 | SELECT SQL_NO_CACHE C.owner AS carowner, Sum(F.costPerPassing) AS totalfee     |
|          |            | FROM Car C JOIN Passing P USING(regno)                                         |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass                                     |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass                                          |
|          |            | JOIN Subscription S USING(regno)                                               |
|          |            | WHERE F.type='withsubscription'                                                |
|        5 | 3.39511616 | SELECT SQL_NO_CACHE C.owner AS carowner, Sum(F.costPerPassing) AS totalfee     |
|          |            | FROM Car C JOIN Passing P USING(regno)                                         |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass                                     |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass                                          |
|          |            | JOIN Subscription S USING(regno)                                               |
|          |            | WHERE F.type='withsubscription'                                                |
+----------+------------+--------------------------------------------------------------------------------+
5 rows in set (0.000 sec)
```

Post denormalization

```
MariaDB [Oblig5]> show profiles;
+----------+------------+----------------------------------------------------------------------------+
| Query_ID | Duration   | Query                                                                      |
|
+----------+------------+----------------------------------------------------------------------------+
|        1 | 2.65984540 | SELECT SQL_NO_CACHE owner AS carowner, Sum(costPerPassing) AS totalfee     |
|          |            | from Passing3                                                              |
|          |            | WHERE type='withsubscription'                                              |
|          |            | GROUP BY owner HAVING totalfee > 4000                                      |
|        2 | 2.66344669 | SELECT SQL_NO_CACHE owner AS carowner, Sum(costPerPassing) AS totalfee     |
|          |            | from Passing3                                                              |
|          |            | WHERE type='withsubscription'                                              |
|          |            | GROUP BY owner HAVING totalfee > 4000                                      |
|        3 | 2.66824841 | SELECT SQL_NO_CACHE owner AS carowner, Sum(costPerPassing) AS totalfee     |
|          |            | from Passing3                                                              |
|          |            | WHERE type='withsubscription'                                              |
|          |            | GROUP BY owner HAVING totalfee > 4000                                      |
|        4 | 2.67204567 | SELECT SQL_NO_CACHE owner AS carowner, Sum(costPerPassing) AS totalfee     |
|          |            | from Passing3                                                              |
|          |            | WHERE type='withsubscription'                                              |
|          |            | GROUP BY owner HAVING totalfee > 4000                                      |
|        5 | 2.66805130 | SELECT SQL_NO_CACHE owner AS carowner, Sum(costPerPassing) AS totalfee     |
|          |            | from Passing3                                                              |
|          |            | WHERE type='withsubscription'                                              |
|          |            | GROUP BY owner HAVING totalfee > 4000                                      |
+----------+------------+----------------------------------------------------------------------------+
5 rows in set (0.000 sec)
```

Yes! The performance seem to have been improved. Only with about a second. Both profiles started after the query was ran 10 times.

Having a look at the explain for fun to see how this new query is being optimized by the optimizer.

```
+------+-------------+----------+------+---------------+------+---------+------+---------+-----------------------------------------------+
| id   | select_type | table    | type | possible_keys | key  | key_len | ref  | rows    | Extra                                         |
+------+-------------+----------+------+---------------+------+---------+------+---------+-----------------------------------------------+
|    1 | SIMPLE      | Passing3 | ALL  | NULL          | NULL | NULL    | NULL | 4812140 | Using where; Using temporary; Using filesort |
+------+-------------+----------+------+---------------+------+---------+------+---------+-----------------------------------------------+
1 row in set (0.001 sec)
```

From the explain we can see there is no joins, and no other tables to take into concideration. If there was; the lack of keys could mean we would need some sort of indexes. But we only have a select on a single table. And since we’re using where clauses I belive this is sufficiently improved from the original query.

c) Query 3

This query does the same as the previous one. This time using joins.

```
MariaDB [Oblig5]> SELECT SQL_NO_CACHE C.owner AS carowner,
    -> Sum(F.costPerPassing) AS totalfee
    ->  FROM Car C JOIN Passing P USING(regno)
    ->  JOIN TaxClass T ON C.taxclass=T.idTaxClass
    ->  JOIN Fee F ON F.taxclass=T.idTaxClass
    ->  WHERE F.type='withsubscription'
    ->  AND C.regno IN (SELECT regno FROM Subscription)
    ->  GROUP BY C.owner HAVING totalfee > 4000;
+----------------+----------+
| carowner       | totalfee |
+----------------+----------+
| Ansgar Oftedal |  4210.00 |
| Rina Kvalheim  |  4605.00 |
+----------------+----------+
2 rows in set (3.330 sec)
```

Explain

```
+------+-------------+--------------+--------+-------------------------+---------------------+---------+---------------------+------+-----------------------------------------------+
| id   | select_type | table        | type   | possible_keys           | key                 | key_len | ref                 | rows | Extra                                         |
+------+-------------+--------------+--------+-------------------------+---------------------+---------+---------------------+------+-----------------------------------------------+
|    1 | PRIMARY     | T            | index  | PRIMARY                 | PRIMARY             | 2       | NULL                |   10 | Using index; Using temporary; Using filesort |
|    1 | PRIMARY     | F            | ref    | TeacherFK               | TeacherFK           | 2       | Oblig5.T.idTaxClass |    1 | Using where                                   |
|    1 | PRIMARY     | C            | ref    | PRIMARY,fk_Car_TaxClass1 | fk_Car_TaxClass1   | 2       | Oblig5.T.idTaxClass |    1 |                                               |
|    1 | PRIMARY     | Subscription | eq_ref | PRIMARY                 | PRIMARY             | 7       | Oblig5.C.regno      |    1 | Using index                                   |
|    1 | PRIMARY     | P            | ref    | PRIMARY,reg             | PRIMARY             | 7       | Oblig5.C.regno      |   13 |                                               |
+------+-------------+--------------+--------+-------------------------+---------------------+---------+---------------------+------+-----------------------------------------------+
5 rows in set (0.001 sec)
```

Here we can see we have available keys on all joins. This is good. The Optimizer will be using indexes some places. The Key length is also fairly small meaning it only needs a few bytes of the key.

Profiling

```
MariaDB [Oblig5]> show profiles;
+----------+------------+----------------------------------------------------+
| Query_ID | Duration   | Query                                              |
|
+----------+------------+----------------------------------------------------+
|        1 | 3.00203704 | SELECT SQL_NO_CACHE C.owner AS carowner,           |
|                       | Sum(F.costPerPassing) AS totalfee                  |
|          |            | FROM Car C JOIN Passing P USING(regno)             |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass         |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass              |
|          |            | WHERE F.type='withsubscription'                    |
|          |            | AND C.regno IN (SELECT regno FROM Subscription)    |
|          |            | GROUP BY C.owner HA                                |
|        2 | 3.05971641 | SELECT SQL_NO_CACHE C.owner AS carowner,           |
|          |            | Sum(F.costPerPassing) AS totalfee                  |
|          |            | FROM Car C JOIN Passing P USING(regno)             |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass         |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass              |
|          |            | WHERE F.type='withsubscription'                    |
|          |            | AND C.regno IN (SELECT regno FROM Subscription)    |
|          |            | GROUP BY C.owner HA                                |
|        3 | 3.00294968 | SELECT SQL_NO_CACHE C.owner AS carowner,           |
|          |            | Sum(F.costPerPassing) AS totalfee                  |
|          |            | FROM Car C JOIN Passing P USING(regno)             |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass         |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass              |
|          |            | WHERE F.type='withsubscription'                    |
|          |            | AND C.regno IN (SELECT regno FROM Subscription)    |
|          |            | GROUP BY C.owner HA                                |
|        4 | 3.11308706 | SELECT SQL_NO_CACHE C.owner AS carowner,           |
|          |            | Sum(F.costPerPassing) AS totalfee                  |
|          |            | FROM Car C JOIN Passing P USING(regno)             |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass         |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass              |
|          |            | WHERE F.type='withsubscription'                    |
|          |            | AND C.regno IN (SELECT regno FROM Subscription)    |
|          |            | GROUP BY C.owner HA                                |
|        5 | 3.08940385 | SELECT SQL_NO_CACHE C.owner AS carowner,           |
|          |            | Sum(F.costPerPassing) AS totalfee                  |
|          |            | FROM Car C JOIN Passing P USING(regno)             |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass         |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass              |
|          |            | WHERE F.type='withsubscription'                    |
|          |            | AND C.regno IN (SELECT regno FROM Subscription)    |
|          |            | GROUP BY C.owner HA                                |
+----------+------------+----------------------------------------------------+
5 rows in set (0.000 sec)
```

In profiling we can see that this Query is performs better than the previous one that used joins instead of subqueries. But still not as good as the denormalized version. Therefore I would suggest using the same sollution as in b)

d) Query 4 (same as 2)

This query will be doing the same as in query 2 but for the people without a subscription

```
MariaDB [Oblig5]> SELECT SQL_NO_CACHE C.owner AS carowner,
Sum(F.costPerPassing) AS totalfee
    -> FROM Car C JOIN Passing P USING(regno)
    -> JOIN TaxClass T ON C.taxclass=T.idTaxClass
    -> JOIN Fee F ON F.taxclass=T.idTaxClass
    -> Left JOIN Subscription S USING(regno)
    -> WHERE F.type='withsubscription'
    -> AND S.regno IS NULL
    -> GROUP BY C.owner HAVING totalfee > 4000;
+------------------+----------+
| carowner         | totalfee |
+------------------+----------+
| Glen FjÃ¸rtoft    |  4010.00 |
| Lasse Nakken     |  4316.00 |
| Ulva Hanssen     |  4130.00 |
| Yulia Lie        |  4260.00 |
+------------------+----------+
4 rows in set (3.746 sec)
```

(difference being that we do a Left JOIN on Subscription and look for the regnr not in the subscription table.)

First lets have a look at the performance of this query as is.

```
MariaDB [Oblig5]> show profiles;
+----------+------------+----------------------------------------------------------------------------------+
| Query_ID | Duration   | Query                                                                            |
|
+----------+------------+----------------------------------------------------------------------------------+
|        1 | 3.72908210 | SELECT SQL_NO_CACHE C.owner AS carowner, Sum(F.costPerPassing) AS totalfee       |
|          |            | FROM Car C JOIN Passing P USING(regno)                                           |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass                                       |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass                                            |
|          |            | Left JOIN Subscription S USING(regno)                                            |
|          |            | WHERE F.type='withsubscription'                                                  |
|        2 | 3.70657548 | SELECT SQL_NO_CACHE C.owner AS carowner, Sum(F.costPerPassing) AS totalfee       |
|          |            | FROM Car C JOIN Passing P USING(regno)                                           |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass                                       |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass                                            |
|          |            | Left JOIN Subscription S USING(regno)                                            |
|          |            | WHERE F.type='withsubscription'                                                  |
|        3 | 3.62856547 | SELECT SQL_NO_CACHE C.owner AS carowner, Sum(F.costPerPassing) AS totalfee       |
|          |            | FROM Car C JOIN Passing P USING(regno)                                           |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass                                       |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass                                            |
|          |            | Left JOIN Subscription S USING(regno)                                            |
|          |            | WHERE F.type='withsubscription'                                                  |
|        4 | 3.70396787 | SELECT SQL_NO_CACHE C.owner AS carowner, Sum(F.costPerPassing) AS totalfee       |
|          |            | FROM Car C JOIN Passing P USING(regno)                                           |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass                                       |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass                                            |
|          |            | Left JOIN Subscription S USING(regno)                                            |
|          |            | WHERE F.type='withsubscription'                                                  |
|        5 | 3.71108078 | SELECT SQL_NO_CACHE C.owner AS carowner, Sum(F.costPerPassing) AS totalfee       |
|          |            | FROM Car C JOIN Passing P USING(regno)                                           |
|          |            | JOIN TaxClass T ON C.taxclass=T.idTaxClass                                       |
|          |            | JOIN Fee F ON F.taxclass=T.idTaxClass                                            |
|          |            | Left JOIN Subscription S USING(regno)                                            |
|          |            | WHERE F.type='withsubscription'                                                  |
+----------+------------+----------------------------------------------------------------------------------+
5 rows in set (0.000 sec)
```

Now lets have a look at why these queries take almost 4s

```
+------+-------------+-------+--------+---------------------------+-------------------+---------+----------------------+------+--------------------------
-------------------+
| id   | select_type | table | type   | possible_keys             | key               | key_len | ref                  | rows | Extra
|
+------+-------------+-------+--------+---------------------------+-------------------+---------+----------------------+------+--------------------------
-------------------+
|    1 | SIMPLE      | T     | index  | PRIMARY                   | PRIMARY           | 2       | NULL                 |   10 | Using index; Using
temporary; Using filesort |
|    1 | SIMPLE      | F     | ref    | TeacherFK                 | TeacherFK         | 2       | Oblig5.T.idTaxClass  |    1 | Using where
|
|    1 | SIMPLE      | C     | ref    | PRIMARY,fk_Car_TaxClass1  | fk_Car_TaxClass1  | 2       | Oblig5.T.idTaxClass  |    1 |
|
|    1 | SIMPLE      | S     | eq_ref | PRIMARY                   | PRIMARY           | 7       | Oblig5.C.regno       |    1 | Using where; Using index;
Not exists                |
|    1 | SIMPLE      | P     | ref    | PRIMARY,reg               | PRIMARY           | 7       | Oblig5.C.regno       |   13 |
|
+------+-------------+-------+--------+---------------------------+-------------------+---------+----------------------+------+--------------------------
-------------------+
```

Now this explain shows a similar story to the others. I don't see a Index help for this as it is using indexes and has available keys.

Will be testing the same normalization scheme as in 2 as this is a very similar operation.

```
+----------+------------+---------------------------------------------------
| Query_ID | Duration   | Query
|
+----------+------------+------
|        1 | 3.16445208 | SELECT SQL_NO_CACHE owner AS carowner, |
Sum(costPerPassing) AS totalfee
from Passing3
WHERE type='regular'
GROUP BY owner HAVING totalfee > 4000 |
|        2 | 3.11762077 | SELECT SQL_NO_CACHE owner AS carowner,
Sum(costPerPassing) AS totalfee
|          |            |from Passing3
|          |            | WHERE type='regular'
|          |            | GROUP BY owner HAVING totalfee > 4000 |
|        3 | 3.12460796 | SELECT SQL_NO_CACHE owner AS carowner,
Sum(costPerPassing) AS totalfee
|          |            | from Passing3
|          |            | WHERE type='regular'
|          |            | GROUP BY owner HAVING totalfee > 4000 |
|        4 | 3.08037990 | SELECT SQL_NO_CACHE owner AS carowner, |
Sum(costPerPassing) AS totalfee
|          |            | from Passing3
|          |            | WHERE type='regular'
|          |GROUP BY owner HAVING totalfee > 4000 |
+----------+------------+---------------------------------------------------
4 rows in set (0.000 sec)
```

Excuse the formatting on this table.

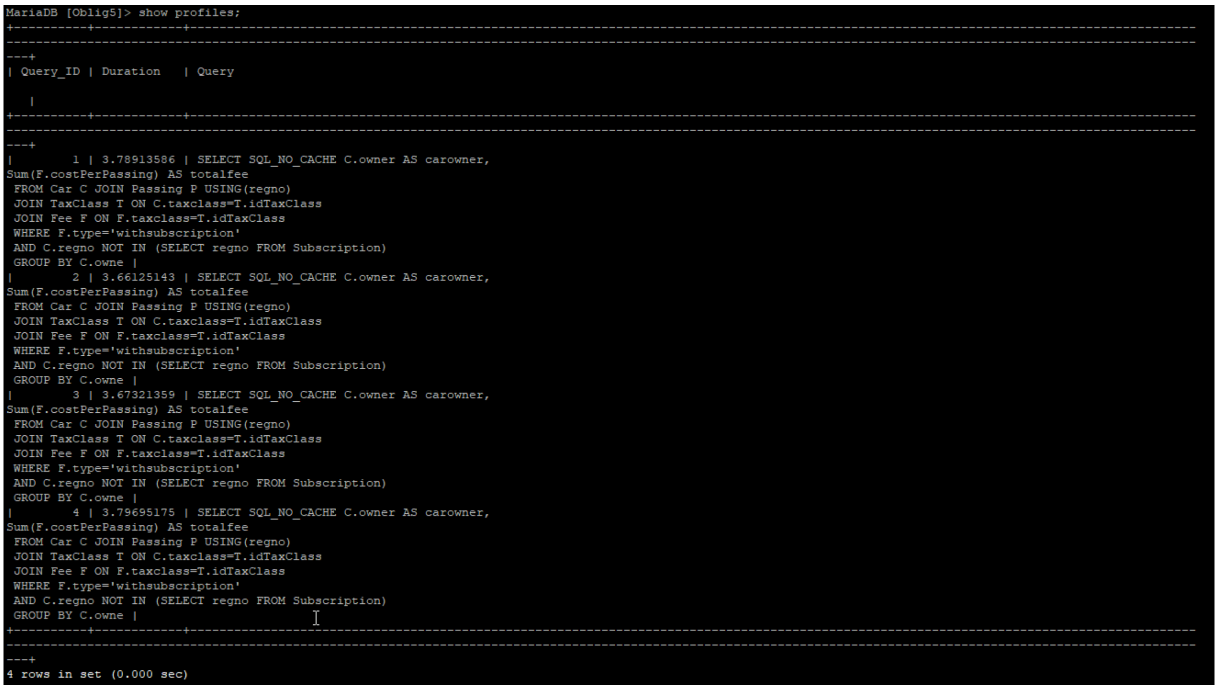
The improvement is there however. 3.08 being the lowest is about .6s from the original query.

e) Query 5 (same as 3)

This query will be the same as Query 4 just using subqueries.

```
MariaDB [Oblig5]> SELECT SQL_NO_CACHE C.owner AS carowner,
    -> Sum(F.costPerPassing) AS totalfee
    ->  FROM Car C JOIN Passing P USING(regno)
    ->  JOIN TaxClass T ON C.taxclass=T.idTaxClass
    ->  JOIN Fee F ON F.taxclass=T.idTaxClass
    ->  WHERE F.type='withsubscription'
    ->  AND C.regno NOT IN (SELECT regno FROM Subscription)
    ->  GROUP BY C.owner HAVING totalfee > 4000
    -> ;
+------------------+----------+
| carowner         | totalfee |
+------------------+----------+
| Glen FjÃ¸rtoft    |  4010.00 |
| Lasse Nakken     |  4316.00 |
| Ulva Hanssen     |  4130.00 |
| Yulia Lie        |  4260.00 |
+------------------+----------+
4 rows in set (3.746 sec)
```

(Difference being from «Query 3» being «not in» regno subscription)



```
MariaDB [Oblig5]> show profiles;
+----------+------------+------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------
---+
| Query_ID | Duration   | Query

   |
+----------+------------+------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------
---+
|        1 | 3.78913586 | SELECT SQL_NO_CACHE C.owner AS carowner,
Sum(F.costPerPassing) AS totalfee
 FROM Car C JOIN Passing P USING(regno)
 JOIN TaxClass T ON C.taxclass=T.idTaxClass
 JOIN Fee F ON F.taxclass=T.idTaxClass
 WHERE F.type='withsubscription'
 AND C.regno NOT IN (SELECT regno FROM Subscription)
 GROUP BY C.owne |
|        2 | 3.66125143 | SELECT SQL_NO_CACHE C.owner AS carowner,
Sum(F.costPerPassing) AS totalfee
 FROM Car C JOIN Passing P USING(regno)
 JOIN TaxClass T ON C.taxclass=T.idTaxClass
 JOIN Fee F ON F.taxclass=T.idTaxClass
 WHERE F.type='withsubscription'
 AND C.regno NOT IN (SELECT regno FROM Subscription)
 GROUP BY C.owne |
|        3 | 3.67321359 | SELECT SQL_NO_CACHE C.owner AS carowner,
Sum(F.costPerPassing) AS totalfee
 FROM Car C JOIN Passing P USING(regno)
 JOIN TaxClass T ON C.taxclass=T.idTaxClass
 JOIN Fee F ON F.taxclass=T.idTaxClass
 WHERE F.type='withsubscription'
 AND C.regno NOT IN (SELECT regno FROM Subscription)
 GROUP BY C.owne |
|        4 | 3.79695175 | SELECT SQL_NO_CACHE C.owner AS carowner,
Sum(F.costPerPassing) AS totalfee
 FROM Car C JOIN Passing P USING(regno)
 JOIN TaxClass T ON C.taxclass=T.idTaxClass
 JOIN Fee F ON F.taxclass=T.idTaxClass
 WHERE F.type='withsubscription'
 AND C.regno NOT IN (SELECT regno FROM Subscription)
 GROUP BY C.owne |
+----------+------------+------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------
---+
4 rows in set (0.000 sec)
```

This query did about as well as the previous one. And since it is the same operation the denormalization from the previous query can be used.

f) Query 6

Denne spørringen finner navnet til de som har gjort en passering på en spesifikk dag og tid.

```
MariaDB [Oblig5]>  SELECT SQL_NO_CACHE C.owner FROM Car C WHERE C.regno
    -> IN (SELECT P.regno FROM Passing P JOIN Tollstation T
    -> ON P.tollstation = T.idTollstation
    -> WHERE T.name LIKE 'Gravdal'
    -> AND YEAR(P.timestamp)=2018
    -> AND MONTH(P.timestamp)=2
    -> AND DAYOFWEEK(P.timestamp)=1
    -> AND HOUR(P.timestamp) = 3);
+------------------------+
| owner                  |
+------------------------+
| Olav HÃ¦tta            |
| Bjarnhild Reistad      |
| Jarle Aarnes           |
--!!Shortening the output--
| Noor Evensen           |
| Danny Straume          |
| Hossein Kolberg        |
| Hassan Haugland        |
| Odin Persson           |
| Enid Nicolaysen        |
| Oddveig Roald          |
| Xhavit HÃ¸iland        |
| VebjÃ¸rn Bauge         |
| Ramona SÃ¸vik          |
| Oda Dammen             |
| Nicholas Heiberg       |
+------------------------+
177 rows in set (3.396 sec)


MariaDB [Oblig5]>  EXPLAIN  SELECT SQL_NO_CACHE C.owner FROM Car C WHERE
C.regno
    -> IN (SELECT P.regno FROM Passing P JOIN Tollstation T
    -> ON P.tollstation = T.idTollstation
    -> WHERE T.name LIKE 'Gravdal'
    -> AND YEAR(P.timestamp)=2018
    -> AND MONTH(P.timestamp)=2
    -> AND DAYOFWEEK(P.timestamp)=1
    -> AND HOUR(P.timestamp) = 3);
+------+-------------+-------+--------+---------------------------+---------+---------+----------------------+--------+---------------------------+
| id   | select_type | table | type   | possible_keys             | key     | key_len | ref                  | rows   | Extra                     |
+------+-------------+-------+--------+---------------------------+---------+---------+----------------------+--------+---------------------------+
|    1 | PRIMARY     | C     | ALL    | PRIMARY                   | NULL    | NULL    | NULL                 | 203998 |                           |
|    1 | PRIMARY     | P     | ref    | PRIMARY,fk_Passing_1,reg  | PRIMARY | 7       | Oblig5.C.regno       |     13 | Using where               |
|    1 | PRIMARY     | T     | eq_ref | PRIMARY                   | PRIMARY | 2       | Oblig5.P.tollstation |      1 | Using where; FirstMatch(C) |
+------+-------------+-------+--------+---------------------------+---------+---------+----------------------+--------+---------------------------+
3 rows in set (0.001 sec)
```

Here we see there being a null under key. This is an indication that perhaps We need an index.

I don't belive much to be gained through denormalization but perhaps having Station name in Passing would help.