This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

Assignment 7

Task 1: Normal forms

1. Is this schma in 1NF?
    a. A schema is 1NF if and only if all underlying domains have atomic values. All data is atomic if we assume "timestamp" is atomic.
2. Is it in 2NF?
    a. A schema is 2NF if and only if it is 1NF and all non-key attributes are associated with a candidate key or another non-key attribute. It can be argued that "Passing" does not follow this rule, as "TollStation" does not depend on either "regno" or "timestamp".
3. Is it in 3NF?
    a. If you assume "station" is linked to "timestamp" and "regno", it is. Otherwise, it is not since it does not fill 2NF.

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

Task 2: Create a physical schema and a test environment

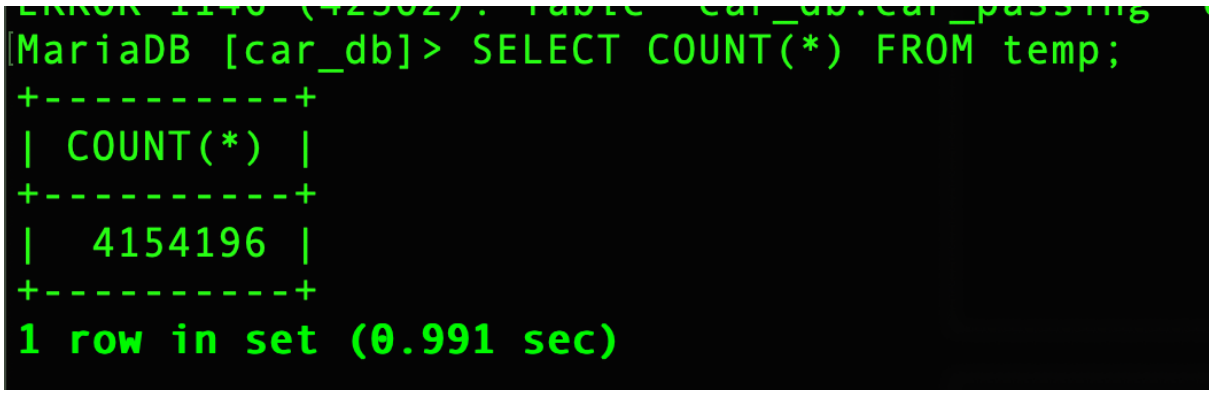
Installed mysqltuner
Sudo dnf install mysqltuner -y
Signed into mariadb
Mariadb -u odnerindheim -p
Created database and use it.
CREATE DATABASE car_db;
USE car_db
Creating aux table for data

```
CREATE TABLE `temp` (
  `passing_datetime` datetime DEFAULT NULL,
  `regno` char(7) DEFAULT NULL,
  `tollstation_id` smallint(5) unsigned DEFAULT NULL,
  `tollstation_name` varchar(85) DEFAULT NULL,
  `owner_name` varchar(85) DEFAULT NULL,
  `taxclass_id` smallint(5) unsigned DEFAULT NULL,
  `taxclass_description` text DEFAULT NULL,
  `has_subscription` enum('yes','no') DEFAULT NULL,
  `fee` decimal(5,2) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci
```

Downloading the data file:
Wget https://eple.hvl.no/fag/dat151/v2024/carpassingdb.txt
Load the data file:
Used DBEAVER to load all the data.

Run some queries:

SELECT COUNT(*) FROM car_passing aux;



SELECT * FROM car_passing_aux ORDER BY passing_timestamp ASC LIMIT 20;

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

```
MariaDB [car_db]> SELECT * FROM temp ORDER BY passing_datetime ASC LIMIT 20;
+---------------------+---------+---------------+---------------------+--------------------+------------+--------------------------+------------------+-------+
| passing_datetime    | regno   | tollstation_id | tollstation_name    | owner_name         | taxclass_id | taxclass_description     | has_subscription | fee   |
+---------------------+---------+---------------+---------------------+--------------------+------------+--------------------------+------------------+-------+
| 2022-08-16 03:48:04 | SR51455 |            19 | Strandkaien         | Magnar Kristiansen |          2 | Ladbar hybrid, taxgroup 1 | yes              | 20.00 |
| 2022-08-19 20:35:04 | SR51455 |            21 | Sotraveien          | Magnar Kristiansen |          2 | Ladbar hybrid, taxgroup 1 | yes              | 20.00 |
| 2022-08-23 23:07:04 | SR51455 |             7 | Straume bro         | Magnar Kristiansen |          2 | Ladbar hybrid, taxgroup 1 | yes              | 20.00 |
| 2022-08-29 01:49:04 | SR51455 |            23 | Straumeveien        | Magnar Kristiansen |          2 | Ladbar hybrid, taxgroup 1 | yes              | 20.00 |
| 2022-08-31 16:02:59 | SR34676 |            24 | Hardangervegen      | Agathe Sæbø        |          3 | Ladbar hybrid, taxgroup 2 | no               | 37.00 |
| 2022-09-04 00:22:04 | SR51455 |            13 | Michael Krohnsgate  | Magnar Kristiansen |          2 | Ladbar hybrid, taxgroup 1 | yes              | 20.00 |
| 2022-09-04 19:17:59 | SR34676 |            23 | Straumeveien        | Agathe Sæbø        |          3 | Ladbar hybrid, taxgroup 2 | no               | 37.00 |
| 2022-09-08 11:01:04 | SR51455 |            28 | Flyplassvegen       | Magnar Kristiansen |          2 | Ladbar hybrid, taxgroup 1 | yes              | 20.00 |
| 2022-09-09 05:23:59 | SR34676 |            17 | Åsamyrane           | Agathe Sæbø        |          3 | Ladbar hybrid, taxgroup 2 | no               | 37.00 |
| 2022-09-09 21:43:46 | EV65954 |            17 | Åsamyrane           | Ekaterina Ismail   |          5 | Elbil, taxgroup 1         | yes              |  8.00 |
| 2022-09-10 04:40:46 | EV65954 |            10 | Florida             | Ekaterina Ismail   |          5 | Elbil, taxgroup 1         | yes              |  8.00 |
| 2022-09-12 09:45:04 | SR51455 |            15 | Tellevikvegen       | Magnar Kristiansen |          2 | Ladbar hybrid, taxgroup 1 | yes              | 20.00 |
| 2022-09-13 14:48:04 | SR51455 |            11 | Gyldenpris          | Magnar Kristiansen |          2 | Ladbar hybrid, taxgroup 1 | yes              | 20.00 |
| 2022-09-14 00:38:17 | EB36167 |            23 | Straumeveien        | Yumiko Hvidsten    |          5 | Elbil, taxgroup 1         | no               | 10.00 |
| 2022-09-14 05:26:22 | SP61094 |            23 | Straumeveien        | Sandra Ødegård     |          2 | Ladbar hybrid, taxgroup 1 | no               | 25.00 |
| 2022-09-14 07:58:17 | EB36167 |            25 | Fritz C. Riebers vei | Yumiko Hvidsten    |          5 | Elbil, taxgroup 1         | no               | 10.00 |
| 2022-09-14 08:17:59 | SR34676 |            10 | Florida             | Agathe Sæbø        |          3 | Ladbar hybrid, taxgroup 2 | no               | 37.00 |
| 2022-09-14 12:34:59 | SR34676 |            12 | Kalfaret            | Agathe Sæbø        |          3 | Ladbar hybrid, taxgroup 2 | no               | 37.00 |
| 2022-09-14 12:46:22 | SP61094 |            25 | Fritz C. Riebers vei | Sandra Ødegård     |          2 | Ladbar hybrid, taxgroup 1 | no               | 25.00 |
| 2022-09-14 13:42:46 | EV65954 |            29 | Skagevegen          | Ekaterina Ismail   |          5 | Elbil, taxgroup 1         | yes              |  8.00 |
+---------------------+---------+---------------+---------------------+--------------------+------------+--------------------------+------------------+-------+
```

SELECT * FROM car_passing_aux ORDER BY passing_timestamp DESC LIMIT 20;

```
MariaDB [car_db]> SELECT * FROM temp ORDER BY passing_datetime DESC LIMIT 20;
+---------------------+---------+---------------+---------------------+--------------------+------------+--------------------------+------------------+-------+
| passing_datetime    | regno   | tollstation_id | tollstation_name    | owner_name         | taxclass_id | taxclass_description     | has_subscription | fee   |
+---------------------+---------+---------------+---------------------+--------------------+------------+--------------------------+------------------+-------+
| 2024-02-08 20:09:47 | EV45094 |            19 | Strandkaien         | Stephanie Ravndal  |          5 | Elbil, taxgroup 1         | yes              |  8.00 |
| 2024-02-08 20:05:39 | SR58844 |            15 | Tellevikvegen       | Xhevrije Hoem      |          4 | Diesel, taxgroup 1        | yes              | 24.00 |
| 2024-02-08 20:00:19 | SR24250 |            13 | Michael Krohnsgate  | Lillan Holen       |          4 | Diesel, taxgroup 1        | no               | 30.00 |
| 2024-02-08 19:56:12 | EV19636 |            12 | Kalfaret            | Zhian Osmundsen    |          5 | Elbil, taxgroup 1         | yes              |  8.00 |
| 2024-02-08 19:56:08 | SP65600 |            28 | Flyplassvegen       | Juni Godø          |          1 | Bensin, taxgroup 1        | no               | 25.00 |
| 2024-02-08 19:54:23 | EB27313 |             6 | Gravdal             | Truls Fjellheim    |          5 | Elbil, taxgroup 1         | yes              |  8.00 |
| 2024-02-08 19:53:04 | ED57003 |            24 | Hardangervegen      | Ted Moen           |          5 | Elbil, taxgroup 1         | yes              |  8.00 |
| 2024-02-08 19:52:59 | SP58784 |            14 | Damsgårdsveien      | Qi Fremstad        |          2 | Ladbar hybrid, taxgroup 1 | no               | 25.00 |
| 2024-02-08 19:49:11 | EF50375 |            18 | Åsaneveien          | Marlene Sævik      |          5 | Elbil, taxgroup 1         | no               | 10.00 |
| 2024-02-08 19:46:45 | EB93110 |            16 | Arnavegen           | Joar Øie           |          6 | Elbil, taxgroup 2         | yes              |  0.00 |
| 2024-02-08 19:46:21 | EV87230 |            20 | Småstrandgaten      | Wojciech Henningsen |          5 | Elbil, taxgroup 1         | no               | 10.00 |
| 2024-02-08 19:45:28 | EV36808 |            11 | Gyldenpris          | Yenny Kallevik     |          5 | Elbil, taxgroup 1         | yes              |  8.00 |
| 2024-02-08 19:45:03 | EC87082 |             1 | Fjøsangerveien      | Queen Steinsvik    |          5 | Elbil, taxgroup 1         | no               | 10.00 |
| 2024-02-08 19:42:20 | SX54186 |             4 | Dolviken            | Zanib Grimstad     |          2 | Ladbar hybrid, taxgroup 1 | no               | 25.00 |
| 2024-02-08 19:41:40 | EB68077 |            29 | Skagevegen          | Eilev Jonassen     |          5 | Elbil, taxgroup 1         | yes              |  8.00 |
| 2024-02-08 19:38:58 | EB62673 |            19 | Strandkaien         | Ulrica Voll        |          5 | Elbil, taxgroup 1         | no               | 10.00 |
| 2024-02-08 19:38:31 | SP58647 |            10 | Florida             | Cassandra Førland  |          2 | Ladbar hybrid, taxgroup 1 | yes              | 20.00 |
| 2024-02-08 19:37:13 | ST42466 |             7 | Straume bro         | Valentina Ask      |          4 | Diesel, taxgroup 1        | yes              | 24.00 |
| 2024-02-08 19:37:10 | SR60544 |             7 | Straume bro         | Zora Voll          |          2 | Ladbar hybrid, taxgroup 1 | no               | 25.00 |
| 2024-02-08 19:35:42 | SP40238 |            21 | Sotraveien          | Quyen Reistad      |          2 | Ladbar hybrid, taxgroup 1 | no               | 25.00 |
+---------------------+---------+---------------+---------------------+--------------------+------------+--------------------------+------------------+-------+
```

Run the mysql tuner
Mysqltuner

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

```
-------- InnoDB Metrics -------------------------------------------------------------------
[--] InnoDB is enabled.
[--] InnoDB Thread Concurrency: 0
[OK] InnoDB File per table is activated
[!!] InnoDB buffer pool / data size: 128.0M/468.0M
[!!] Ratio InnoDB log file size / InnoDB Buffer pool size (75 %): 96.0M * 1/128.0M should be equal to 25%
[--] Number of InnoDB Buffer Pool Chunk : 1 for 1 Buffer Pool Instance(s)
[OK] Innodb_buffer_pool_size aligned with Innodb_buffer_pool_chunk_size & Innodb_buffer_pool_instances
[OK] InnoDB Read buffer efficiency: 99.99% (46894525 hits/ 46900422 total)
[OK] InnoDB Write log efficiency: 99.24% (58893 hits/ 59344 total)
[OK] InnoDB log waits: 0.00% (0 waits / 451 writes)

-------- Aria Metrics ---------------------------------------------------------------------
[--] Aria Storage Engine is enabled.
[OK] Aria pagecache size / total Aria indexes: 128.0M/320.0K
[!!] Aria pagecache hit rate: 52.0% (25 cached / 12 reads)

-------- TokuDB Metrics -------------------------------------------------------------------
[--] TokuDB is disabled.

-------- XtraDB Metrics -------------------------------------------------------------------
[--] XtraDB is disabled.

-------- Galera Metrics -------------------------------------------------------------------
[--] Galera is disabled.

-------- Replication Metrics --------------------------------------------------------------
[--] Galera Synchronous replication: NO
[--] No replication slave(s) for this server.
[--] Binlog format: MIXED
[--] XA support enabled: ON
[--] Semi synchronous replication Master: OFF
[--] Semi synchronous replication Slave: OFF
[--] This is a standalone server

-------- Recommendations ------------------------------------------------------------------
General recommendations:
    Reduce or eliminate unclosed connections and network issues
    Configure your accounts with ip or subnets only, then update your configuration with skip-name-resolve=1
    Performance schema should be activated for better diagnostics
    Consider installing Sys schema from https://github.com/mysql/mysql-sys for MySQL
    Consider installing Sys schema from https://github.com/FromDual/mariadb-sys for MariaDB
    Before changing innodb_log_file_size and/or innodb_log_files_in_group read this: https://bit.ly/2TcGgtU
Variables to adjust:
    performance_schema = ON enable PFS
    innodb_buffer_pool_size (>= 468.0M) if possible.
    innodb_log_file_size should be (=32M) if possible, so InnoDB total log files size equals to 25% of buffer pool size.
[odnerindheim@localhost ~]$ mysqltuner
 >>  MySQLTuner 1.8.3 - Major Hayden <major@mhtx.net>
 >>  Bug reports, feature requests, and downloads at http://mysqltuner.pl/
 >>  Run with '--help' for additional options and output filtering
```

Increading the buffer_pool_size:
Using SET GLOBAL as this doesn't require a restart: [MariaDB Enterprise Server Configure the InnoDB Buffer Pool — MariaDB Documentation]

SET GLOBAL innodb_buffer_pool_size=(2*1024*1024*1024);
SHOW GLOBAL STATUS
LIKE 'Innodb_buffer_pool_resize_status';

```
MariaDB [(none)]> SHOW GLOBAL STATUS
    -> LIKE 'Innodb_buffer_pool_resize_status';
+----------------------------------+----------------------------------------------------+
| Variable_name                    | Value                                              |
+----------------------------------+----------------------------------------------------+
| Innodb_buffer_pool_resize_status | Completed resizing buffer pool at 240419 16:38:43. |
+----------------------------------+----------------------------------------------------+
1 row in set (0.000 sec)
```

Create normalized tables:
-- car_db.TaxClass definition

```
CREATE TABLE `TaxClass` (
  `taxclass` smallint(5) unsigned NOT NULL AUTO_INCREMENT,
  `description` text NOT NULL,
  PRIMARY KEY (`taxclass`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;
```

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

```
-- car_db.Tollstation definition

CREATE TABLE `Tollstation` (
  `tollstation` smallint(5) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(85) NOT NULL,
  PRIMARY KEY (`tollstation`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;


-- car_db.Car definition

CREATE TABLE `Car` (
  `regno` char(7) NOT NULL,
  `owner` varchar(85) NOT NULL,
  `taxclass` smallint(5) unsigned NOT NULL,
  PRIMARY KEY (`regno`),
  KEY `taxclass` (`taxclass`),
  CONSTRAINT `Car_ibfk_1` FOREIGN KEY (`taxclass`) REFERENCES `TaxClass`
(`taxclass`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;


-- car_db.Fee definition

CREATE TABLE `Fee` (
  `fee` smallint(5) unsigned NOT NULL AUTO_INCREMENT,
  `taxclass` smallint(5) unsigned NOT NULL,
  `type` enum('regular','withsubscription') DEFAULT NULL,
  `costPerPassing` decimal(5,2) DEFAULT NULL,
  PRIMARY KEY (`fee`),
  UNIQUE KEY `taxclass` (`taxclass`,`type`),
  CONSTRAINT `Fee_ibfk_1` FOREIGN KEY (`taxclass`) REFERENCES `TaxClass`
(`taxclass`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;


-- car_db.Passing definition

CREATE TABLE `Passing` (
  `timestamp` datetime NOT NULL,
  `regno` char(7) NOT NULL,
  `tollstation` smallint(5) unsigned NOT NULL,
  PRIMARY KEY (`timestamp`,`regno`),
  KEY `tollstation` (`tollstation`),
  KEY `regno` (`regno`),
```

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

```
  CONSTRAINT `Passing_ibfk_1` FOREIGN KEY (`tollstation`) REFERENCES
`Tollstation` (`tollstation`),
  CONSTRAINT `Passing_ibfk_2` FOREIGN KEY (`regno`) REFERENCES `Car`
(`regno`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;


-- car_db.Subscription definition

CREATE TABLE `Subscription` (
  `regno` char(7) NOT NULL,
  PRIMARY KEY (`regno`),
  CONSTRAINT `Subscription_ibfk_1` FOREIGN KEY (`regno`) REFERENCES `Car`
(`regno`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;
```

Populate tables:
Insert into Tollstation (tollstation, name) select distinct tollstation_id, tollstation_name from temp;

Insert into TaxClass (taxclass, desciprtion) select distinct taxclass_id, taxclass_description from temp;

Insert into Car (regno, owner, taxclass) select distinct regno, owner_name, taxclass_id from temp;

Insert into Subscription select distinct regno from temp where has_subscription='yes';

Insert into passing (timestamp,regno,tollstation) select distinct passing_datetime, regno, tollstation_id from temp;

Insert into fee(taxclass,type,costperpassing) select distinct taxclass_id,'regular',fee from temp where has_subscription='no';

Insert into fee(taxclass,type,costperpassing) select distinct taxclass_id,'withsubscription',fee from temp where has_subscription='yes';

Odne Rindheim Redone Task 2 and Task 3

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

Verifing data integrity:

```
SELECT * FROM Tollstation LIMIT 10;
SELECT * FROM TaxClass LIMIT 10;
SELECT * FROM Car LIMIT 10;
SELECT * FROM Subscription LIMIT 10;
SELECT * FROM Passing LIMIT 10;
SELECT * FROM Fee LIMIT 10;
```

All data looks in order!

Dropping temp

```
DROP TABLE temp;
```

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

Task 3: Performance of the physical model

Cache was cleared and mariadb was restarted before each query.

Query 1:

A)
Finds the owner name of passings of a certain time and date.

```
SELECT C.owner, P.timestamp
FROM Car C JOIN Passing P USING(regno)
WHERE YEAR(P.timestamp)=2023 AND MONTH(P.timestamp)=2
AND DAYOFMONTH(P.timestamp)=1 AND HOUR(P.timestamp)=23;
```

B)
SET profiling = 1;
Running the query 3 times.

```
MariaDB [car_db]> SHOW PROFILES;
+----------+------------+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| Query_ID | Duration   | Query                                                                                                                                                                                  |
+----------+------------+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
|        1 | 1.02631599 | SELECT C.owner, P.timestamp FROM Car C JOIN Passing P USING (regno) WHERE YEAR(P.timestamp)=2023 AND MONTH(P.timestamp)=2 AND DAYOFMONTH(P.timestamp)=1 AND HOUR(P.timestamp)=23 |
|        2 | 1.01596361 | SELECT C.owner, P.timestamp FROM Car C JOIN Passing P USING (regno) WHERE YEAR(P.timestamp)=2023 AND MONTH(P.timestamp)=2 AND DAYOFMONTH(P.timestamp)=1 AND HOUR(P.timestamp)=23 |
|        3 | 1.02022030 | SELECT C.owner, P.timestamp FROM Car C JOIN Passing P USING (regno) WHERE YEAR(P.timestamp)=2023 AND MONTH(P.timestamp)=2 AND DAYOFMONTH(P.timestamp)=1 AND HOUR(P.timestamp)=23 |
+----------+------------+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

MariaDB [car_db]> EXPLAIN SELECT C.owner, P.timestamp FROM Car C JOIN Passing P USING (regno) WHERE
YEAR(P.timestamp)=2023 AND MONTH(P.timestamp)=2 AND DAYOFMONTH(P.timestamp)=1 AND HOUR(P.timestamp)=23;
+------+-------------+-------+------+---------------+-------+---------+---------------+-------+--------------------------+
| id   | select_type | table | type | possible_keys | key   | key_len | ref           | rows  | Extra                    |
+------+-------------+-------+------+---------------+-------+---------+---------------+-------+--------------------------+
|    1 | SIMPLE      | C     | ALL  | PRIMARY       | NULL  | NULL    | NULL          | 62896 |                          |
|    1 | SIMPLE      | P     | ref  | regno         | regno | 17      | car_db.C.regno | 31    | Using where; Using index |
+------+-------------+-------+------+---------------+-------+---------+---------------+-------+--------------------------+
2 rows in set (0.000 sec)
```

C)

Denormalisating Passing:
I made an attempt to denormalize passing table, to avoid using function calls like YEAR(), MONTH(), DAYOFMONTH(), HOUR(), which can prevent the use of index:

```
CREATE TABLE Passing_denorm AS
SELECT P.*,
    YEAR(P.timestamp) AS year,
    MONTH(P.timestamp) AS month,
    DAYOFMONTH(P.timestamp) AS day,
    HOUR(P.timestamp) AS hour
```

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

FROM Passing P;

new query:
SELECT C.owner, P.timestamp
FROM Car C JOIN Passing_denorm P USING(regno)
WHERE P.year = 2023 AND P.month = 2
AND P.day = 1 AND P.hour = 23;

Profiling results (mariadb was restarted and cache was cleared):

```
MariaDB [car_db]> SHOW PROFILES;
+----------+------------+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| Query_ID | Duration   | Query                                                                                                                                                                      |
+----------+------------+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
|        1 | 1.35718505 | SELECT C.owner, P.timestamp FROM Car C JOIN Passing_denorm P USING(regno) WHERE P.year = 2023 AND P.month = 2 AND P.day = 1 AND P.hour = 23 |
|        2 | 1.36653832 | SELECT C.owner, P.timestamp FROM Car C JOIN Passing_denorm P USING(regno) WHERE P.year = 2023 AND P.month = 2 AND P.day = 1 AND P.hour = 23 |
|        3 | 1.36108667 | SELECT C.owner, P.timestamp FROM Car C JOIN Passing_denorm P USING(regno) WHERE P.year = 2023 AND P.month = 2 AND P.day = 1 AND P.hour = 23 |
+----------+------------+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
3 rows in set (0.000 sec)
```

Not exactly better results, trying explain:

```
MariaDB [car_db]> EXPLAIN SELECT C.owner, P.timestamp FROM Car C JOIN Passing_denorm P USING(regno) WHERE P.year = 2023 AND P.month = 2 AND P.day = 1 AND P.hour = 23;
+------+-------------+-------+--------+---------------+---------+---------+----------------+---------+-------------+
| id   | select_type | table | type   | possible_keys | key     | key_len | ref            | rows    | Extra       |
+------+-------------+-------+--------+---------------+---------+---------+----------------+---------+-------------+
|    1 | SIMPLE      | P     | ALL    | NULL          | NULL    | NULL    | NULL           | 4138561 | Using where |
|    1 | SIMPLE      | C     | eq_ref | PRIMARY       | PRIMARY | 17      | car_db.P.regno | 1       |             |
+------+-------------+-------+--------+---------------+---------+---------+----------------+---------+-------------+
2 rows in set (0.000 sec)
```

This denormalization removed the use of index.
This denormalization violates the 3NF as it introduces transitive dependencies – each one of these date parts is non-prime and is determined by 'timestamp', which is not part of the primary key.

I can try denormalizing by adding a owner column to passing:
This denormalization also violates normalization rule of storing non-key attributes (attributes that do not contribute to a composite primary key), in this case 'owner', which is related to car and not directly to the passing event.

```
CREATE TABLE Passing_with_owner AS
SELECT P.timestamp, P.regno, P.tollstation, C.owner
FROM Passing P
JOIN Car C ON P.regno = C.regno;
```

Optimized query:

```
SELECT owner, timestamp
FROM Passing_with_owner
WHERE YEAR(timestamp) = 2023 AND MONTH(timestamp) = 2
```

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

AND DAYOFMONTH(timestamp) = 1 AND HOUR(timestamp) = 23;

Profile:

```
MariaDB [car_db]> SHOW PROFILES;
+----------+------------+-----------------------------------------------------------------------------------------------------------------------------------------------
----------+
| Query_ID | Duration   | Query                                                                                              |
+----------+------------+-----------------------------------------------------------------------------------------------------------------------------------------------
----------+
|        1 | 1.33344789 | SELECT owner,timestamp FROM Passing_with_owner WHERE YEAR(timestamp)=2023 AND
MONTH(timestamp)=2 AND DAYOFMONTH(timestamp)=1 AND HOUR(timestamp)=23 |
|        2 | 1.32670961 | SELECT owner,timestamp FROM Passing_with_owner WHERE YEAR(timestamp)=2023 AND
MONTH(timestamp)=2 AND DAYOFMONTH(timestamp)=1 AND HOUR(timestamp)=23 |
|        3 | 1.32485803 | SELECT owner,timestamp FROM Passing_with_owner WHERE YEAR(timestamp)=2023 AND
MONTH(timestamp)=2 AND DAYOFMONTH(timestamp)=1 AND HOUR(timestamp)=23 |
+----------+------------+-----------------------------------------------------------------------------------------------------------------------------------------------
----------+
3 rows in set (0.000 sec)

MariaDB [car_db]> EXPLAIN SELECT owner,timestamp FROM Passing_with_owner WHERE YEAR(timestamp)=2023 AND
MONTH(timestamp)=2 AND DAYOFMONTH(timestamp)=1 AND HOUR(timestamp)=23;
+------+-------------+--------------------+------+---------------+------+---------+------+---------+-------------+
| id   | select_type | table              | type | possible_keys | key  | key_len | ref  | rows    | Extra       |
+------+-------------+--------------------+------+---------------+------+---------+------+---------+-------------+
|    1 | SIMPLE      | Passing_with_owner | ALL  | NULL          | NULL | NULL    | NULL | 3381507 | Using where |
+------+-------------+--------------------+------+---------------+------+---------+------+---------+-------------+
1 row in set (0.000 sec)
```

D)

Could I make the dbms use index in this query? To make the dbms more likely to use an index, I could change the query to use a range-based function instead of functions, heres how I could create an index, and use it:

CREATE INDEX idx_passing_owner_timestamp ON Passing_with_owner(timestamp);

SELECT owner, timestamp
FROM Passing_with_owner
WHERE timestamp >= '2023-02-01 23:00:00' AND timestamp < '2023-02-02 00:00:00';

```
MariaDB [car_db]> EXPLAIN SELECT owner, timestamp FROM Passing_with_owner WHERE timestamp >= '2023-02-01 23:00:00'
AND timestamp < '2023-02-02 00:00:00
';
+------+-------------+--------------------+-------+-----------------------------+-----------------------------+---------+------+------+-----------------------+
| id   | select_type | table              | type  | possible_keys               | key                         | key_len | ref  | rows | Extra                 |
+------+-------------+--------------------+-------+-----------------------------+-----------------------------+---------+------+------+-----------------------+
|    1 | SIMPLE      | Passing_with_owner | range | idx_passing_owner_timestamp | idx_passing_owner_timestamp | 5       | NULL | 151  |
Using index condition |
+------+-------------+--------------------+-------+-----------------------------+-----------------------------+---------+------+------+-----------------------+
1 row in set (0.001 sec)
```

Creating an index made the Select function very fast:

CREATE INDEX idx_passing_with_owner ON Passing_with_owner(timestamp, owner);

```
MariaDB [car_db]> EXPLAIN SELECT owner,timestamp FROM Passing_with_owner WHERE timestamp >= '2023-02-01 23:00:00' AND
timestamp <'2023-02-02 00:00:00';
+------+-------------+--------------------+-------+----------------------------------------------------+------------------------+---------+------+------+-------
-------------------+
| id   | select_type | table              | type  | possible_keys                                      | key                    | key_len | ref  | rows | Extra                    |
```

Odne Rindheim Redone Task 2 and Task 3

This probably didn’t include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

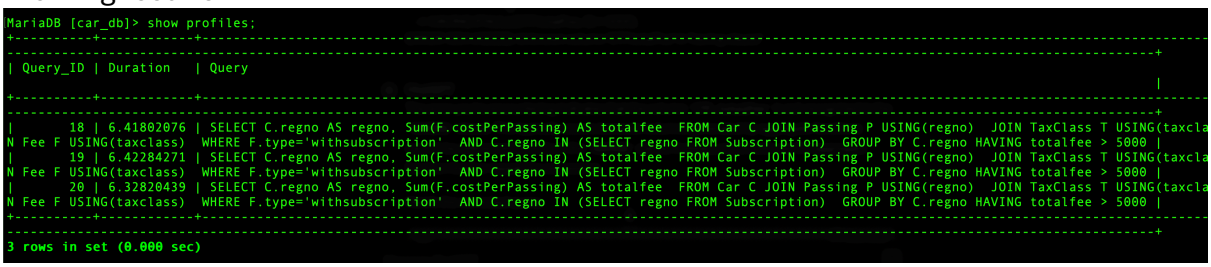
```
+------+-------------+--------------------+-------+-----------------------------------------------------+------------------------+---------+------+------+-------------------------+
|   1 | SIMPLE     | Passing_with_owner | range | idx_passing_owner_timestamp,idx_passing_with_owner | idx_passing_with_owner | 5
| NULL | 151  | Using where; Using index |
+------+-------------+--------------------+-------+-----------------------------------------------------+------------------------+---------+------+------+-------------------------+
1 row in set (0.000 sec)
MariaDB [car_db]> SHOW PROFILES;
+----------+------------+-------------------------------------------------------------------------------------------------------------------------------------------------+
| Query_ID | Duration   | Query                                                                                                                                           |
+----------+------------+-------------------------------------------------------------------------------------------------------------------------------------------------+
|       1 | 0.00244040 | SELECT owner,timestamp FROM Passing_with_owner WHERE timestamp >= '2023-02-01 23:00:00' AND
timestamp <'2023-02-02 00:00:00' |
|       2 | 0.00038712 | SELECT owner,timestamp FROM Passing_with_owner WHERE timestamp >= '2023-02-01 23:00:00' AND
timestamp <'2023-02-02 00:00:00' |
|       3 | 0.00038903 | SELECT owner,timestamp FROM Passing_with_owner WHERE timestamp >= '2023-02-01 23:00:00' AND
timestamp <'2023-02-02 00:00:00' |
+----------+------------+-------------------------------------------------------------------------------------------------------------------------------------------------+
3 rows in set (0.000 sec)
```

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

Query 2:

A) Retrieves the registration numbers of cars that have a subscription and calculates the total fee for each car, where the total fee is more than 5000.

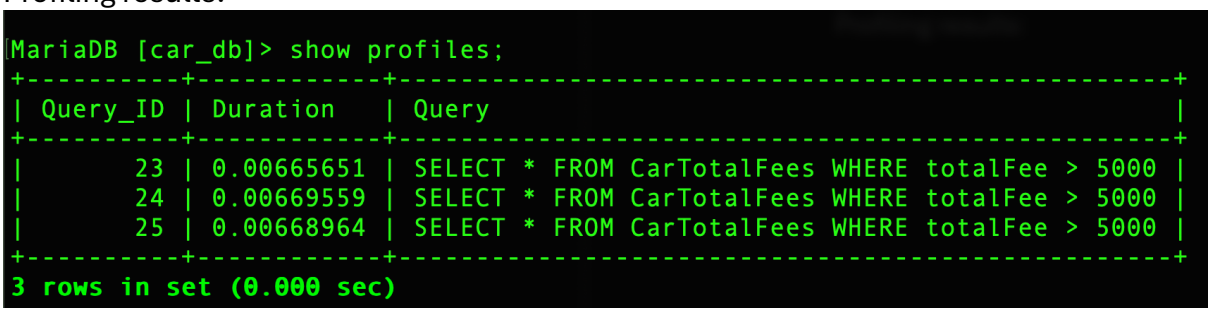
B) Profiling results:



C) Can create a table that stores the pre-calculated sums of fees for cars with subscriptions. This way, we avoid the need to calculate the sum dynamically each time the query is executed. Since both attributes are atomic values, and there are no partial dependencies it is 1NF and 2NF, but since totalfee attribute is transitively dependent on the car tables attirbutes, it violates 3NF
Create Denormalized Table:

```
CREATE TABLE CarTotalFees (
regno CHAR(7) PRIMARY KEY,
totalFee DECIMAL(10,2)
);

INSERT INTO CarTotalFees (regno, totalFee)
SELECT C.regno, SUM(F.costPerPassing) AS totalfee
FROM Car C
JOIN Passing P USING (regno)
JOIN Fee F USING (taxclass)
JOIN Subscription S USING (regno)
WHERE F.type = 'withsubscription'
GROUP BY C.regno;

SELECT *
FROM CarTotalFees
WHERE totalFee > 5000;
```

Profiling results:



Odne Rindheim Redone Task 2 and Task 3

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

D) Explain the original query:

```
MariaDB [car_db]> EXPLAIN SELECT C.regno AS regno, Sum(F.costPerPassing) AS totalfee  FROM Car C JOIN Passing P USING(regno)  JOIN TaxClass T USING(taxclass)  JOIN Fee
+------+-------------+--------------+--------+------------------+----------+---------+--------------------------+------+--------------------------------+
| id   | select_type | table        | type   | possible_keys    | key      | key_len | ref                      | rows | Extra                          |
+------+-------------+--------------+--------+------------------+----------+---------+--------------------------+------+--------------------------------+
|    1 | PRIMARY     | T            | ALL    | PRIMARY          | NULL     | NULL    | NULL                     | 10   | Using temporary; Using filesort |
|    1 | PRIMARY     | F            | eq_ref | taxclass         | taxclass | 4       | car_db.T.taxclass,const  | 1    | Using index condition          |
|    1 | PRIMARY     | C            | ref    | PRIMARY,taxclass | taxclass | 2       | car_db.T.taxclass        | 1    | Using index                    |
|    1 | PRIMARY     | Subscription | eq_ref | PRIMARY          | PRIMARY  | 21      | car_db.C.regno           | 1    | Using index                    |
|    1 | PRIMARY     | P            | ref    | regno            | regno    | 21      | car_db.C.regno           | 1    |                                |
+------+-------------+--------------+--------+------------------+----------+---------+--------------------------+------+--------------------------------+
```

E) Created some indexes, to see if it could speed up original qeuyies:
CREATE INDEX idx_regno ON Car (regno);
CREATE INDEX idx_type ON Fee (type, taxclass, costPerPassing);
CREATE INDEX idx_regno_subscription ON Subscription (regno);

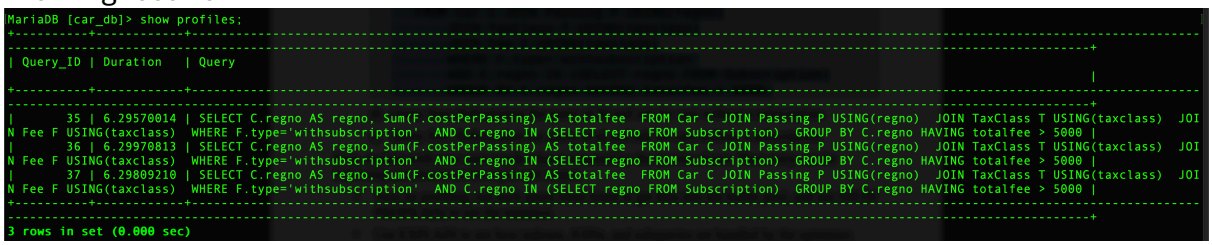
```
MariaDB [car_db]> SHOW PROFILES;
+----------+------------+-----------------------------------------------------------------------
| Query_ID | Duration   | Query
+----------+------------+-----------------------------------------------------------------------
|       28 | 6.31994821 | SELECT C.regno AS regno, Sum(F.costPerPassing) AS totalfee  FROM Car C JOIN Passing P USING(regno)  JOIN TaxClass T USING(taxclass)  JOIN Fee F USING(taxclass)  WHERE F.type='withsubscription'  AND C.regno IN (SELECT regno FROM Subscription)  GROUP BY C.regno HAVING totalfee > 5000 |
|       29 | 6.31393725 | SELECT C.regno AS regno, Sum(F.costPerPassing) AS totalfee  FROM Car C JOIN Passing P USING(regno)  JOIN TaxClass T USING(taxclass)  JOIN Fee F USING(taxclass)  WHERE F.type='withsubscription'  AND C.regno IN (SELECT regno FROM Subscription)  GROUP BY C.regno HAVING totalfee > 5000 |
|       30 | 6.32113335 | SELECT C.regno AS regno, Sum(F.costPerPassing) AS totalfee  FROM Car C JOIN Passing P USING(regno)  JOIN TaxClass T USING(taxclass)  JOIN Fee F USING(taxclass)  WHERE F.type='withsubscription'  AND C.regno IN (SELECT regno FROM Subscription)  GROUP BY C.regno HAVING totalfee > 5000 |
+----------+------------+-----------------------------------------------------------------------

MariaDB [car_db]> EXPLAIN SELECT C.regno AS regno, Sum(F.costPerPassing) AS totalfee  FROM Car C JOIN Passing P USING(regno)  JOIN TaxClass T USING(taxclass)  JOIN Fee F USING(taxclass)  WHERE F.type='withsubscription'  AND C.regno IN (SELECT regno FROM Subscription)  GROUP BY C.regno HAVING totalfee > 5000;
+------+-------------+--------------+--------+---------------------------------+----------+---------+-------------------+------+-------------------------------------------------------
| id   | select_type | table        | type   | possible_keys                   | key      | key_len | ref               | rows | Extra                                                 |
+------+-------------+--------------+--------+---------------------------------+----------+---------+-------------------+------+-------------------------------------------------------
|    1 | PRIMARY     | F            | ref    | taxclass,idx_type               | idx_type | 2       | const             | 10   | Using where; Using index; Using temporary; Using filesort |
|    1 | PRIMARY     | T            | eq_ref | PRIMARY                         | PRIMARY  | 2       | car_db.F.taxclass | 1    |                                                       |
|    1 | PRIMARY     | C            | ref    | PRIMARY,taxclass,idx_regno      | taxclass | 2       | car_db.F.taxclass | 3593 | Using index                                           |
|    1 | PRIMARY     | Subscription | eq_ref | PRIMARY,idx_regno_subscription  | PRIMARY  | 21      | car_db.C.regno    | 1    | Using index                                           |
|    1 | PRIMARY     | P            | ref    | regno                           | regno    | 21      | car_db.C.regno    | 1    |                                                       |
+------+-------------+--------------+--------+---------------------------------+----------+---------+-------------------+------+-------------------------------------------------------
5 rows in set (0.001 sec)
```

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

Query 3

A) Like query two, but instead of directly joining with the subscription table, it uses a subquery to find cars with subscriptions.

B) Profiling results:



C) Possible denormalizing the query, very similar to the denormalizing of query 2, this also breaks the 3NF by introducing redundancy.

```
CREATE TABLE query 3 (
regno CHAR(7) PRIMARY KEY,
totalFee DECIMAL (10,2)
);
```

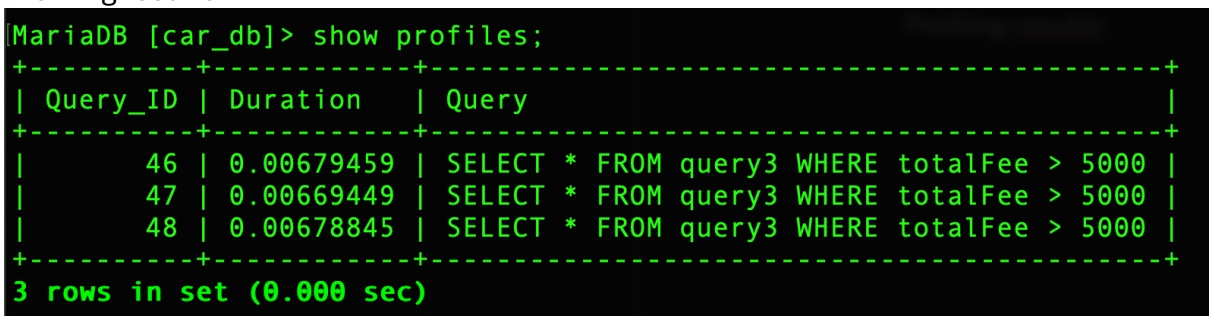
Inserting data into new table:

```
INSERT INTO query3 (regno, totalFee)
  -> SELECT C,regno AS regno, SUM(F.costPerPassing) AS totalfee
  -> FROM Car C JOIN Passing P USING (regno)
  -> JOIN TaxClass T USING(taxclass)
  -> JOIN Fee F USING(taxclass)
  -> WHERE F.type='withsubscription'
  -> AND C.regno IN (SELECT regno FROM Subscription)
  -> GROUP BY C.regno;
```

New query:

```
SELECT *
FROM query3
WHERE totalFee > 5000;
```

Profiling results:



D) Adding an index to optimize the join and where condiiton: CREATE INDEX idx_car_with_subscription ON Car_with_subscription(regno, taxclass, subscription); MariaDB [car_db]> SHOW PROFILES; +----------+------------+---------- ---------- |

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

Query_ID | Duration | Query | | 1 | 0.74527054 | SELECT CWS.regno, SUM(F.costPerPassing) AS totalfee FROM Car_with_subscription CWS JOIN Passing P ON CWS.regno = P.regno JOIN TaxClass T on CWS.taxclass = T.taxclass JOIN Fee F ON T.taxclass = F.taxclass WHERE F.type = 'withsubscription' AND CWS.subscription = 'yes' GROUP BY CWS.regno HAVING totalfe | | 2 | 0.72084794 | SELECT CWS.regno, SUM(F.costPerPassing) AS totalfee FROM Car_with_subscription CWS JOIN Passing P ON CWS.regno = P.regno JOIN TaxClass T on CWS.taxclass = T.taxclass JOIN Fee F ON T.taxclass = F.taxclass WHERE F.type = 'withsubscription' AND CWS.subscription = 'yes' GROUP BY CWS.regno HAVING totalfe | | 3 | 0.71515412 | SELECT CWS.regno, SUM(F.costPerPassing) AS totalfee FROM Car_with_subscription CWS JOIN Passing P ON CWS.regno = P.regno JOIN TaxClass T on CWS.taxclass = T.taxclass JOIN Fee F ON T.taxclass = F.taxclass WHERE F.type = 'withsubscription' AND CWS.subscription = 'yes' GROUP BY CWS.regno HAVING totalfe | +----------+------------+---+ 3 rows in set (0.000 sec) MariaDB [car_db]> EXPLAIN SELECT CWS.regno, SUM(F.costPerPassing) AS totalfee FROM Car_with_subscription CWS JOIN Passing P ON CWS.regno = P.regno JOIN TaxClass T on CWS.taxclass = T.taxclass JOIN Fee F ON T.taxclass = F.taxclass WHERE F.type = 'withsubscription' AND CWS.subscription = 'yes' GROUP BY CWS.regno HAVING totalfee > 5000; +------+-------------+-------+--------+---------------------------+---------------------------+---------+--------------------------+-------+-------------------------+ | id | select_type | table | type | possible_keys | key | key_len | ref | rows | Extra | +------+-------------+-------+--------+---------------------------+---------------------------+---------+--------------------------+-------+-------------------------+ | 1 | SIMPLE | CWS | index | idx_car_with_subscription | idx_car_with_subscription | 24 | NULL | 62566 | Using where; Using index | | 1 | SIMPLE | F | ref | taxclass,idx_fee_type_cost | idx_fee_type_cost | 6 | const,car_db.CWS.taxclass | 1 | Using where; Using index | | 1 | SIMPLE | T | eq_ref | PRIMARY | PRIMARY | 4 | car_db.CWS.taxclass | 1 | Using index | | 1 | SIMPLE | P | ref | regno | regno | 17 | car_db.CWS.regno | 31 | Using index | +------+-------------+-------+--------+---------------------------+---------------------------+---------+--------------------------+-------+-------------------------+ 4 rows in set (0.000 sec)

E)

Replacing the 'IN' subquery with an 'INNER JOIN', which is often more e icient as it can utilize indexes better than a subquery that might result in a full scan. Odne Rindheim This query will allow the DBMS to use indexes e iciently during the join operation instead of possibly running a separate subquery for each row processed in the main query: SELECT C.regno AS regno, SUM(F.costPerPassing) AS totalfee FROM Car C JOIN Passing P USING(regno) JOIN TaxClass T USING(taxclass) JOIN Fee F USING(taxclass) JOIN Subscription S ON C.regno = S.regno WHERE F.type = 'withsubscription' GROUP BY C.regno HAVING totalfee > 5000;

MariaDB [car_db]> EXPLAIN SELECT C.regno AS regno, SUM(F.costPerPassing) AS totalfee FROM Car C JOIN Passing P USING (regno) JOIN TaxClass T USING(taxclass) JOIN Fee F USING(taxclass) JOIN Subscription S on C.regno = S.regno WHERE F.type = 'withsubscription' GROUP BY C.regno HAVING totalfee > 5000; +------+-------------+-------+--------+---------------------------+-------------------+---------+-------------------+------+---+ | id | select_type | table | type | possible_keys | key | key_len | ref | rows | Extra | +------+-------------+-------+--------+---------------------------+-------------------+---------+-------

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

------------+------+--+ | 1 | SIMPLE | F | ref | taxclass,idx_fee_type_cost | idx_fee_type_cost | 1 temporary; Using filesort | | 1 | SIMPLE | T | eq_ref | PRIMARY | PRIMARY | 4 | const | 10 | Using where; Using index; Using | car_db.F.taxclass | 1 | | | | 1 | SIMPLE | C | ref | PRIMARY,taxclass 1 | SIMPLE | S | eq_ref | PRIMARY 1 | SIMPLE | P | ref | regno | regno | taxclass | PRIMARY | 17 | 5 | 17 | car_db.F.taxclass | 3494 | Using index | car_db.C.regno | 1 | Using index | car_db.C.regno | 31 | Using index | | | | +------+-------------+-------+--------+---------------------------+-------------------+---------+-------------------+------+--+ 5 rows in set (0.000 sec)

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

Query 4:

A) Similar to query two, but it asks for cars without a subscription.
Query:
SELECT C.regno AS regno,
 SUM(F.costPerPassing) AS totalfee
FROM Car C
JOIN Passing P USING(regno)
JOIN TaxClass T USING(taxclass)
JOIN Fee F USING(taxclass)
LEFT JOIN Subscription S ON C.regno = S.regno
WHERE F.type = 'withsubscription' AND S.regno IS NULL
GROUP BY C.regno
HAVING totalfee > 5000;

B) Profiling results of original query:

```
MariaDB [car_db]> SHOW PROFILES;
+----------+------------+---------------------------------------------------------------------------------------------------------------
---+
| Query_ID | Duration   | Query

  |
+----------+------------+---------------------------------------------------------------------------------------------------------------
---+
|       51 | 4.33079953 | SELECT C.regno AS regno,          SUM(F.costPerPassing) AS totalfee FROM Car C  JOIN Passing P USING(regno) JOIN TaxClass T USING(taxclas
s) JOIN Fee F USING(taxclass) LEFT JOIN Subscription S ON C.regno = S.regno WHERE F.type = 'withsubscription' AND S.regno IS NULL GROUP BY C.regno HAVING totalfee
 > |
|       52 | 4.34868470 | SELECT C.regno AS regno,          SUM(F.costPerPassing) AS totalfee FROM Car C  JOIN Passing P USING(regno) JOIN TaxClass T USING(taxclas
s) JOIN Fee F USING(taxclass) LEFT JOIN Subscription S ON C.regno = S.regno WHERE F.type = 'withsubscription' AND S.regno IS NULL GROUP BY C.regno HAVING totalfee
 > |
|       53 | 4.31732821 | SELECT C.regno AS regno,          SUM(F.costPerPassing) AS totalfee FROM Car C  JOIN Passing P USING(regno) JOIN TaxClass T USING(taxclas
s) JOIN Fee F USING(taxclass) LEFT JOIN Subscription S ON C.regno = S.regno WHERE F.type = 'withsubscription' AND S.regno IS NULL GROUP BY C.regno HAVING totalfee
 > |
+----------+------------+---------------------------------------------------------------------------------------------------------------
---+
3 rows in set (0.000 sec)
```

C) Denormalisation approach would be the same as query 2 and query 3, just with cars without a subscription
Create table query4 (
regno CHAR(7) PRIMARY KEY,
totalFee DECIMAL(10,2)
);
Insert into query4 (regno, totalFee)
SELECT C.regno, SUM(F.costPerPassing) AS totalfee
FROM Car C
JOIN Passing P USING (regno)
JOING TaxClass T USING(taxclass)
JOIN Fee F USING (taxclass)
LEFT JOIN Subscription S on C.regno = S.regno
WHERE f.type = 'withsubscription' AND S.regno IS NULL
GROUP BY C.regno;

SELECT * FRIN query4 WHERE totalFee > 5000;

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

D) Profiling results:

```
MariaDB [car_db]> show profiles;
+----------+------------+-----------------------------------------------------------------------------+
| Query_ID | Duration   | Query                                                                       |
+----------+------------+-----------------------------------------------------------------------------+
|       55 | 0.00460380 | SELECT * FROM CarTotalFeesWithoutSubscription WHERE totalFee > 5000 |
|       56 | 0.00465601 | SELECT * FROM CarTotalFeesWithoutSubscription WHERE totalFee > 5000 |
|       57 | 0.00461748 | SELECT * FROM CarTotalFeesWithoutSubscription WHERE totalFee > 5000 |
+----------+------------+-----------------------------------------------------------------------------+
3 rows in set (0.000 sec)

MariaDB [car_db]> EXPLAIN SELECT C.regno AS regno,         SUM(F.costPerPassing) AS totalfee FROM Car C  JOIN Passing P USING(regno) JOIN TaxClass T USING(taxclas
s) JOIN Fee F USING(taxclass) LEFT JOIN Subscription S ON C.regno = S.regno WHERE F.type = 'withsubscription' AND S.regno IS NULL GROUP BY C.regno HAVING totalfee
 > 5000;
+------+-------------+-------+--------+-------------------------------+----------+---------+-------------------+------+-----------------------------------------------
----------------+
| id   | select_type | table | type   | possible_keys                 | key      | key_len | ref               | rows | Extra
               |
+------+-------------+-------+--------+-------------------------------+----------+---------+-------------------+------+-----------------------------------------------
----------------+
|    1 | SIMPLE      | F     | ref    | taxclass,idx_type             | idx_type | 2       | const             | 10   | Using where; Using index; Using temporary
; Using filesort |
|    1 | SIMPLE      | T     | eq_ref | PRIMARY                       | PRIMARY  | 2       | car_db.F.taxclass | 1    |
               |
|    1 | SIMPLE      | C     | ref    | PRIMARY,taxclass,idx_regno    | taxclass | 2       | car_db.F.taxclass | 3593 | Using index
               |
|    1 | SIMPLE      | S     | eq_ref | PRIMARY,idx_regno_subscription | PRIMARY  | 21      | car_db.C.regno    | 1    | Using where; Using index; Not exists
               |
|    1 | SIMPLE      | P     | ref    | regno                         | regno    | 21      | car_db.C.regno    | 1    |
               |
+------+-------------+-------+--------+-------------------------------+----------+---------+-------------------+------+-----------------------------------------------
```

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

Query 5:

a) Similar to query three, but it asks for cars without subscription.
b) Profiling results of query and query:
SELECT C.regno AS regno,   SUM(F.costPerPassing) AS totalfee FROM Car C JOIN Passing P USING(regno) JOIN TaxClass T USING(taxclass) JOIN Fee F USING(taxclass) LEFT JOIN Subscription S ON C.regno = S.regno WHERE F.type = 'withsubscription' AND S.regno IS NULL GROUP BY C.regno HAVING totalfee > 5000;

```
MariaDB [car_db]> show profiles;
+----------+------------+-----------------------------------------------------------------------------------------------------------
---+
| Query_ID | Duration   | Query
   |
+----------+------------+-----------------------------------------------------------------------------------------------------------
---+
|       59 | 4.23746201 | SELECT C.regno AS regno,         SUM(F.costPerPassing) AS totalfee FROM Car C  JOIN Passing P USING(regno) JOIN TaxClass T USING(taxclas
s) JOIN Fee F USING(taxclass) LEFT JOIN Subscription S ON C.regno = S.regno WHERE F.type = 'withsubscription' AND S.regno IS NULL GROUP BY C.regno HAVING totalfee
 > |
|       60 | 4.23484727 | SELECT C.regno AS regno,         SUM(F.costPerPassing) AS totalfee FROM Car C  JOIN Passing P USING(regno) JOIN TaxClass T USING(taxclas
s) JOIN Fee F USING(taxclass) LEFT JOIN Subscription S ON C.regno = S.regno WHERE F.type = 'withsubscription' AND S.regno IS NULL GROUP BY C.regno HAVING totalfee
 > |
|       61 | 4.23186782 | SELECT C.regno AS regno,         SUM(F.costPerPassing) AS totalfee FROM Car C  JOIN Passing P USING(regno) JOIN TaxClass T USING(taxclas
s) JOIN Fee F USING(taxclass) LEFT JOIN Subscription S ON C.regno = S.regno WHERE F.type = 'withsubscription' AND S.regno IS NULL GROUP BY C.regno HAVING totalfee
 > |
+----------+------------+-----------------------------------------------------------------------------------------------------------
---+
3 rows in set (0.000 sec)
```

c) Denormalizing would be about the same as all the other queries. Creating a table which pre calculates the sum of each of the cars, makes queriying the denormalized table simple. But it does compromise on the normalizing of the table and breaks 3NF.

```
CREATE TABLE query5 (
  -> regno CHAR(7) PRIMARY KEY,
  -> totalFee DECIMAL (10,2)
  -> );
INSERT INTO query5 (regno, totalfee)
  -> SELECT C.regno, SUM(F.costPerPassing) as totalfee
  -> from Car C
  -> join Passing P using(regno)
  -> join TaxClass T using(taxclass)
  -> join Fee F using(taxclass)
  -> left join Subscription S on C.regno = S.regno
  -> where F.type = 'withsubscription' AND S.regno is null
  -> group by C.regno;

SELECT * FROM query5 WHERE totalFee > 5000;
```

This probably didn't include all the required profiling/index results, but the other profiling/index results can be found in the previous attempt

Profiling results:

d)

```
MariaDB [car_db]> show profiles;
+----------+------------+-----------------------------------------------+
| Query_ID | Duration   | Query                                         |
+----------+------------+-----------------------------------------------+
|       69 | 0.00467680 | select * from query5 where totalfee > 5000 |
|       70 | 0.00464582 | select * from query5 where totalfee > 5000 |
|       71 | 0.00462014 | select * from query5 where totalfee > 5000 |
+----------+------------+-----------------------------------------------+
3 rows in set (0.000 sec)

MariaDB [car_db]> EXPLAIN SELECT C.regno AS regno,        SUM(F.costPerPassing) AS totalfee FROM Car C  JOIN Passing P USING(regno) JOIN TaxClass T USING(taxclass)
 > 5000;
+------+-------------+-------+--------+-------------------------------+----------+---------+-------------------+------+-----------------------------------------------+
| id   | select_type | table | type   | possible_keys                 | key      | key_len | ref               | rows | Extra                                         |
+------+-------------+-------+--------+-------------------------------+----------+---------+-------------------+------+-----------------------------------------------+
|    1 | SIMPLE      | F     | ref    | taxclass,idx_type             | idx_type | 2       | const             | 10   | Using where; Using index; Using temporary; Using filesort |
|    1 | SIMPLE      | T     | eq_ref | PRIMARY                       | PRIMARY  | 2       | car_db.F.taxclass | 1    |                                               |
|    1 | SIMPLE      | C     | ref    | PRIMARY,taxclass,idx_regno    | taxclass | 2       | car_db.F.taxclass | 3593 | Using index                                   |
|    1 | SIMPLE      | S     | eq_ref | PRIMARY,idx_regno_subscription | PRIMARY  | 21      | car_db.C.regno    | 1    | Using where; Using index; Not exists          |
|    1 | SIMPLE      | P     | ref    | regno                         | regno    | 21      | car_db.C.regno    | 1    |                                               |
+------+-------------+-------+--------+-------------------------------+----------+---------+-------------------+------+-----------------------------------------------+
5 rows in set (0.000 sec)
```