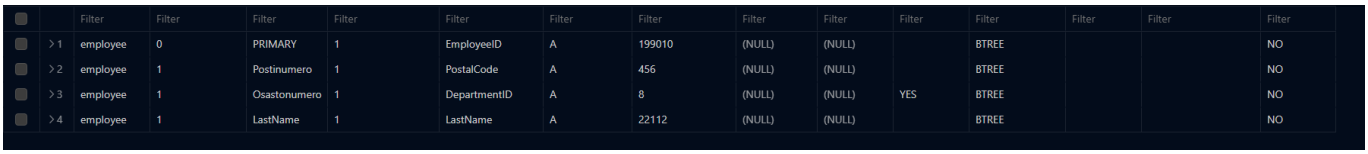
Assignment_Indexing_v4.pdf

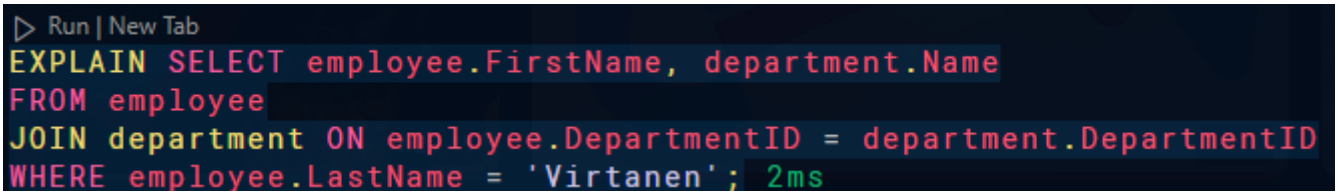
Task 1



		Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
	>1	employee	0	PRIMARY	1	EmployeeID	A	199010	(NULL)	(NULL)		BTREE		NO
	>2	employee	1	Postinumero	1	PostalCode	A	456	(NULL)	(NULL)		BTREE		NO
	>3	employee	1	Osastonumero	1	DepartmentID	A	8	(NULL)	(NULL)	YES	BTREE		NO
	>4	employee	1	LastName	1	LastName	A	22112	(NULL)	(NULL)		BTREE		NO

You can verify the existance of and index by running the SHOW INDEX command on a specified table or database. The command will return a list of indexes.

Task 2



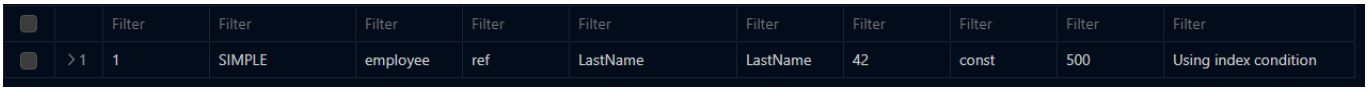
```
Run | New Tab
EXPLAIN SELECT employee.FirstName, department.Name
FROM employee
JOIN department ON employee.DepartmentID = department.DepartmentID
WHERE employee.LastName = 'Virtanen'; 2ms
```

	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
>1	1	SIMPLE	employee	ref	Osastonumero,LastName	LastName	42	const	500	Using index condition
>2	1	SIMPLE	department	ALL	PRIMARY	(NULL)	(NULL)	(NULL)	5	Using where; Using join buf

The EXPLAIN command in SQL is used to analyze and provide information about how a specific SQL query will be executed by the database engine. It helps in understanding the execution plan and performance of the query.

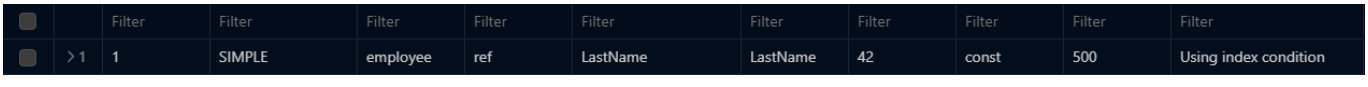
Task 3

The `FORCE INDEX` tells the query optimizer to use the specified index for the query.



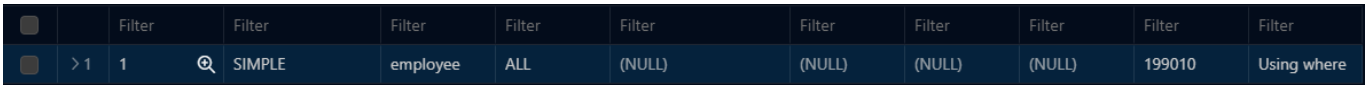
		Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
	>1	1	SIMPLE	employee	ref	LastName	LastName	42	const	500	Using index condition

The `USE INDEX` suggests to the query optimizer to consider the specified indexes when executing the query. It does not force the optimizer to use the index.



		Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
	>1	1	SIMPLE	employee	ref	LastName	LastName	42	const	500	Using index condition

The `IGNORE INDEX` tells the query optimizer to ignore the specified indexes when executing the query.



		Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
	>1	1	SIMPLE	employee	ALL	(NULL)	(NULL)	(NULL)	(NULL)	199010	Using where

```
Run | New Tab | JSON
SELECT FirstName FROM employee FORCE INDEX(LastName) WHERE `LastName` = 'Virtanen'; 1ms

Run | New Tab | JSON
SELECT FirstName FROM employee USE INDEX (LastName) WHERE `LastName` = 'Virtanen'; 1ms

Run | New Tab | JSON
SELECT FirstName FROM employee IGNORE INDEX (LastName) WHERE `LastName` = 'Virtanen'; 8ms
```

Task 4

```
Run | New Tab | JSON
SELECT FirstName, LastName FROM Employee WHERE Salary/2 > 1500; 2ms
```

		Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
	>1	1	SIMPLE	Employee	ALL	(NULL)	(NULL)	(NULL)	(NULL)	199010	Using where

Adding the force tag to the query.

```
Run | New Tab | JSON
SELECT FirstName, LastName FROM Employee USE INDEX(Salary) WHERE Salary/2 > 1500; 2ms
```

The index has no effect on the query time.

Task 5

(Side note! The query example for creating the index is plainly wrong)

Creating the index.

```
Run | New Tab
CREATE INDEX LastName ON employee (LastName(6)) 273ms
```

Index with length of 2

```
Benchmark
        Average number of seconds to run all queries: 0.015 seconds
        Minimum number of seconds to run all queries: 0.000 seconds
        Maximum number of seconds to run all queries: 0.032 seconds
        Number of clients running queries: 5
        Average number of queries per client: 1
```

Index with length of 4

```
Benchmark
        Average number of seconds to run all queries: 0.006 seconds
        Minimum number of seconds to run all queries: 0.000 seconds
        Maximum number of seconds to run all queries: 0.016 seconds
        Number of clients running queries: 5
        Average number of queries per client: 1
```

Index with length of 6

```
Benchmark
        Average number of seconds to run all queries: 0.003 seconds
        Minimum number of seconds to run all queries: 0.000 seconds
        Maximum number of seconds to run all queries: 0.016 seconds
        Number of clients running queries: 5
        Average number of queries per client: 1
```

Index with length of 8

```
Benchmark
        Average number of seconds to run all queries: 0.003 seconds
        Minimum number of seconds to run all queries: 0.000 seconds
        Maximum number of seconds to run all queries: 0.016 seconds
        Number of clients running queries: 5
        Average number of queries per client: 1
```

The first 6 characters seems to be enough for the index to be as effective as possible.

Task 6

(Side note! There is no such variable as "slow_queries_log" in MySQL)

Cant be assed to debug the log file.