Faculty Of Engineering

Cairo University

Advanced Database System

Report

Ahmed Mohamed Ismail Nabeel	1180501
Mostafa Ashraf Ahmed	1180406
Moaz Mohamed Elsherbini	1180528
Nader Youhanna Adib	1180477

Table of Contents

System Information

Processor	Intel(R) Core(TM) i7-3612QM CPU @ 2.10GHz 2.10 GHz
Installed RAM	16.0 GB (15.9 GB usable)

Operating System: Windows 10 Pro

Hard disk: 1 TB SSD

Item	Value
OS Name	Microsoft Windows 10 Pro
Version	10.0.19045 Build 19045
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	AHMED-DELL
System Manufacturer	Dell Inc.
System Model	Inspiron 7520
System Type	x64-based PC
System SKU	Inspiron 7520
Processor	Intel(R) Core(TM) i7-3612QM CPU @ 2.10GHz, 2101 Mhz, 4 Core(s), 8 Logical ...
BIOS Version/Date	Dell Inc. A14, 17-May-18
SMBIOS Version	2.7
Embedded Controller Version	1.01
BIOS Mode	Legacy
BaseBoard Manufacturer	Dell Inc.
BaseBoard Product	0PXH02
BaseBoard Version	A00
Platform Role	Mobile
Secure Boot State	Unsupported
PCR7 Configuration	Binding Not Possible
Windows Directory	C:\Windows
System Directory	C:\Windows\system32
Boot Device	\Device\HarddiskVolume1
Locale	United States
Hardware Abstraction Layer	Version = "10.0.19041.2251"

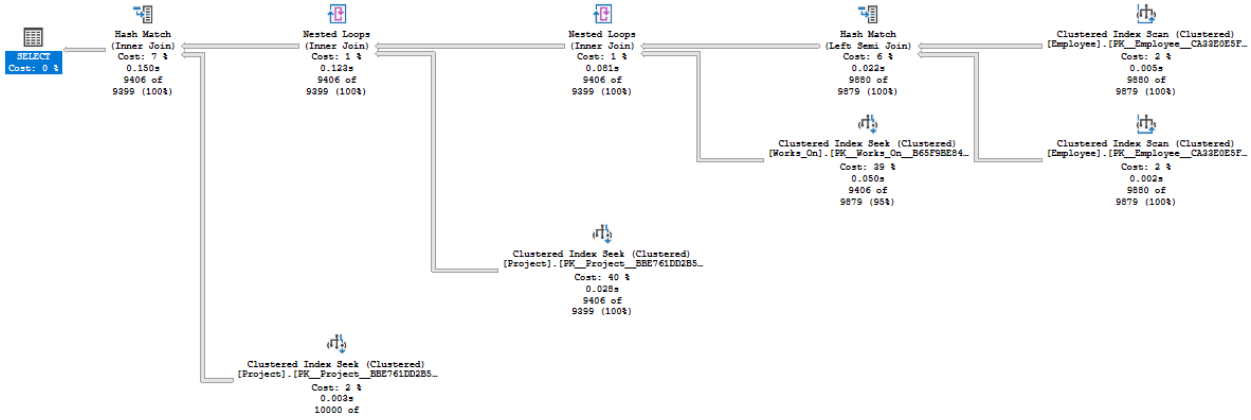
Installed Physical Memory (RAM)	16.0 GB
Total Physical Memory	15.9 GB
Available Physical Memory	6.59 GB
Total Virtual Memory	18.3 GB
Available Virtual Memory	7.63 GB
Page File Space	2.38 GB
Page File	F:\pagefile.sys
Kernel DMA Protection	Off

Query 1

Before Optimization

```
SELECT E.Fname, E.Salary, E.Super_Ssn, P.Pname, W.Hours
FROM Employee E INNER LOOP JOIN Works_On W ON E.Ssn = W.Essn
                            INNER LOOP JOIN Project P ON W.Pno = P.Pnumber
WHERE  W.Hours > 500
              AND E.Salary in (
                     SELECT E.Salary FROM Employee E
                     WHERE E.Salary > 10000
              )
              AND P.Pnumber in (
                     SELECT P.Pnumber FROM Project P
                     WHERE  P.Pnumber > 500
    )
GO
```

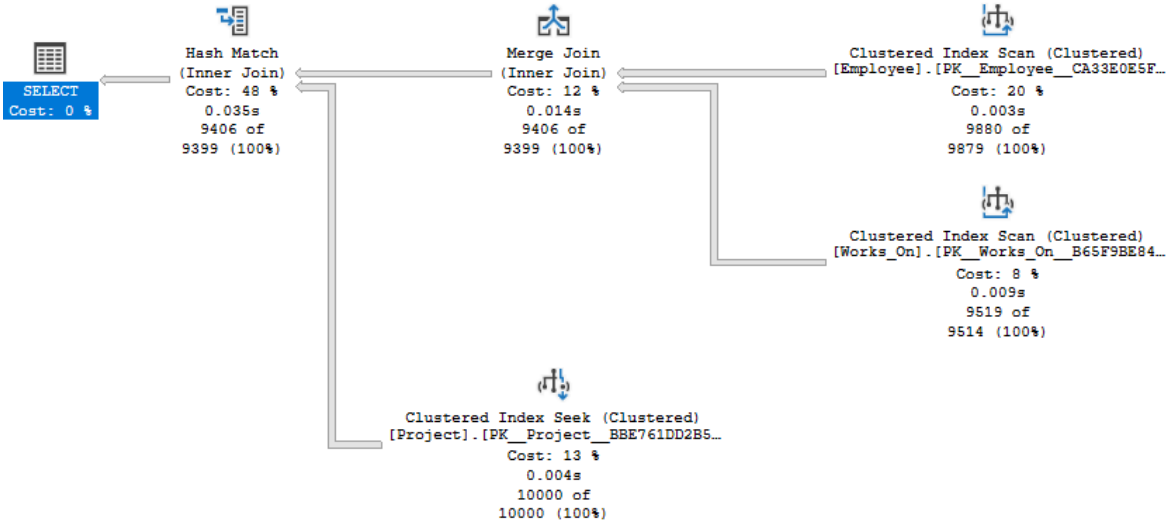
Execution Plan



After Optimization

```
SELECT  E.Fname, E.Salary, E.Super_Ssn,  P.Pname,  W.Hours
FROM
    Employee E
    INNER JOIN Works_On W ON E.Ssn = W.Essn
    INNER JOIN Project P ON W.Pno = P.Pnumber
WHERE
    E.Salary > 10000
    AND P.Pnumber > 500
    AND W.Hours > 500
GO
```

Execution Plan



Optimization Done

- INNER JOIN instead of NESTED LOOP JOIN
- Non-clustered index on Ssn & Pnumber

Action	CPU Time (s)	Elapsed Time (s)
Query 1 (No optimization)	0.688	2.180
Query 1 (Optimized INNER JOIN)	0.136	2.106
Adding non-clustering index	0.123	1.902
NoSQL MongoDB	2.223	

Theoretical Parallel Query Processing Report

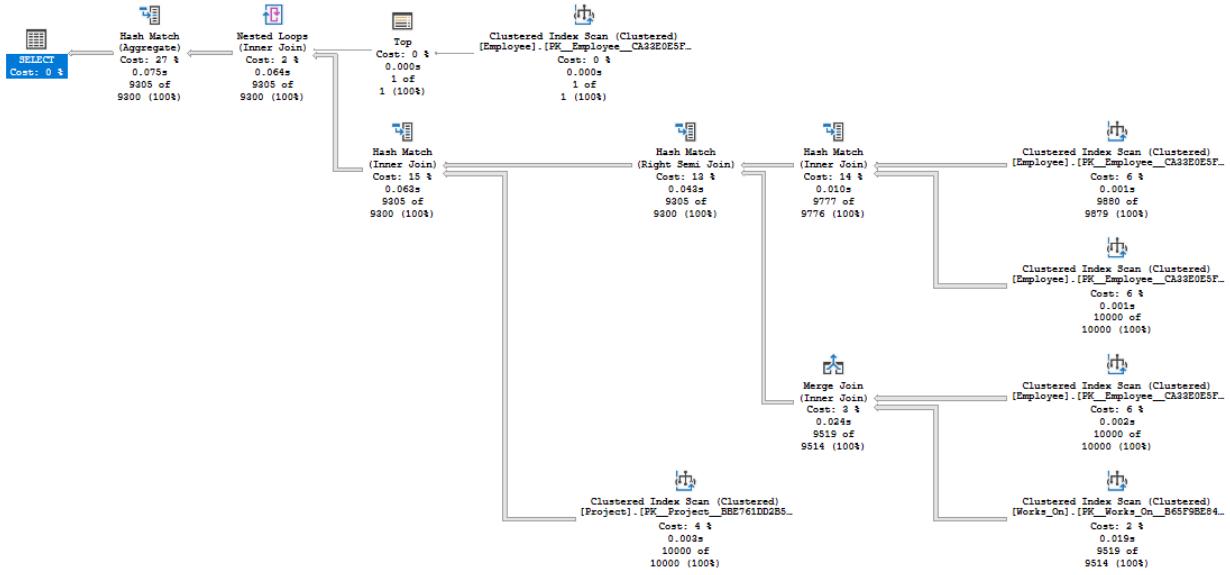
- Clustered index scan on Employee table
- Clustered index scan on Project table
- Clustered index scan on Works_On table

Query 2

Before Optimization

```
SELECT DISTINCT E.Fname, E.Salary, E.Super_Ssn, P.Pname, W.Hours
FROM Employee E, Employee M, Works_On W, Project P
WHERE
    W.Hours > 500
    AND E.Ssn = W.Essn
    AND W.Pno = P.Pnumber
    AND E.Super_Ssn in (
        SELECT
            M.Ssn
        FROM
            Employee E,
            Employee M
        WHERE
            E.Super_Ssn = M.Ssn
            AND E.Salary > 10000
    )
GO
```

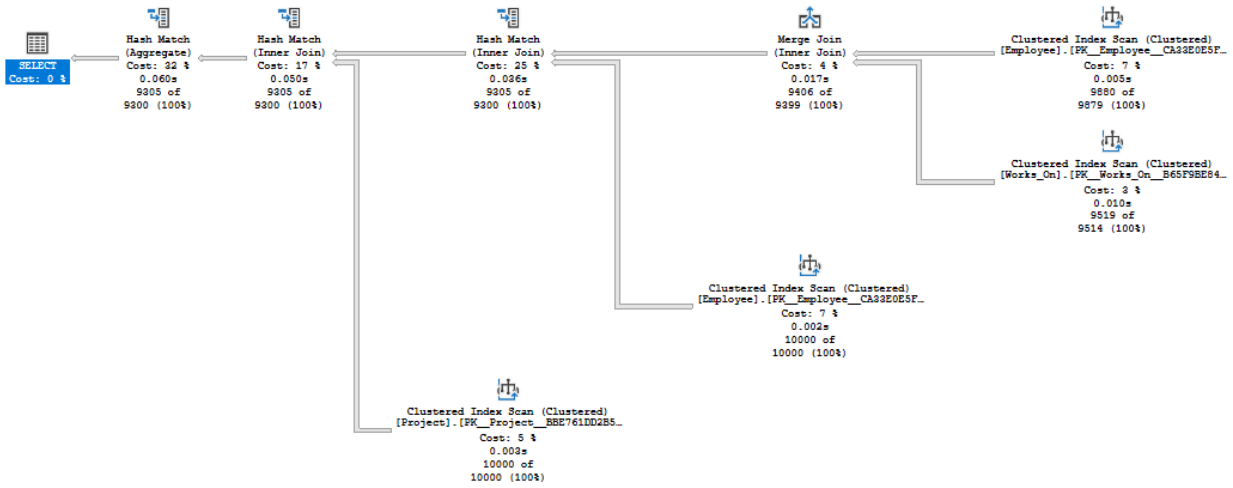
Execution Plan



After Optimization

```
SELECT DISTINCT E.Fname, E.Salary, E.Super_Ssn, P.Pname, W.Hours
FROM Employee E
    INNER join Employee M on E.Super_Ssn = M.Ssn
    inner join Works_On W on E.Ssn = W.Essn
    INNER join Project P on W.Pno = P.Pnumber
WHERE
    Hours > 500
    AND E.Salary > 10000
GO
```

Execution plan



Optimization Done

- INNER JOIN instead of nested loop.
- Non-clustered index on Ssn, Pnumber, Salary and Hours.

Action	CPU Time (s)	Elapsed Time (s)
Query 1 (No optimization)	0.476	4.467
Query 1 (Optimized INNER JOIN)	0.438	1.922
Adding non-clustering index	0.317	1.800
NoSQL MongoDB	310.751	

Theoretical Parallel Query Processing Report

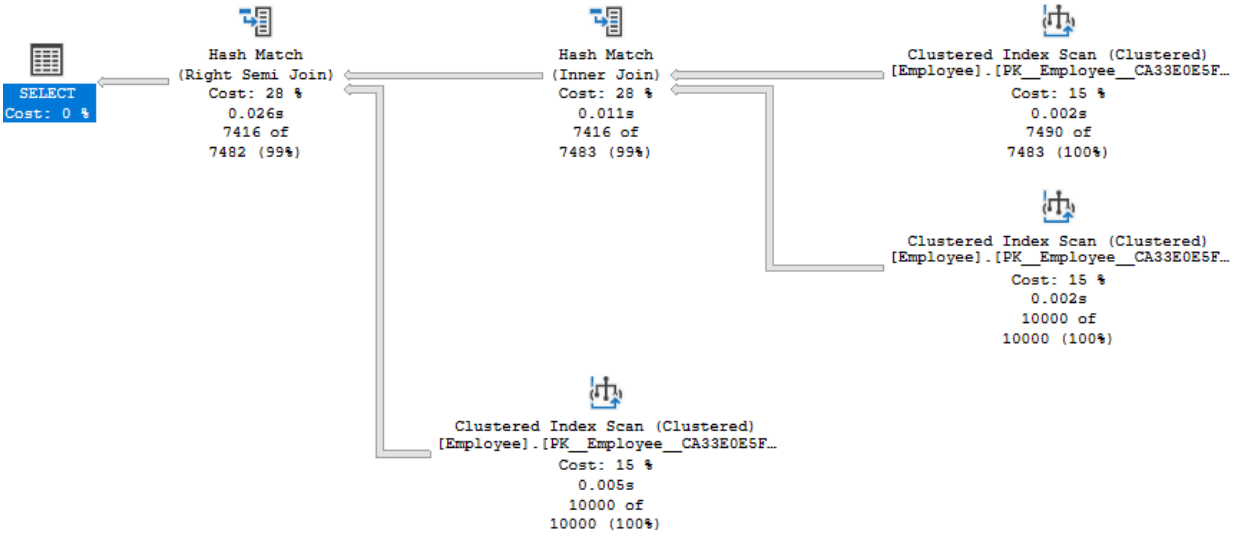
- Clustered index scan on Project table
- Clustered index scan on Works_On table
- Clustered index scan on Employee

Query 3

Before Optimization

```
SELECT * FROM Employee E
WHERE
    E.Super_Ssn IN (
        SELECT
            M.SSN
        FROM
            Employee E,
            Employee M
        WHERE
            E.Super_Ssn = M.Ssn
            AND M.Bdate > '1970-01-01'
    );
```

Execution Plan



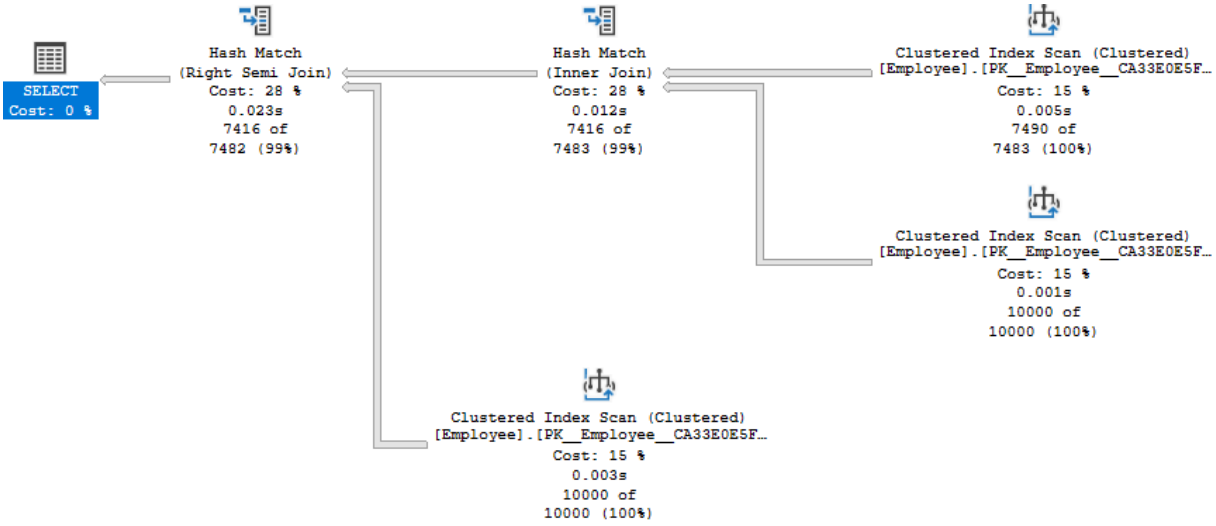
After Optimization

```
EXEC sp_Employee_Super_Ssn '1970-01-01'
```

Stored Procedure

```
CREATE PROCEDURE sp_Employee_Super_Ssn @Bdate DATETIME AS BEGIN
Select
    *
from
    Employee E
Where
    E.Super_Ssn in (
        Select
            M.Ssn
        From
            Employee E,
            Employee M
        where
            E.Super_Ssn = M.Ssn
            And M.Bdate > @Bdate
    )
END
```

Execution Plan



Optimization Done

- Used stored procedure.
- Non-clustered index on Ssn & Bdate.

Action	CPU Time (s)	Elapsed Time (s)
Query 1 (No optimization)	0.139	1.983
Query 1 (Stored Procedure)	0.145	1.751
Adding non-clustering index	0.121	2.002
NoSQL MongoDB	17.946	

Theoretical Parallel Query Processing Report

- Clustered index scan on Employee

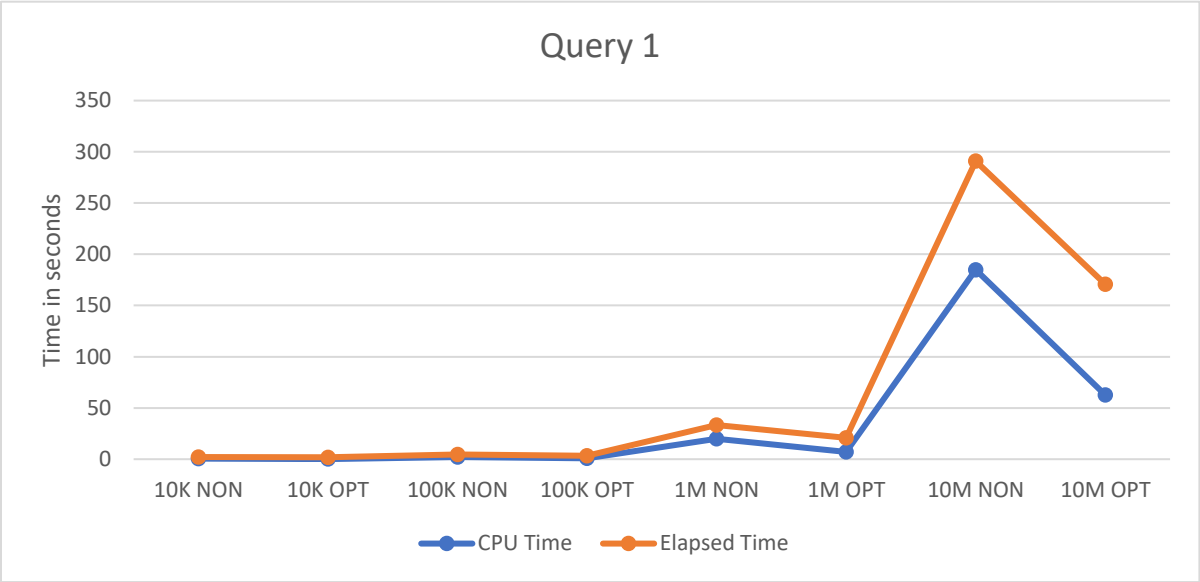
Database Statistics

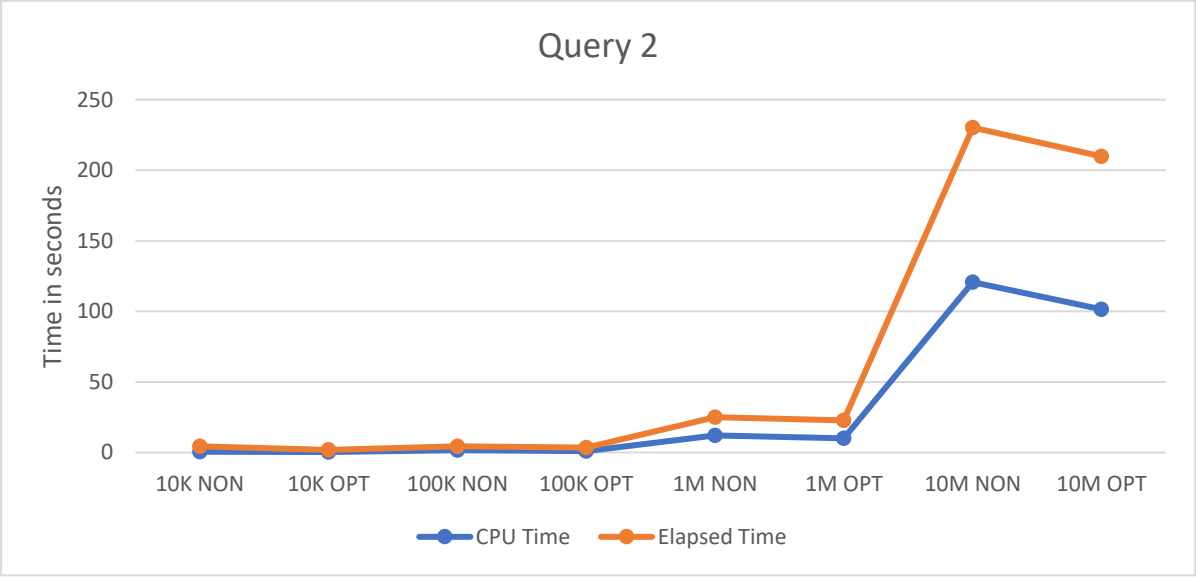
Table Name	Row Count	Main Key	Indexes	FK	Identity Column	Max Row Size (Bytes)
Employee	1M	Yes	1	2	Yes	71
Department	1M	Yes	1	1	Yes	19
Project	1M	Yes	1	1	Yes	38
Works_On	1M	No	1	2	No	12
Department_Location	1M	No	1	1	No	19
Dependent	1M	No	1	1	No	38

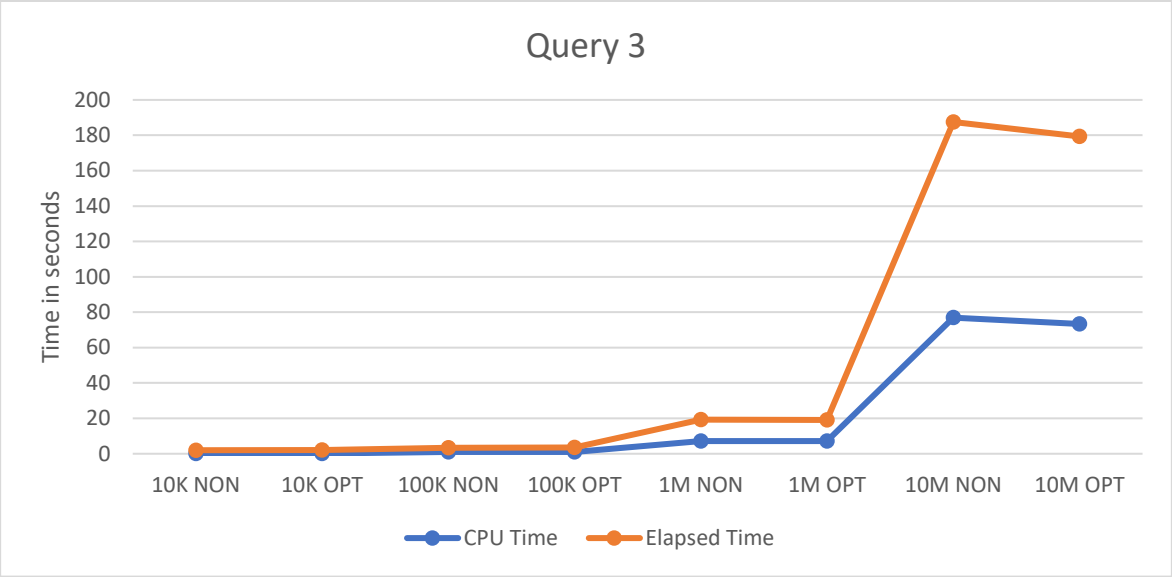
Times for Different Database Sizes

Query \ Size	10K (s)	100K (s)	1M (s)	10M (s)
Query 1 (Non-optimized) CPU Time	0.688	2.183	19.771	184.889
Query 1 (Non-optimized) Elapsed Time	2.180	4.666	33.409	290.898
Query 1 (Optimized) CPU Time	0.123	0.769	7.126	62.678
Query 1 (Optimized) Elapsed Time	1.902	3.315	20.760	170.614
Query 2 (Non-optimized) CPU Time	0.476	1.739	12.106	120.700
Query 2 (Non-optimized) Elapsed Time	4.467	4.348	25.066	230.224
Query 2 (Optimized) CPU Time	0.317	0.997	10.092	101.365
Query 2 (Optimized) Elapsed Time	1.800	3.495	22.735	209.833
Query 3 (Non-optimized) CPU Time	0.139	0.944	7.204	76.916
Query 3 (Non-optimized) Elapsed Time	1.983	3.432	19.330	187.511
Query 3 (Optimized) CPU Time	0.121	0.976	7.184	73.292
Query 3 (Optimized) Elapsed Time	2.002	3.542	19.047	179.315

Time Analysis






Query 3
Time in seconds
200
180
160
140
120
100
80
60
40
20
0
10K NON
10K OPT
100K NON
100K OPT
1M NON
1M OPT
10M NON
10M OPT
CPU Time
Elapsed Time

NoSQL Queries

Query 1

```
db.getCollection("Employee").aggregate(
    [
        {
            "$project" : {
                "_id" : NumberInt(0),
                "E" : "$$ROOT"
            }
        },
        {
            "$lookup" : {
                "localField" : "E.Ssn",
                "from" : "Works_On",
                "foreignField" : "Essn",
                "as" : "W"
            }
        },
        {
            "$unwind" : {
                "path" : "$W",
                "preserveNullAndEmptyArrays" : false
            }
        },
        {
            "$lookup" : {
                "localField" : "W.Pno",
                "from" : "Project",
                "foreignField" : "Pnumber",
                "as" : "P"
            }
        },
        {
            "$unwind" : {
                "path" : "$P",
                "preserveNullAndEmptyArrays" : false
            }
        },
        {
            "$match" : {
                "E.Salary" : {
                    "$gt" : NumberLong(10000)
                },
                "P.Pnumber" : {
                    "$gt" : NumberLong(500)
                },
                "W.Hours" : {
                    "$gt" : NumberLong(500)
                }
            }
        },
        {
            "$project" : {
                "E.Fname" : "$E.Fname",
                "E.Salary" : "$E.Salary",
                "E.Super_Ssn" : "$E.Super_Ssn",
                "P.Pname" : "$P.Pname",
                "W.Hours" : "$W.Hours",
                "_id" : NumberInt(0)
            }
        }
    ],
    {
        "allowDiskUse" : true
    }
);
```

Query 2

```
db.getCollection("Employee").aggregate(
    [
        {
            "$project" : {
                "_id" : NumberInt(0),
                "E" : "$$ROOT"
            }
        },
        {
            "$lookup" : {
                "localField" : "E.Super_Ssn",
                "from" : "Employee",
                "foreignField" : "Ssn",
                "as" : "M"
            }
        },
        {
            "$unwind" : {
                "path" : "$M",
                "preserveNullAndEmptyArrays" : false
            }
        },
        {
            "$lookup" : {
                "localField" : "E.Ssn",
                "from" : "Works_On",
                "foreignField" : "Essn",
                "as" : "W"
            }
        },
        {
            "$unwind" : {
                "path" : "$W",
                "preserveNullAndEmptyArrays" : false
            }
        },
        {
            "$lookup" : {
                "localField" : "W.Pno",
                "from" : "Project",
                "foreignField" : "Pnumber",
                "as" : "P"
            }
        },
        {
            "$unwind" : {
                "path" : "$P",
                "preserveNullAndEmptyArrays" : false
            }
        },
        {
            "$match" : {
                "W.Hours" : {
                    "$gt" : NumberLong(500)
                },
```

```
                "E.Salary" : {
                    "$gt" : NumberLong(10000)
                }
            }
        },
        {
            "$project" : {
                "E.Fname" : "$E.Fname",
                "E.Salary" : "$E.Salary",
                "E.Super_Ssn" : "$E.Super_Ssn",
                "P.Pname" : "$P.Pname",
                "W.Hours" : "$W.Hours",
                "_id" : NumberInt(0)
            }
        },
        {
            "$group" : {
                "_id" : null,
                "distinct" : {
                    "$addToSet" : "$$ROOT"
                }
            }
        },
        {
            "$unwind" : {
                "path" : "$distinct",
                "preserveNullAndEmptyArrays" : false
            }
        },
        {
            "$replaceRoot" : {
                "newRoot" : "$distinct"
            }
        }
    ],
    {
        "allowDiskUse" : true
    }
);
```

Query 3

```
db.getCollection("Employee").aggregate(
    [
        {
            "$project" : {
                "_id" : NumberInt(0),
                "E" : "$$ROOT"
            }
        },
        {
            "$lookup" : {
                "localField" : "E.non_existing_field",
                "from" : "Employee",
                "foreignField" : "non_existing_field",
                "as" : "M"
            }
        },
        {
            "$unwind" : {
                "path" : "$M",
                "preserveNullAndEmptyArrays" : false
            }
        },
        {
            "$match" : {
                "$and" : [
                    {
                        "$expr" : {
                            "$eq" : [
                                "$E.Super_Ssn",
                                "$M.Ssn"
                            ]
                        }
                    },
                    {
                        "$expr" : {
                            "$gt" : [
                                "$M.Bdate",
                                "1970-01-01"
                            ]
                        }
                    }
                ]
            }
        }
    ],
    {
        "allowDiskUse" : true
    }
);
```

Optimizations

Schema Enhancement

- We used the same schema used in Phase 1

Memory and Cache Management Enhancement

- We used stored procedures to enhance memory and cache management

Indexes Modifications

- We added some non-clustered inexes to speed up the data selection as follows:
    - Non-clustered index on: Ssn (Employee Table)
    - Non-clustered index on: Pnumber (Project Table)
    - Non-clustered index on: Bdate (Works_on Table)
    - Non-clustered index on: Salary (Employee Table)

Query Rewriting Modifications

- We rewrote the queries to replace the nested loops with INNER JOIN to make the query execute faster

Conclusion

- Using SQL server is recommended for relational schemas as NoSQL has worse performance
- Index tuning has a great effect in the execution time for executing the queries
- Optimization for SQL query has a more noticeable effect for larger database sizes
- Stored procedures did not yield the expected optimization
- Seeing the execution plan and the index scan is very useful when tryin to generate non-clustered indexes to convert the scanning operations into seeking operations.