Report

Laboratory Work 4

Dmitry Ladutsko

July 19, 2022

Task 1: Auto Trace configuration training

№	Auto Trace Configuration Options	Expected Results	Description
	`set autotrace off`	No AUTOTRACE report is generated.	This is the default.
	`set autotrace on`	The AUTOTRACE report	Includes both the optimizer execution path and the SQL statement execution statistics.
	`set autotrace traceonly`	Like SET AUTOTRACE ON	Suppresses the printing of the user's query output, if any. This option is useful when you are tuning a large query, but do not want to see the query report.
	`set autotrace on explain`	The AUTOTRACE report	Shows only the optimizer execution path.
	`set autotrace on statistics`	The AUTOTRACE report	Shows only the SQL statement execution statistics.
	`set autotrace on explain statistics`	Shows only the SQL statement execution statistics.	= set autotrace on
	`set autotrace traceonly explain`	Execution plan w/o query result	These options should be used when a large result set is expected.
	`set autotrace traceonly statistics`	Statistics w/o query result	
	`set autotrace traceonly explain statistics`	Execution plan + statistics w/o query result	
	`set autotrace off explain`	Disables autotrace utility	
	`set autotrace off statistics`		
	`set autotrace off explain statistics`		

Task 2: Nested Loop Joins

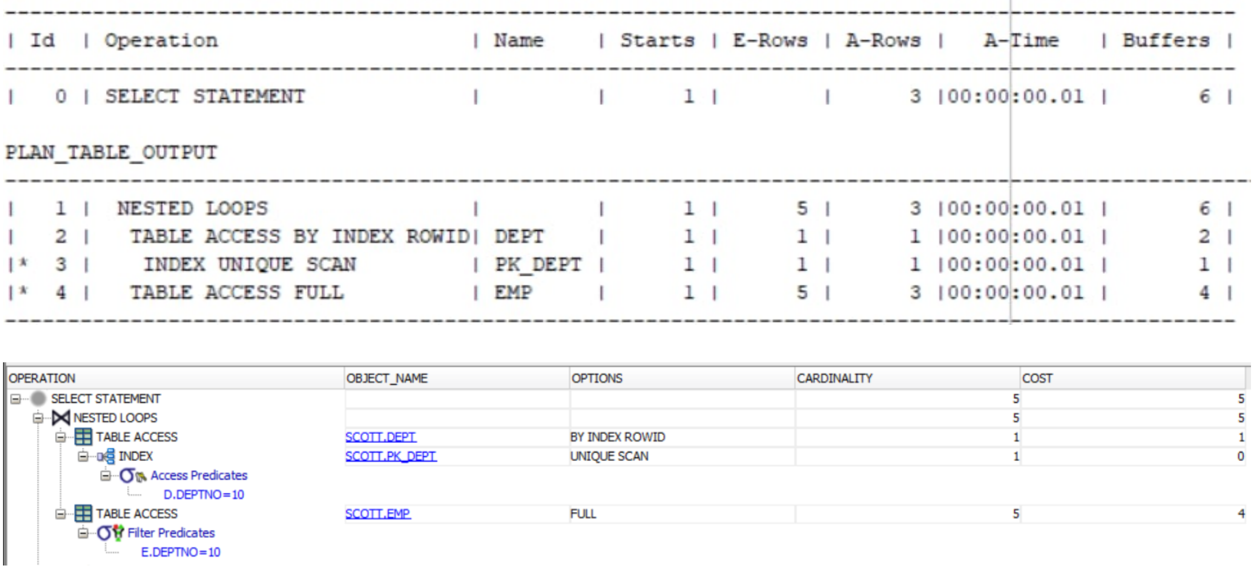
Example:

SELECT /*+ gather_plan_statistics */ *

FROM scott.emp e, scott.dept d

WHERE e.deptno = d.deptno

AND d.deptno = 10



```
-----------------------------------------------------------------------------------------------
| Id  | Operation                    | Name    | Starts | E-Rows | A-Rows |   A-Time   | Buffers |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |         |      1 |        |      3 |00:00:00.01 |       6 |

PLAN_TABLE_OUTPUT
-----------------------------------------------------------------------------------------------
|   1 |  NESTED LOOPS                |         |      1 |      5 |      3 |00:00:00.01 |       6 |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPT    |      1 |      1 |      1 |00:00:00.01 |       2 |
|*  3 |    INDEX UNIQUE SCAN         | PK_DEPT |      1 |      1 |      1 |00:00:00.01 |       1 |
|*  4 |   TABLE ACCESS FULL          | EMP     |      1 |      5 |      3 |00:00:00.01 |       4 |
-----------------------------------------------------------------------------------------------
```

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			5	5
NESTED LOOPS			5	5
TABLE ACCESS	SCOTT.DEPT	BY INDEX ROWID	1	1
INDEX	SCOTT.PK_DEPT	UNIQUE SCAN	1	0
Access Predicates				
D.DEPTNO=10				
TABLE ACCESS	SCOTT.EMP	FULL	5	4
Filter Predicates				
E.DEPTNO=10				

Picture 2.1 - Query Result

Note. I used /*+ gather_plan_statistics */ hint because for some reason explain plan statement did not give me full needed stats (only id, operation and name columns). Nested loops join compare every row of table B with current row in table A. If it is a row, which satisfies the specified condition, it is selected, otherwise goes next and compare second raw from table A with all rows from table B. This method is quite useful with small table, but not with big.

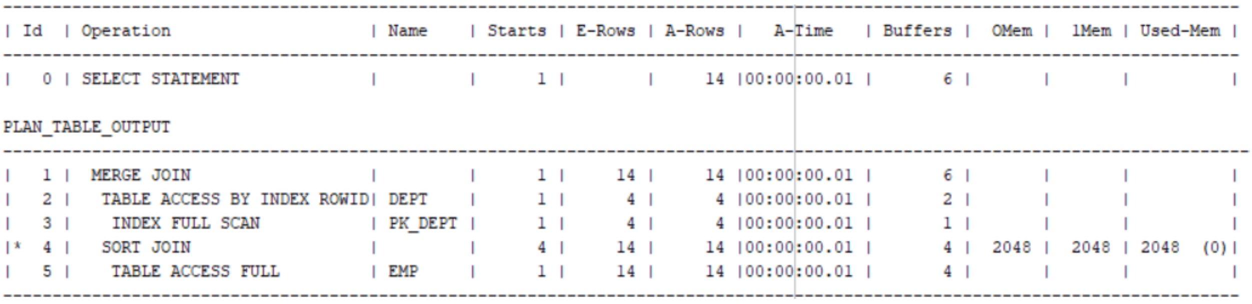
Task 3: Sort-Merge Joins

set autotrace on explain;

SELECT /*+ use_merge(e d) gather_plan_statistics */ *

FROM scott.emp e, scott.dept d

WHERE e.deptno = d.deptno



```
------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name    | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |         |      1 |        |     14 |00:00:00.01 |       6 |       |       |          |

PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------------------------------------------------
|   1 |  MERGE JOIN                  |         |      1 |     14 |     14 |00:00:00.01 |       6 |       |       |          |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPT    |      1 |      4 |      4 |00:00:00.01 |       2 |       |       |          |
|   3 |    INDEX FULL SCAN           | PK_DEPT |      1 |      4 |      4 |00:00:00.01 |       1 |       |       |          |
|*  4 |   SORT JOIN                  |         |      4 |     14 |     14 |00:00:00.01 |       4 |  2048 |  2048 | 2048  (0)|
|   5 |    TABLE ACCESS FULL         | EMP     |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
------------------------------------------------------------------------------------------------------------------------
```

Picture 3.1 - Query Result

Note. Using this type of join we need to be sure that all used tables are sorted by columns, mentioned in query condition. Oracle scans each row once. This gives us nice benefits, but we should keep in mind that sorting loads the system a lot if we use big tables.

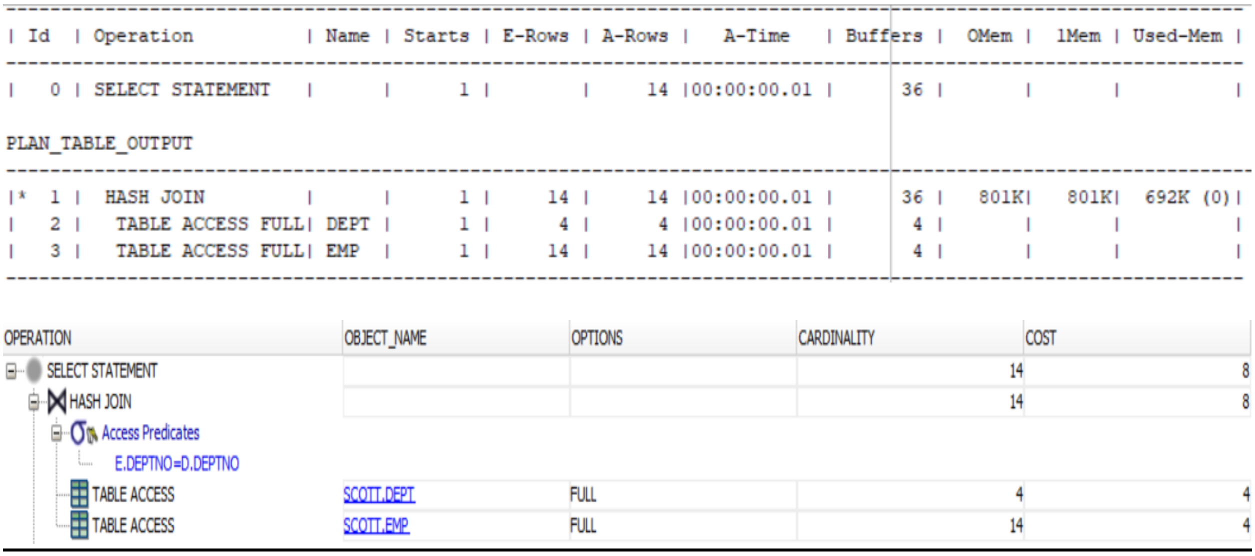
Task 4: Hash Joins

set autotrace on explain;

SELECT /*+ gather_plan_statistics USE_HASH(e d) */ *

FROM scott.emp e, scott.dept d

WHERE e.deptno = d.deptno



```
-----------------------------------------------------------------------------------------------------------------
| Id  | Operation          | Name | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |      1 |        |     14 |00:00:00.01 |      36 |       |       |          |

PLAN_TABLE_OUTPUT
-----------------------------------------------------------------------------------------------------------------
|*  1 |  HASH JOIN         |      |      1 |     14 |     14 |00:00:00.01 |      36 |   801K|   801K|  692K (0)|
|   2 |   TABLE ACCESS FULL| DEPT |      1 |      4 |      4 |00:00:00.01 |       4 |       |       |          |
|   3 |   TABLE ACCESS FULL| EMP  |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
-----------------------------------------------------------------------------------------------------------------
```

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			14	8
HASH JOIN			14	8
Access Predicates				
E.DEPTNO=D.DEPTNO				
TABLE ACCESS	SCOTT.DEPT	FULL	4	4
TABLE ACCESS	SCOTT.EMP	FULL	14	4

Picture 4.1 - Query Result

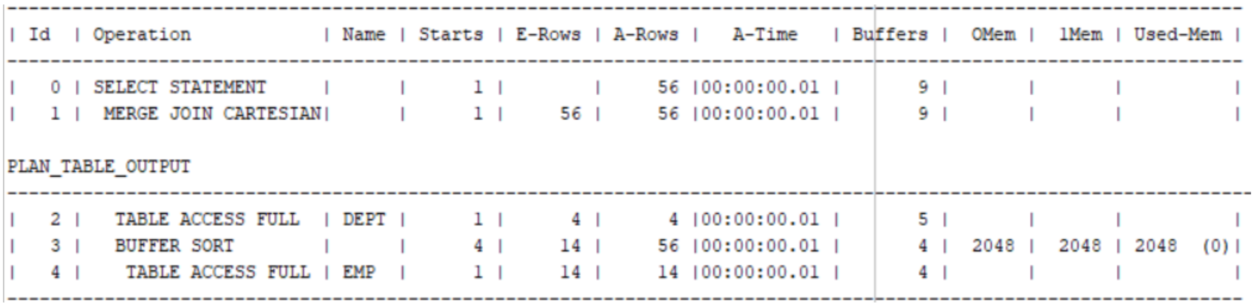
Note. Hash join is used when projections of the joined tables are not sorted on the join columns. In this case, the optimizer builds an in-memory hash table on the inner table's join column. The optimizer then scans the outer table for matches to the hash table, and joins data from the two tables accordingly.

Task 5: Cartesian Joins

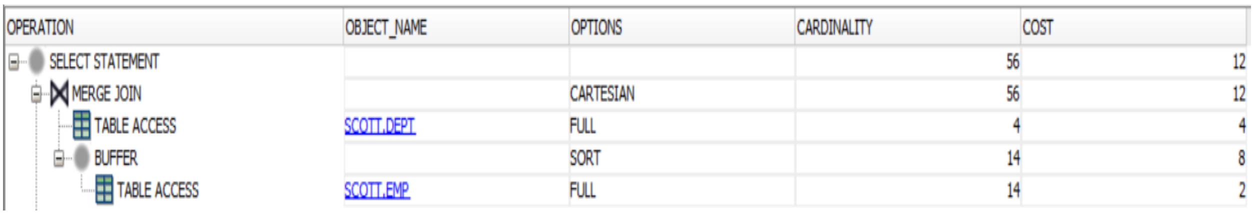
set autotrace on explain;

SELECT /*+ gather_plan_statistics */ *

FROM scott.emp, scott.dept



```
-----------------------------------------------------------------------------------------------------------------------
| Id  | Operation             | Name | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |      |      1 |        |     56 |00:00:00.01 |       9 |       |       |          |
|   1 |  MERGE JOIN CARTESIAN|      |      1 |     56 |     56 |00:00:00.01 |       9 |       |       |          |

PLAN_TABLE_OUTPUT
-----------------------------------------------------------------------------------------------------------------------
|   2 |   TABLE ACCESS FULL  | DEPT |      1 |      4 |      4 |00:00:00.01 |       5 |       |       |          |
|   3 |   BUFFER SORT        |      |      4 |     14 |     56 |00:00:00.01 |       4 |  2048 |  2048 | 2048  (0)|
|   4 |    TABLE ACCESS FULL | EMP  |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			56	12
MERGE JOIN		CARTESIAN	56	12
TABLE ACCESS	SCOTT.DEPT	FULL	4	4
BUFFER		SORT	14	8
TABLE ACCESS	SCOTT.EMP	FULL	14	2

Picture 5.1 - Query Result

Note. Cartesian joins are used when all rows in all tables listed in a query: each row in the first table is paired with all the rows in the second table. This happens when there is no relationship defined between the two tables.

Task 6: Left/Right Outer Joins

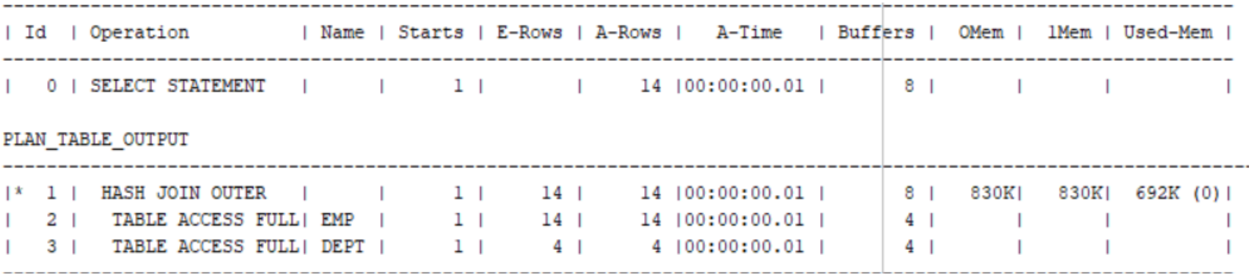
1. Left outer JOIN

set autotrace on explain;

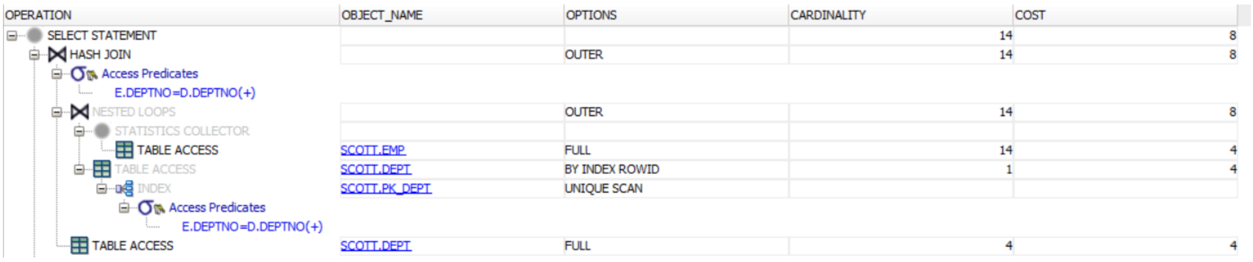
select /*+ gather_plan_statistics */ e.ename , e.deptno, e.job, d.dname

from scott.emp e, scott.dept d

where e.deptno = d.deptno(+);



```
-----------------------------------------------------------------------------------------------------------------------
| Id  | Operation           | Name | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |      |      1 |        |     14 |00:00:00.01 |       8 |       |       |          |

PLAN_TABLE_OUTPUT
-----------------------------------------------------------------------------------------------------------------------
|*  1 |  HASH JOIN OUTER    |      |      1 |     14 |     14 |00:00:00.01 |       8 |   830K|   830K|  692K (0)|
|   2 |   TABLE ACCESS FULL| EMP  |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
|   3 |   TABLE ACCESS FULL| DEPT |      1 |      4 |      4 |00:00:00.01 |       4 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			14	8
HASH JOIN		OUTER	14	8
Access Predicates				
E.DEPTNO=D.DEPTNO(+)				
NESTED LOOPS		OUTER	14	8
STATISTICS COLLECTOR				
TABLE ACCESS	SCOTT.EMP	FULL	14	4
TABLE ACCESS	SCOTT.DEPT	BY INDEX ROWID	1	4
INDEX	SCOTT.PK_DEPT	UNIQUE SCAN		
Access Predicates				
E.DEPTNO=D.DEPTNO(+)				
TABLE ACCESS	SCOTT.DEPT	FULL	4	4

Picture 6.1 - Query Result

Note. The specific of this type of join is that result includes unmatched rows from only the table that is specified before the LEFT OUTER JOIN clause. If we are joining two tables and want the result set to include unmatched rows from only one table, we use a LEFT OUTER JOIN clause (or a RIGHT OUTER JOIN) clause.

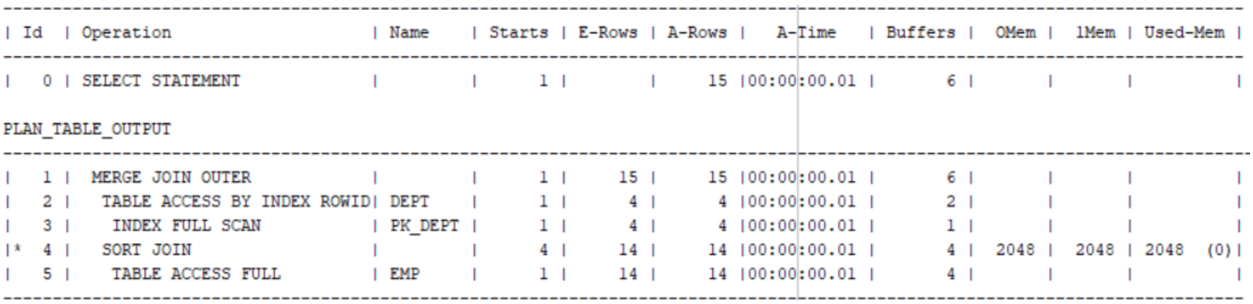
2. Right outer JOIN

set autotrace on explain;

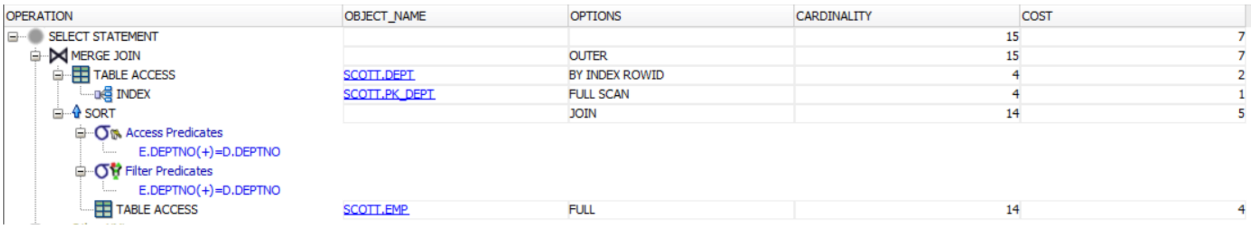
select /*+ gather_plan_statistics */ e.ename , e.deptno, e.job, d.dname

from scott.emp e, scott.dept d

where e.deptno(+) = d.deptno;



```
------------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name    | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |         |      1 |        |     15 |00:00:00.01 |       6 |       |       |          |

PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------------------------------------
|   1 |  MERGE JOIN OUTER            |         |      1 |     15 |     15 |00:00:00.01 |       6 |       |       |          |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPT    |      1 |      4 |      4 |00:00:00.01 |       2 |       |       |          |
|   3 |    INDEX FULL SCAN           | PK_DEPT |      1 |      4 |      4 |00:00:00.01 |       1 |       |       |          |
|*  4 |   SORT JOIN                  |         |      4 |     14 |     14 |00:00:00.01 |       4 |  2048 |  2048 | 2048  (0)|
|   5 |    TABLE ACCESS FULL         | EMP     |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
------------------------------------------------------------------------------------------------------------
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			15	7
MERGE JOIN		OUTER	15	7
TABLE ACCESS	SCOTT.DEPT	BY INDEX ROWID	4	2
INDEX	SCOTT.PK_DEPT	FULL SCAN	4	1
SORT		JOIN	14	5
Access Predicates				
E.DEPTNO(+)=D.DEPTNO				
Filter Predicates				
E.DEPTNO(+)=D.DEPTNO				
TABLE ACCESS	SCOTT.EMP	FULL	14	4

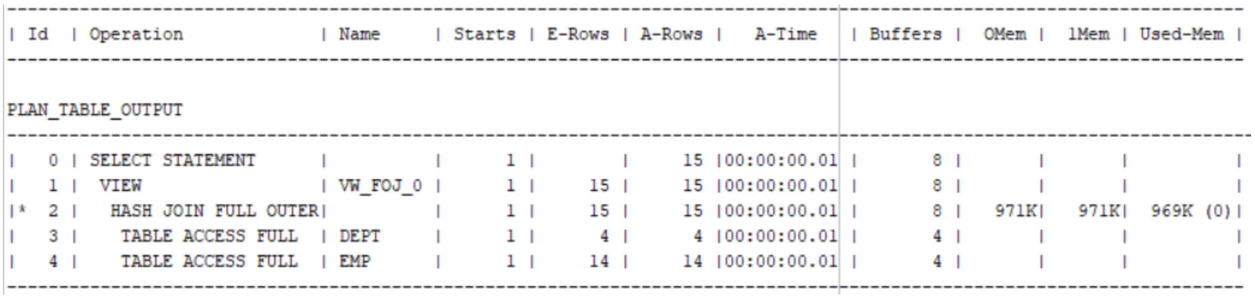
Picture 6.2 - Result

Task 7: Full Outer Join

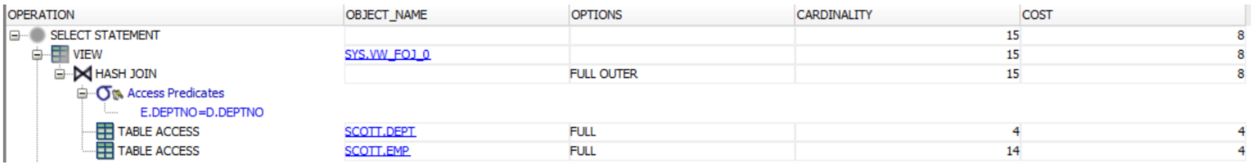
set autotrace on explain;

select /*+ gather_plan_statistics */ e.ename, e.deptno, e.job, d.dname

```
from scott.emp e

full outer join

scott.dept d

on(e.deptno = d.deptno);
```



```
------------------------------------------------------------------------------------------------------------------------
| Id  | Operation             | Name     | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
------------------------------------------------------------------------------------------------------------------------

PLAN_TABLE_OUTPUT
-------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |          |      1 |        |     15 |00:00:00.01 |       8 |       |       |          |
|   1 |  VIEW                 | VW_FOJ_0 |      1 |     15 |     15 |00:00:00.01 |       8 |       |       |          |
|*  2 |   HASH JOIN FULL OUTER|          |      1 |     15 |     15 |00:00:00.01 |       8 |   971K|   971K|  969K (0)|
|   3 |    TABLE ACCESS FULL  | DEPT     |      1 |      4 |      4 |00:00:00.01 |       4 |       |       |          |
|   4 |    TABLE ACCESS FULL  | EMP      |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
------------------------------------------------------------------------------------------------------------------------
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			15	8
VIEW	SYS.VW_FOJ_0		15	8
HASH JOIN		FULL OUTER	15	8
Access Predicates				
E.DEPTNO=D.DEPTNO				
TABLE ACCESS	SCOTT.DEPT	FULL	4	4
TABLE ACCESS	SCOTT.EMP	FULL	14	4

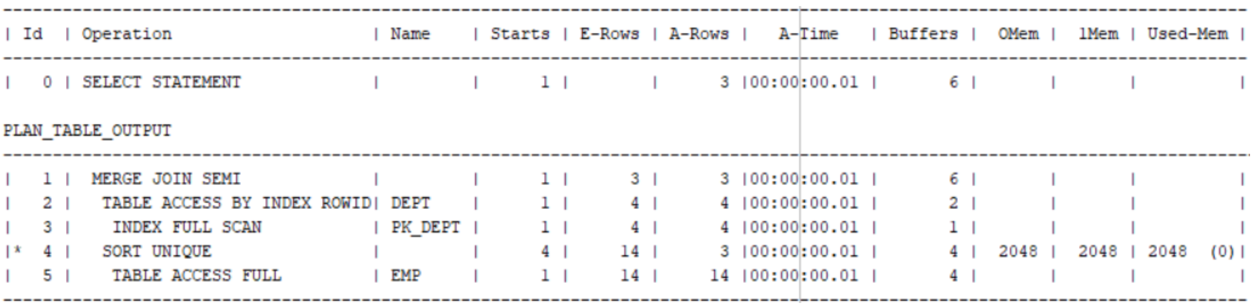
Picture 7.1 - Result

Note. We use Full outer join if we want the result set to include unmatched rows from both tables.

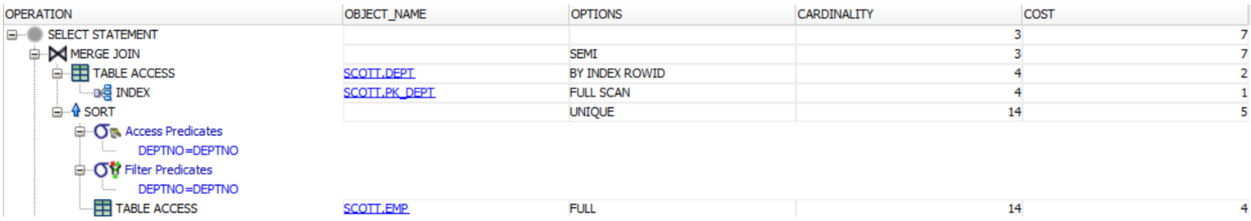
Task 8: Semi Joins

1. Using /*semijoin*/ :

```
set autotrace on explain;

select /*+  gather_plan_statistics semijoin */ dname

from scott.dept d

where deptno in (select deptno from scott.emp e);
```



```
---------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name    | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
---------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |         |      1 |        |      3 |00:00:00.01 |       6 |       |       |          |

PLAN_TABLE_OUTPUT
---------------------------------------------------------------------------------------------------------------------------
|   1 |  MERGE JOIN SEMI             |         |      1 |      3 |      3 |00:00:00.01 |       6 |       |       |          |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPT    |      1 |      4 |      4 |00:00:00.01 |       2 |       |       |          |
|   3 |    INDEX FULL SCAN           | PK_DEPT |      1 |      4 |      4 |00:00:00.01 |       1 |       |       |          |
|*  4 |   SORT UNIQUE                |         |      4 |     14 |      3 |00:00:00.01 |       4 |  2048 |  2048 | 2048  (0)|
|   5 |    TABLE ACCESS FULL         | EMP     |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
---------------------------------------------------------------------------------------------------------------------------
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			3	7
MERGE JOIN		SEMI	3	7
TABLE ACCESS	SCOTT.DEPT	BY INDEX ROWID	4	2
INDEX	SCOTT.PK_DEPT	FULL SCAN	4	1
SORT		UNIQUE	14	5
Access Predicates				
DEPTNO=DEPTNO				
Filter Predicates				
DEPTNO=DEPTNO				
TABLE ACCESS	SCOTT.EMP	FULL	14	4

Picture 8.1 -Query Result

Note. A semi-join returns one copy of each row in first table for which at least one match is found. Semi-joins are also written using the EXISTS construct.

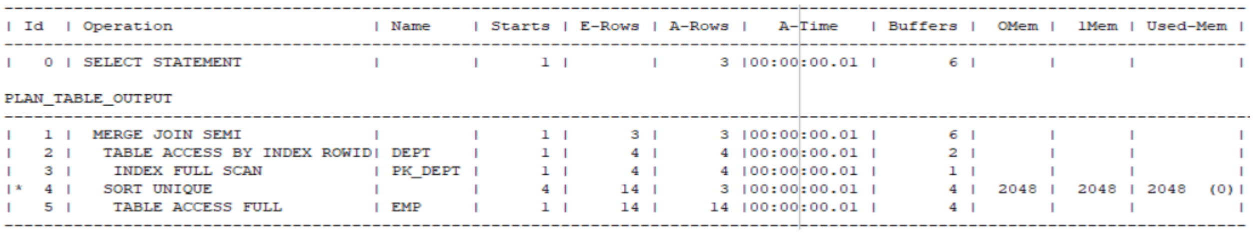
2. Using /*no_semijoin*/ :

set autotrace on explain;

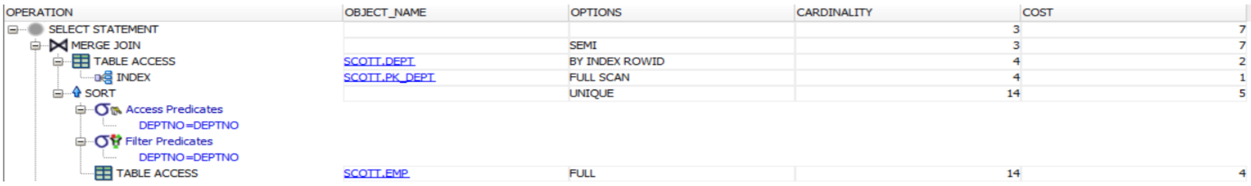
select /*+ gather_plan_statistics no_semijoin */ dname

from scott.dept d

where deptno in (select deptno from scott.emp e);



```
-------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name    | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
-------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |         |      1 |        |      3 |00:00:00.01 |       6 |       |       |          |

PLAN_TABLE_OUTPUT
-------------------------------------------------------------------------------------------------------------------------
|   1 |  MERGE JOIN SEMI              |         |      1 |      3 |      3 |00:00:00.01 |       6 |       |       |          |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPT    |      1 |      4 |      4 |00:00:00.01 |       2 |       |       |          |
|   3 |    INDEX FULL SCAN            | PK_DEPT |      1 |      4 |      4 |00:00:00.01 |       1 |       |       |          |
|*  4 |   SORT UNIQUE                 |         |      4 |     14 |      3 |00:00:00.01 |       4 |  2048 |  2048 | 2048  (0)|
|   5 |    TABLE ACCESS FULL          | EMP     |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
-------------------------------------------------------------------------------------------------------------------------
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			3	7
MERGE JOIN		SEMI	3	7
TABLE ACCESS	SCOTT.DEPT	BY INDEX ROWID	4	2
INDEX	SCOTT.PK_DEPT	FULL SCAN	4	1
SORT		UNIQUE	14	5
Access Predicates				
DEPTNO=DEPTNO				
Filter Predicates				
DEPTNO=DEPTNO				
TABLE ACCESS	SCOTT.EMP	FULL	14	4

Picture 8.2 -Query Result

Note. As a consequence, NO vital difference

3. Using /*using exists*/

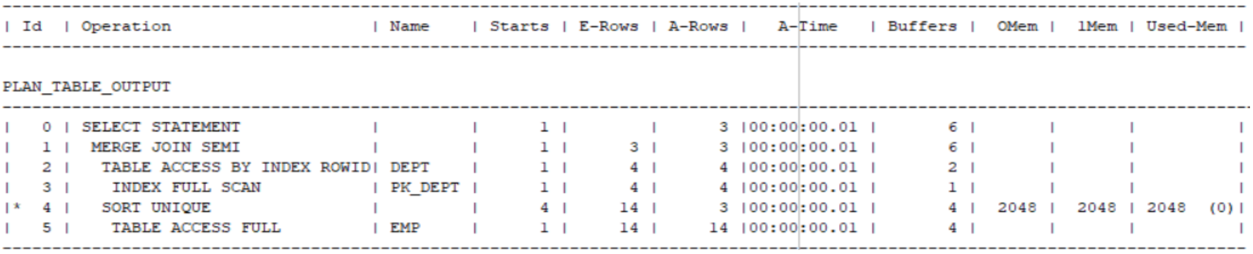
set autotrace on explain;

select /*+ gather_plan_statistics */ dname

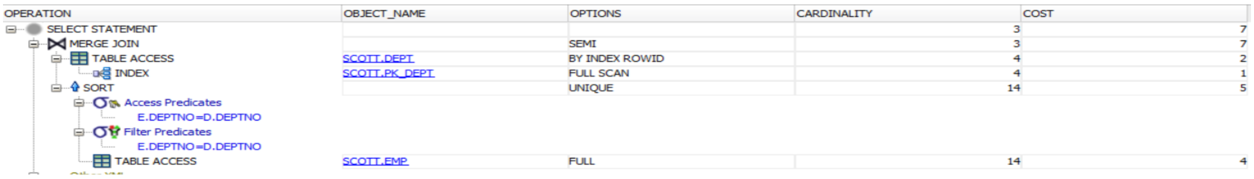
from scott.dept d

where exists (select null from scott.emp e

where e.deptno = d.deptno);



```
-----------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name    | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------

PLAN_TABLE_OUTPUT
-----------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |         |      1 |        |      3 |00:00:00.01 |       6 |       |       |          |
|   1 |  MERGE JOIN SEMI             |         |      1 |      3 |      3 |00:00:00.01 |       6 |       |       |          |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPT    |      1 |      4 |      4 |00:00:00.01 |       2 |       |       |          |
|   3 |    INDEX FULL SCAN           | PK_DEPT |      1 |      4 |      4 |00:00:00.01 |       1 |       |       |          |
|*  4 |   SORT UNIQUE                |         |      4 |     14 |      3 |00:00:00.01 |       4 |  2048 |  2048 | 2048  (0)|
|   5 |    TABLE ACCESS FULL         | EMP     |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------------
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			3	7
MERGE JOIN		SEMI	3	7
TABLE ACCESS	SCOTT.DEPT	BY INDEX ROWID	4	2
INDEX	SCOTT.PK_DEPT	FULL SCAN	4	1
SORT		UNIQUE	14	5
Access Predicates				
E.DEPTNO=D.DEPTNO				
Filter Predicates				
E.DEPTNO=D.DEPTNO				
TABLE ACCESS	SCOTT.EMP	FULL	14	4

Picture 8.3 -Query Result

Note. The EXISTS function checks to find a single matching row to return the result in a subquery. Because the IN function retrieves and checks all rows, so it is slower.

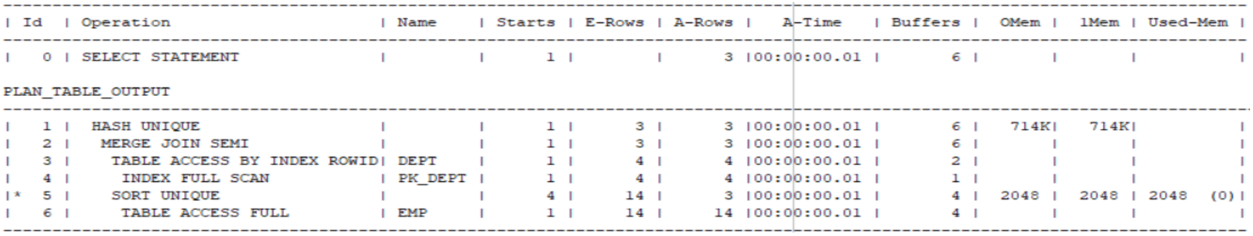
4. Select distinct using /* inner join with distinct */ hint

set autotrace on explain;

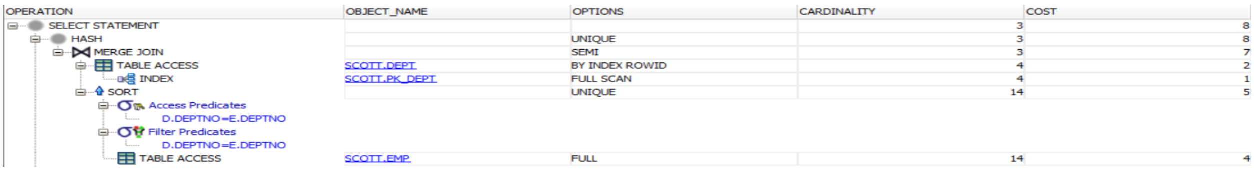
select /*+ gather_plan_statistics inner join with distinct */ distinct dname

from scott.dept d ,scott.emp e

where d.deptno = e.deptno ;



```
-----------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name    | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |         |      1 |        |      3 |00:00:00.01 |       6 |       |       |          |

PLAN_TABLE_OUTPUT
-----------------------------------------------------------------------------------------------------------------------------
|   1 |  HASH UNIQUE                  |         |      1 |      3 |      3 |00:00:00.01 |       6 |   714K|   714K|          |
|   2 |   MERGE JOIN SEMI             |         |      1 |      3 |      3 |00:00:00.01 |       6 |       |       |          |
|   3 |    TABLE ACCESS BY INDEX ROWID| DEPT    |      1 |      4 |      4 |00:00:00.01 |       2 |       |       |          |
|   4 |     INDEX FULL SCAN           | PK_DEPT |      1 |      4 |      4 |00:00:00.01 |       1 |       |       |          |
|*  5 |    SORT UNIQUE                |         |      4 |     14 |      3 |00:00:00.01 |       4 |  2048 |  2048 | 2048  (0)|
|   6 |     TABLE ACCESS FULL         | EMP     |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------------
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			3	8
HASH		UNIQUE	3	8
MERGE JOIN		SEMI	3	7
TABLE ACCESS	SCOTT.DEPT	BY INDEX ROWID	4	2
INDEX	SCOTT.PK_DEPT	FULL SCAN	4	1
SORT		UNIQUE	14	5
Access Predicates				
D.DEPTNO=E.DEPTNO				
Filter Predicates				
D.DEPTNO=E.DEPTNO				
TABLE ACCESS	SCOTT.EMP	FULL	14	4

Picture 8.4 -Query Result

Note. The DISTINCT clause is used in a SELECT statement to filter duplicate rows in the result set. This way we can ensure that returned rows are unique for the column or columns specified in the SELECT clause.

Task 9: Anti Joins

1. Not exists using /*not exists*/ hint

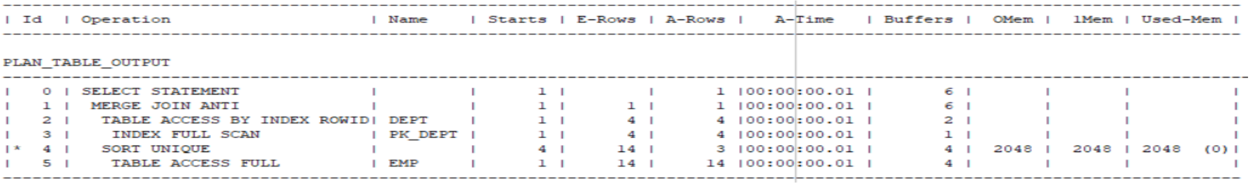
set autotrace on explain;

select /*+ gather_plan_statistics not exists ANTIJOIN */ dname

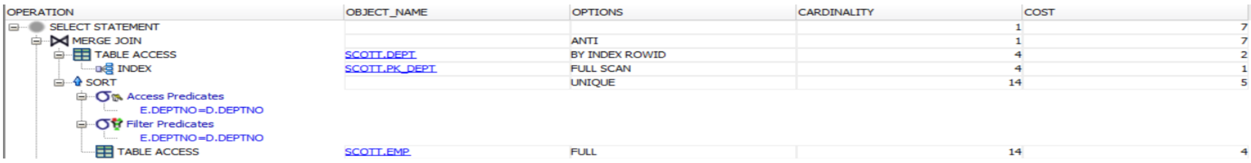
from scott.dept d

where not exists (select null from scott.emp e

where e.deptno = d.deptno);



```
-------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name    | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
-------------------------------------------------------------------------------------------------------------------------------

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |         |      1 |        |      1 |00:00:00.01 |       6 |       |       |          |
|   1 |  MERGE JOIN ANTI             |         |      1 |      1 |      1 |00:00:00.01 |       6 |       |       |          |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPT    |      1 |      4 |      4 |00:00:00.01 |       2 |       |       |          |
|   3 |    INDEX FULL SCAN           | PK_DEPT |      1 |      4 |      4 |00:00:00.01 |       1 |       |       |          |
|*  4 |   SORT UNIQUE                |         |      4 |     14 |      3 |00:00:00.01 |       4 |  2048 |  2048 | 2048  (0)|
|   5 |    TABLE ACCESS FULL         | EMP     |      1 |     14 |     14 |00:00:00.01 |       4 |       |       |          |
-------------------------------------------------------------------------------------------------------------------------------
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			1	7
MERGE JOIN		ANTI	1	7
TABLE ACCESS	SCOTT.DEPT	BY INDEX ROWID	4	2
INDEX	SCOTT.PK_DEPT	FULL SCAN	4	1
SORT		UNIQUE	14	5
Access Predicates				
E.DEPTNO=D.DEPTNO				
Filter Predicates				
E.DEPTNO=D.DEPTNO				
TABLE ACCESS	SCOTT.EMP	FULL	14	4

Note. The NOT EXISTS operator returns true if the subquery returns no row. Otherwise, it returns false. We should keep in mind that NOT EXISTS operator returns false if the subquery returns any rows with a NULL value**.**

2. Not in using /*not in*/ hint

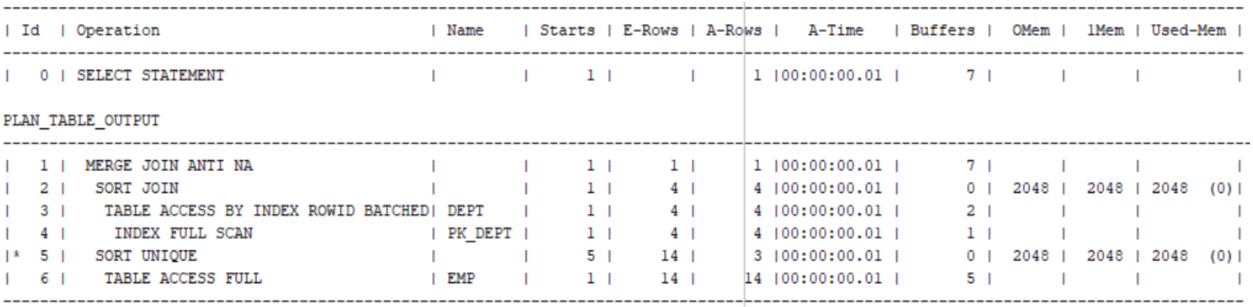
set autotrace on explain;

select /*+ gather_plan_statistics not in */ dname

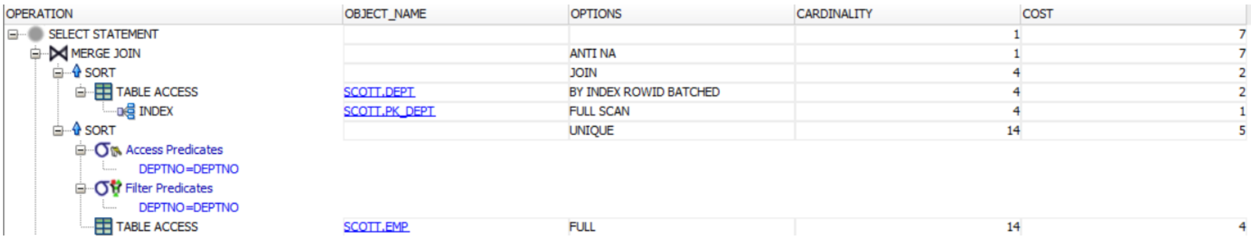
from scott.dept d

where deptno not in

(select deptno from scott.emp e)



```
--------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name    | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
--------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |         |      1 |        |      1 |00:00:00.01 |       7 |       |       |          |

PLAN_TABLE_OUTPUT
---------------------------------------------------------------------------------------------------------------------------------
|   1 |  MERGE JOIN ANTI NA           |         |      1 |      1 |      1 |00:00:00.01 |       7 |       |       |          |
|   2 |   SORT JOIN                   |         |      1 |      4 |      4 |00:00:00.01 |       0 |  2048 |  2048 | 2048  (0)|
|   3 |    TABLE ACCESS BY INDEX ROWID BATCHED| DEPT    |      1 |      4 |      4 |00:00:00.01 |       2 |       |       |          |
|   4 |     INDEX FULL SCAN           | PK_DEPT |      1 |      4 |      4 |00:00:00.01 |       1 |       |       |          |
|*  5 |   SORT UNIQUE                 |         |      5 |     14 |      3 |00:00:00.01 |       0 |  2048 |  2048 | 2048  (0)|
|   6 |    TABLE ACCESS FULL          | EMP     |      1 |     14 |     14 |00:00:00.01 |       5 |       |       |          |
--------------------------------------------------------------------------------------------------------------------------------
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			1	7
MERGE JOIN		ANTI NA	1	7
SORT		JOIN	4	2
TABLE ACCESS	SCOTT.DEPT	BY INDEX ROWID BATCHED	4	2
INDEX	SCOTT.PK_DEPT	FULL SCAN	4	1
SORT		UNIQUE	14	5
Access Predicates				
DEPTNO=DEPTNO				
Filter Predicates				
DEPTNO=DEPTNO				
TABLE ACCESS	SCOTT.EMP	FULL	14	4

Note. Using NOT IN operator can be risky because it does not return null values

NOT IN can be just as efficient as NOT EXISTS even if an anti-join can be used (if the subquery is known to not return nulls)

3. Minus using /*minus*/ hint

set autotrace on explain;

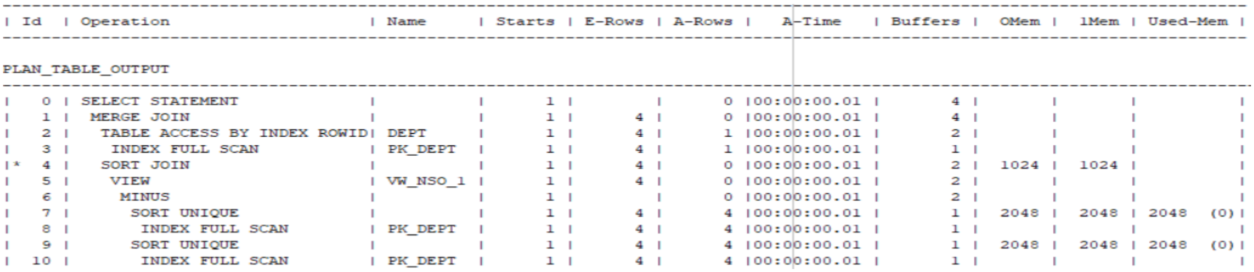
select /*+ gather_plan_statistics minus*/ dname

from scott.dept

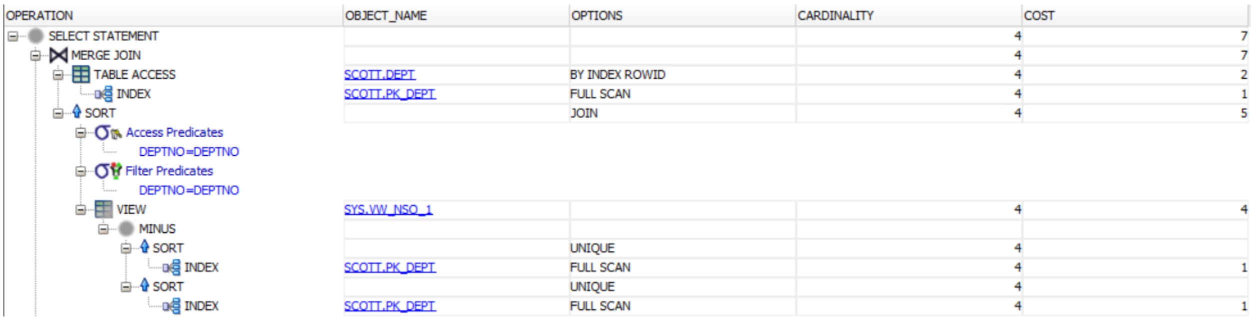
where deptno in

(select deptno from scott.dept minus

select deptno from scott.dept);



```
-----------------------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name     | Starts | E-Rows | A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------

PLAN_TABLE_OUTPUT
-----------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |          |      1 |        |      0 |00:00:00.01 |       4 |       |       |          |
|   1 |  MERGE JOIN                  |          |      1 |      4 |      0 |00:00:00.01 |       4 |       |       |          |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPT     |      1 |      4 |      1 |00:00:00.01 |       2 |       |       |          |
|   3 |    INDEX FULL SCAN           | PK_DEPT  |      1 |      4 |      1 |00:00:00.01 |       1 |       |       |          |
|*  4 |   SORT JOIN                  |          |      1 |      4 |      0 |00:00:00.01 |       2 |  1024 |  1024 |          |
|   5 |    VIEW                      | VW_NSO_1 |      1 |      4 |      0 |00:00:00.01 |       2 |       |       |          |
|   6 |     MINUS                    |          |      1 |        |      0 |00:00:00.01 |       2 |       |       |          |
|   7 |      SORT UNIQUE             |          |      1 |      4 |      4 |00:00:00.01 |       1 |  2048 |  2048 | 2048  (0)|
|   8 |       INDEX FULL SCAN        | PK_DEPT  |      1 |      4 |      4 |00:00:00.01 |       1 |       |       |          |
|   9 |      SORT UNIQUE             |          |      1 |      4 |      4 |00:00:00.01 |       1 |  2048 |  2048 | 2048  (0)|
|  10 |       INDEX FULL SCAN        | PK_DEPT  |      1 |      4 |      4 |00:00:00.01 |       1 |       |       |          |
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			4	7
MERGE JOIN			4	7
TABLE ACCESS	SCOTT.DEPT	BY INDEX ROWID	4	2
INDEX	SCOTT.PK_DEPT	FULL SCAN	4	1
SORT		JOIN	4	5
Access Predicates				
DEPTNO=DEPTNO				
Filter Predicates				
DEPTNO=DEPTNO				
VIEW	SYS.VW_NSO_1		4	4
MINUS				
SORT		UNIQUE	4	
INDEX	SCOTT.PK_DEPT	FULL SCAN	4	1
SORT		UNIQUE	4	
INDEX	SCOTT.PK_DEPT	FULL SCAN	4	1

Note. The Oracle MINUS operator is used to return all rows in the first SELECT statement that are not returned by the second SELECT statement. Each SELECT statement will define a dataset. The MINUS operator retrieve all records from the first dataset and then remove from the results all records from the second dataset.

4. Left outer anti join

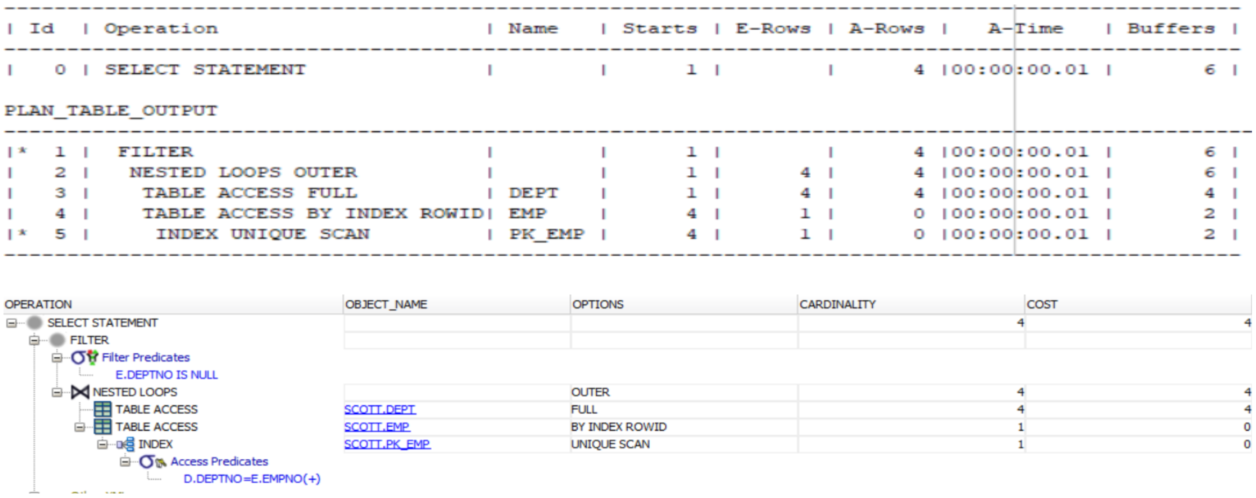
set autotrace on explain;

select /*+ gather_plan_statistics left outer */ dname

from scott.dept d, scott.emp e

where d.deptno = e.empno(+)

and e.deptno is null;



```
-----------------------------------------------------------------------------------------------
| Id  | Operation                    | Name    | Starts | E-Rows | A-Rows |   A-Time   | Buffers |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |         |      1 |        |      4 |00:00:00.01 |       6 |

PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------------------------
|*  1 |  FILTER                      |         |      1 |        |      4 |00:00:00.01 |       6 |
|   2 |   NESTED LOOPS OUTER         |         |      1 |      4 |      4 |00:00:00.01 |       6 |
|   3 |    TABLE ACCESS FULL         | DEPT    |      1 |      4 |      4 |00:00:00.01 |       4 |
|   4 |    TABLE ACCESS BY INDEX ROWID| EMP    |      4 |      1 |      0 |00:00:00.01 |       2 |
|*  5 |     INDEX UNIQUE SCAN        | PK_EMP |      4 |      1 |      0 |00:00:00.01 |       2 |
-----------------------------------------------------------------------------------------------
```

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			4	4
FILTER				
Filter Predicates				
E.DEPTNO IS NULL				
NESTED LOOPS		OUTER	4	4
TABLE ACCESS	SCOTT.DEPT	FULL	4	4
TABLE ACCESS	SCOTT.EMP	BY INDEX ROWID	1	0
INDEX	SCOTT.PK_EMP	UNIQUE SCAN	1	0
Access Predicates				
D.DEPTNO=E.EMPNO(+)				

Picture 9.1 -Query Result

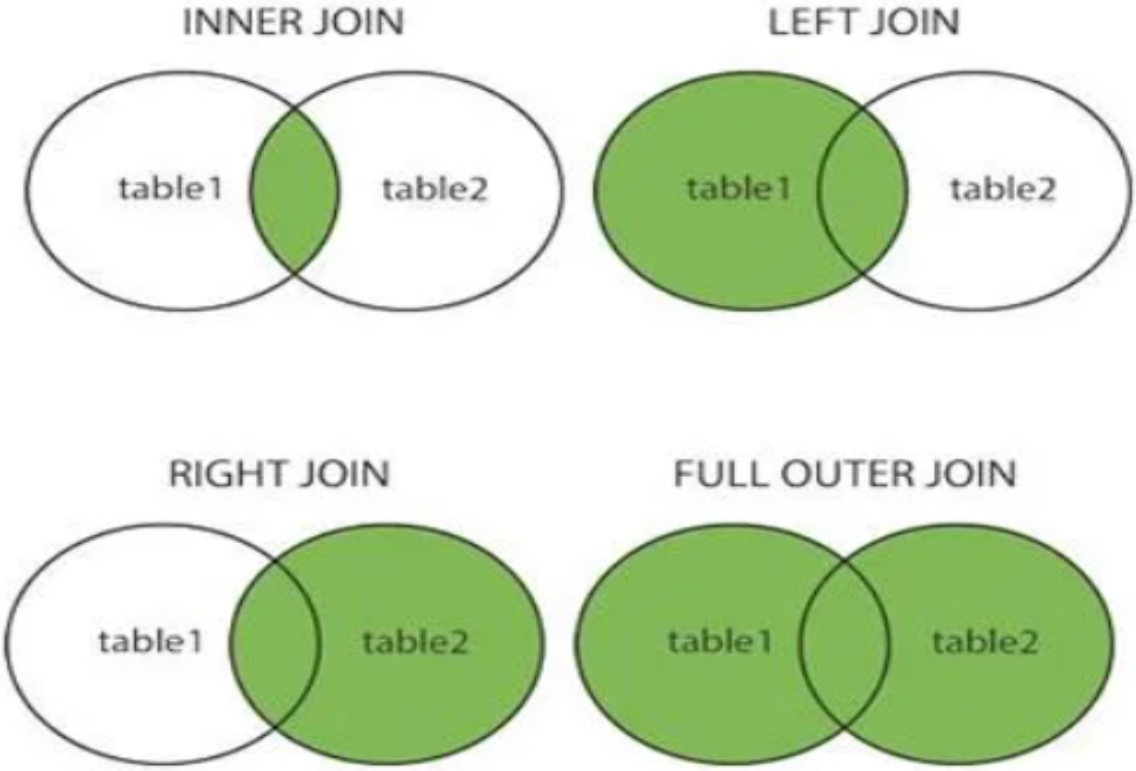
Task 10: Summary table

<table>
<tr><th>Join Access “A”</th><th>Join Access “B”</th><th>Nested Loop</th><th>Hash Join</th><th>Sort-Merge Join</th><th>Anti-Join</th><th>Semi-Join</th></tr>
<tr><td>Small Table</td><td>Small Table</td><td rowspan="3">Good</td><td rowspan="3">Ineffective</td><td rowspan="2">Ineffective</td><td rowspan="7" colspan="2">Contextual</td></tr>
<tr><td>Small Table</td><td>Small Table(Indexed)</td></tr>
<tr><td>Small Table(Indexed)</td><td>Small Table(Indexed)</td><td>Good</td></tr>
<tr><td>Big table</td><td>Big table</td><td rowspan="4">Awful</td><td rowspan="3">Good</td><td rowspan="2">Ineffective</td></tr>
<tr><td>Big table(Indexed)</td><td>Big table</td></tr>
<tr><td>Big table(Indexed)</td><td>Big table(Indexed)</td><td>Good</td></tr>
<tr><td>Small</td><td>Big</td><td colspan="2">Ineffective</td></tr>
</table>

Laboratory work summary:

We used many join methods to see which has better performance in different situations.

At the table above and insights from each task (or sub – task) you can see some results and conclusion.


INNER JOIN
table1
table2
LEFT JOIN
table1
table2
RIGHT JOIN
table1
table2
FULL OUTER JOIN
table1
table2