Labwork5

|  |  |  |
| --- | --- | --- |
| № | Auto Trace Configuration Options | Expected Results |
| 1 | Set autotrace off | Result Set |
| 2 | set autotrace on | Result Set, Execution Plan,  Statistics |
| 3 | set autotrace traceonly | Execution Plan, Statistics |
| 4 | set autotrace on explain | Result Set, Execution Plan |
| 5 | set autotrace on statistics | Result Set, Statistics |
| 6 | set autotrace on explain statistics | Result Set, Execution Plan,  Statistics |
| 7 | set autotrace traceonly explain | Execution Plan |
| 8 | set autotrace traceonly statistics | Statistics |
| 9 | set autotrace traceonly explain statistics | Execution Plan, Statistics |
| 10 | set autotrace off explain | Result Set |
| 11 | set autotrace off statistics | Result Set |
| 12 | set autotrace off explain statistics | Result Set |

**Task1**

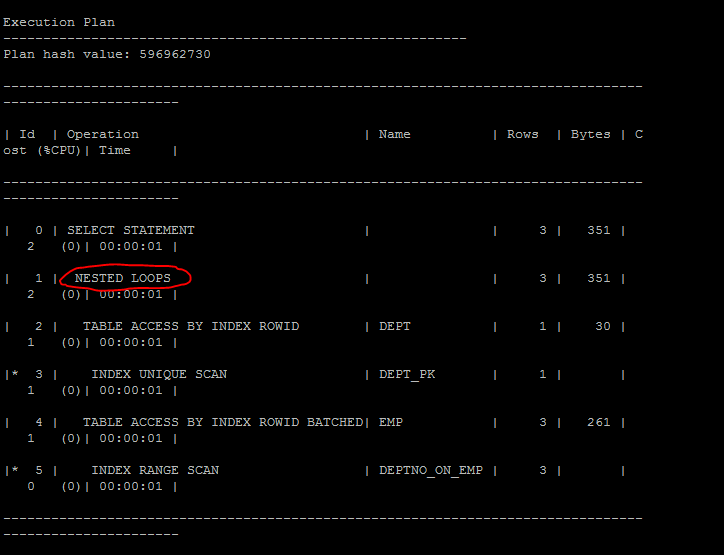
Nested loops joins use each row of the query result reached through one access operation to drive into another table. These joins are typically most effective if the result set is limited in size and indexes are present on the columns used for the join.

SELECT \*

FROM scott.emp e, scott.dept d

WHERE e.deptno = d.deptno

AND d.deptno = 10



**Task2**

Sort-merge joins read the two tables to be joined independently, sorts the rows from each table (but only those rows that meet the conditions for the table in the WHERE clause) in order by the join key, and then merges the sorted rowsets.

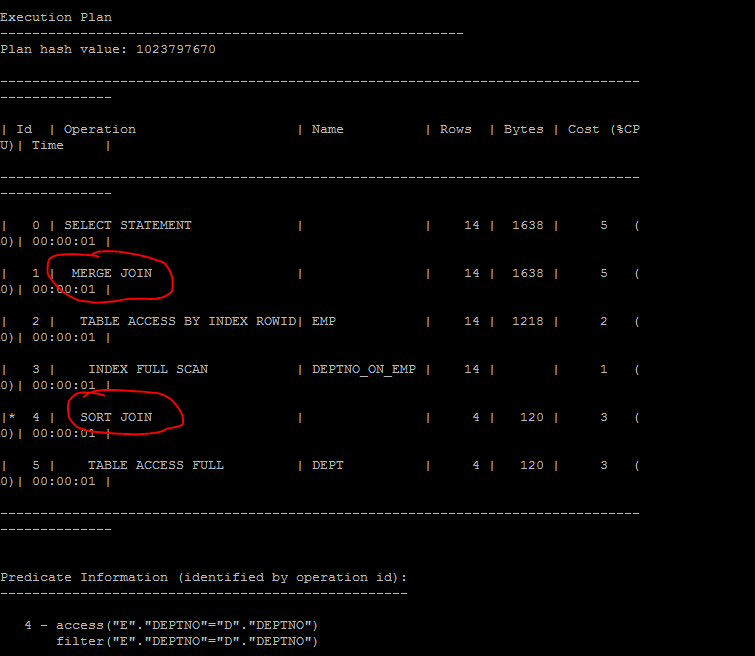
The sort operations are the expensive part for this join method.

SELECT \* FROM scott.emp e

Inner join scott.dept d

on e.deptno = d.deptno

where d.deptno = 10



**Task3**

Hash joins, like sort-merge joins, first reads the two tables to be joined independently and applies the criteria in the WHERE clause. Based on table and index statistics, the table that is determined to return the fewest rows will be hashed in its entirety into memory. This hash table includes all the row data for that table and is loaded into hash buckets based on a randomizing function that converts the join key to a hash value.

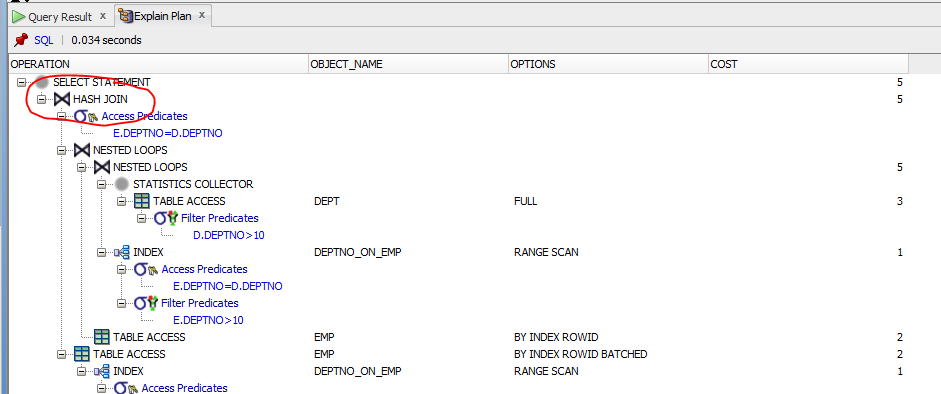
SELECT \*

FROM scott.emp e

join scott.dept d

on e.deptno = d.deptno

where d.deptno>10

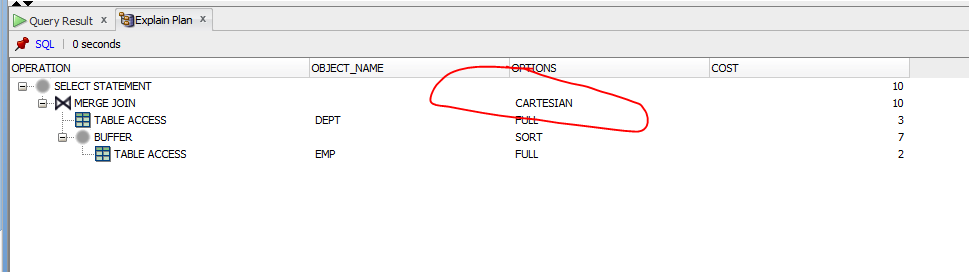


**Task4**

Cartesian joins occur when all the rows from one table are joined to all the rows of another table.

SELECT \*

FROM scott.emp e,scott.dept d



**Task5**

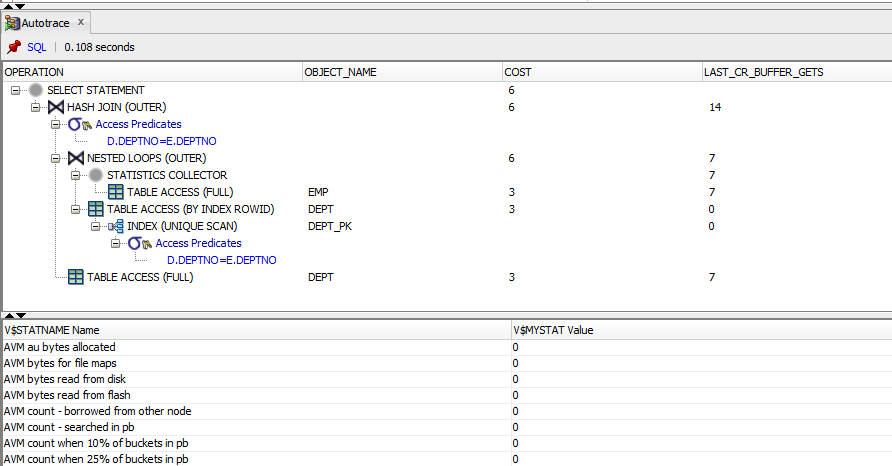
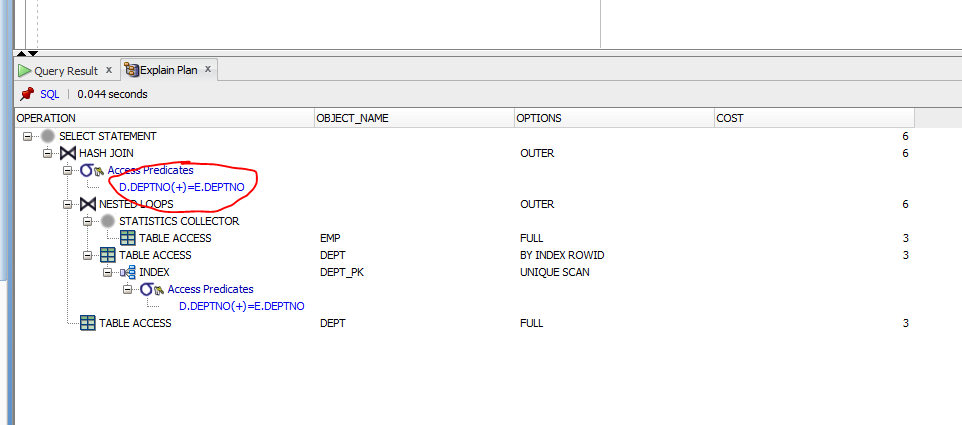
An outer join returns all rows from one table and only those rows from the joined table where the join condition is met.

SELECT \*

FROM scott.emp e

left outer join scott.dept d

on d.deptno=e.deptno

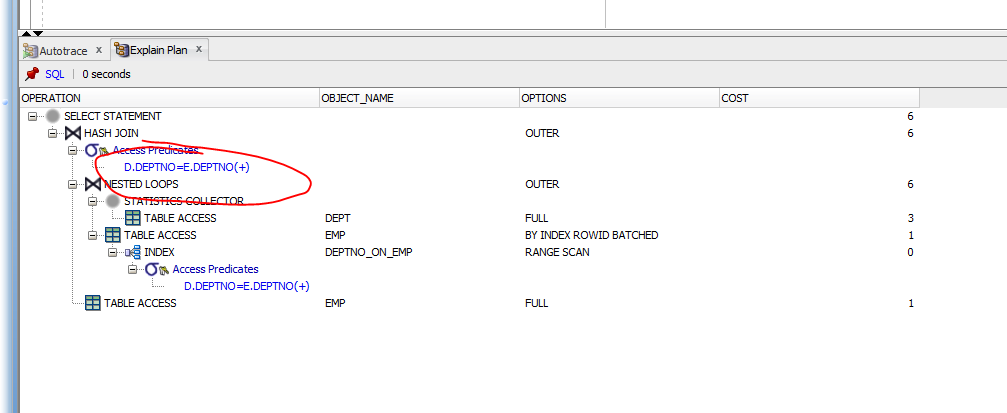


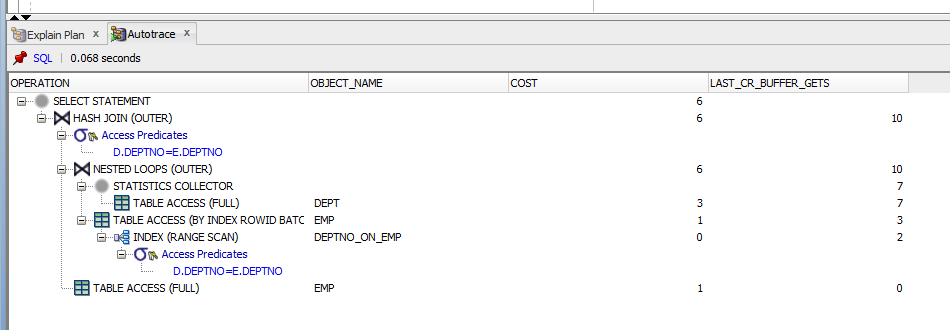
SELECT \*

FROM scott.emp e

right outer join scott.dept d

on d.deptno=e.deptno





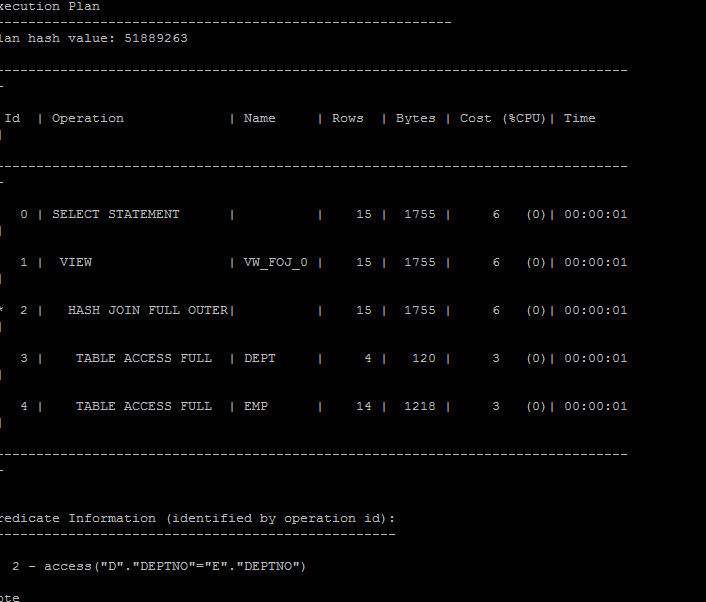
**Task6**

SELECT \*

FROM scott.emp e

full outer join scott.dept d

on d.deptno=e.deptno

A full outer join will join two tables from left-to-right and right-to-left.

**Task7**

A semi-join is a join between two sets of data (tables) where rows from the first set are returned, based on the presence or absence of at least one matching row in the other set.

SELECT \*

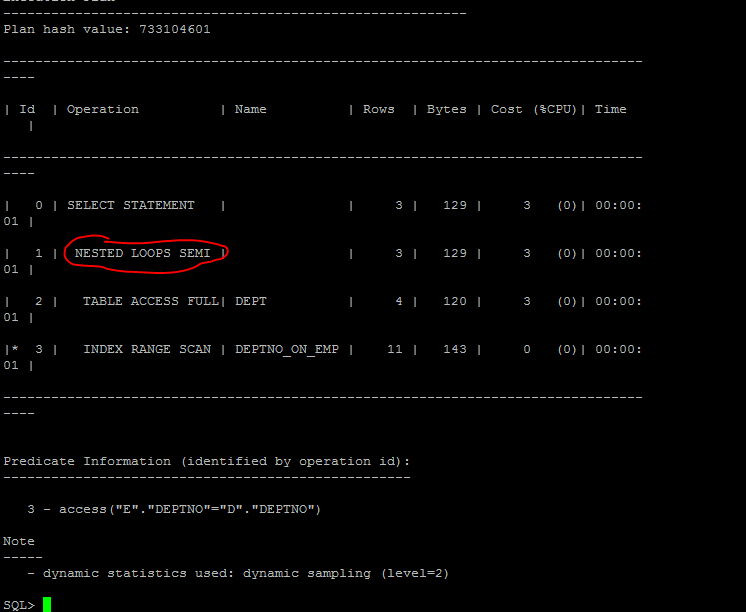
FROM scott.dept d

WHERE EXISTS (

SELECT \*

FROM scott.emp e

WHERE e.deptno = d.deptno);



SELECT \*

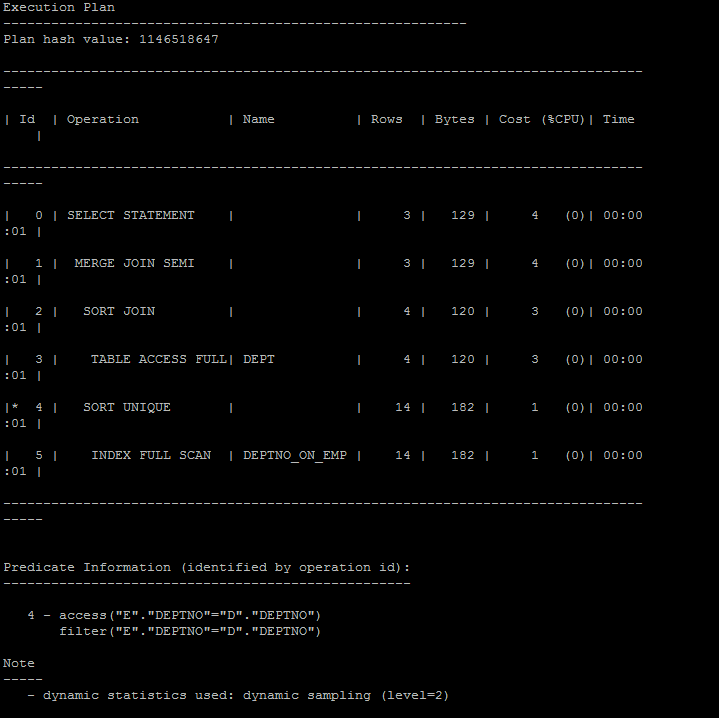
FROM scott.dept d

WHERE EXISTS (

SELECT /\*+ MERGE\_SJ\*/ \*

FROM scott.emp e

WHERE e.deptno = d.deptno);



SELECT \*

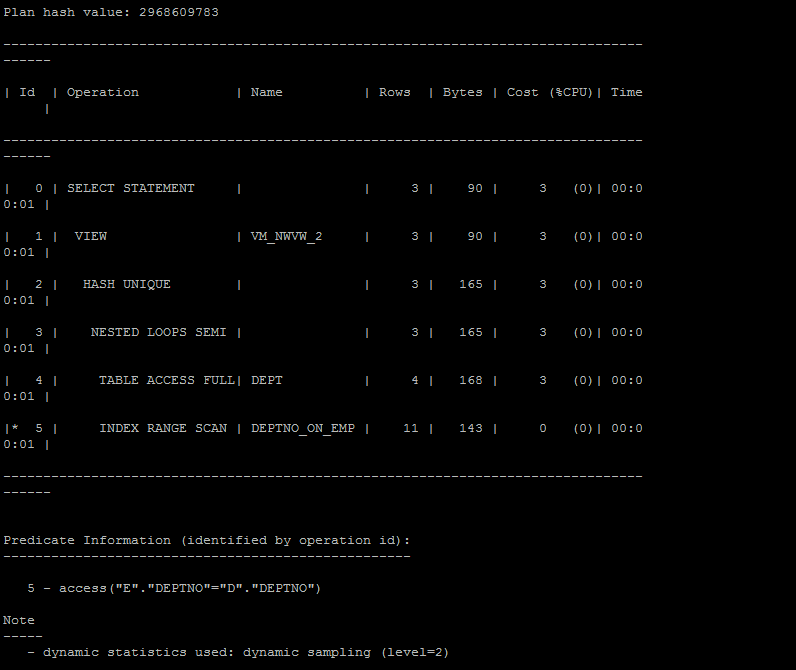
FROM scott.dept d

WHERE EXISTS (

SELECT /\*+ NO\_SEMIJOIN \*/ \*

FROM scott.emp e

WHERE e.deptno = d.deptno);



SELECT \*

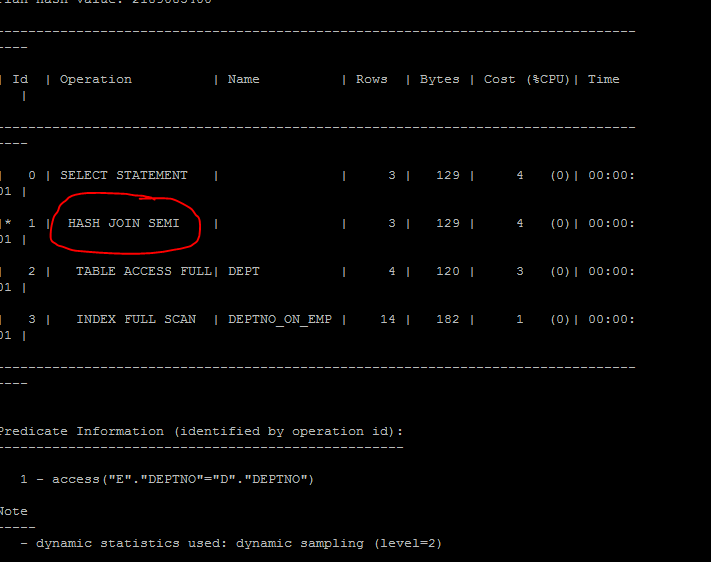
FROM scott.dept d

WHERE EXISTS (

SELECT /\*+ HASH\_SJ \*/ \*

FROM scott.emp e

WHERE e.deptno = d.deptno);



**Task8**

Anti-joins are basically the same as semi-joins in that they are an optimization option that can be applied to nested loop, hash, and merge joins. However, they are the opposite of semi-joins in terms of the data they return. Those mathematician types familiar with relational algebra would say that anti joins can be defined as the complement of semi-joins.

SELECT \*

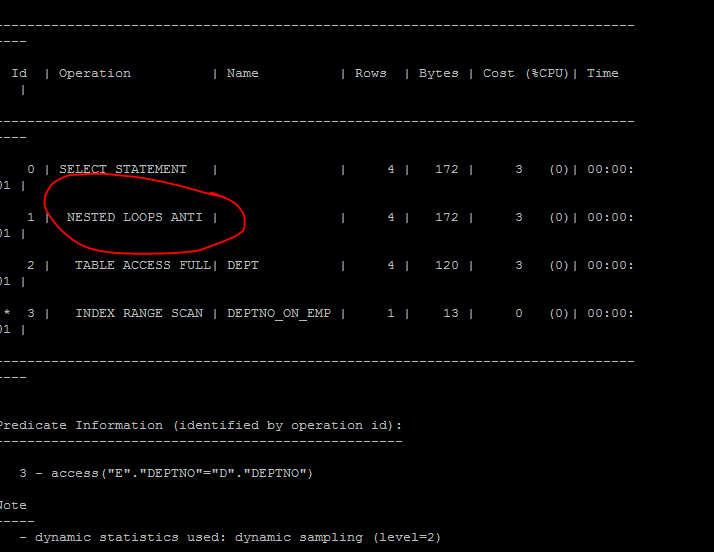
FROM scott.dept d

WHERE not EXISTS (

SELECT \*

FROM scott.emp e

WHERE e.deptno = d.deptno);



**Task10**

Marks

v– very good performance;

g – good performance;

b – bad performance;

w – the worst performance.

B - Big\_Table

S – Small\_Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Join Access “A”** | **Join Access “B”** | **Nested Loop** | **Hash Join** | **Sort-Merge Join** | **Anti-Join** | **Semi-Join** |
| Heap Table (S) | Heap Table (S) | v | g | g | v | v |
| Heap Table (S) | Heap Indexed Table (S) | v | v | g | v | v |
| Heap Indexed Table (S) | Heap Indexed Table (S) | v | v | v | v | v |
| Heap Table (B) | Heap Table (B) | w | b | w | w | w |
| Heap Table (B) | Heap Indexed Table (B) | b | b | g | w | w |
| Heap Indexed Table (B) | Heap Indexed Table (B) | g | b | v | w | w |
| Heap Table (S) | Heap Table (B) | b | v | b | g | g |
| Heap Table (S) | Heap Indexed Table (B) | g | v | g | g | g |
| Heap Indexed Table (S) | Heap Table (B) | b | v | g | g | g |
| Heap Indexed Table (B) | Heap Indexed Table (S) | g | v | v | g | g |