**Task 5 by Dzmitry Shautsou**

# Auto Trace & Explain Plan

## Task 1: Auto Trace configuration training

|  |  |  |  |
| --- | --- | --- | --- |
| № | Auto Trace Configuration Options | Expected Results | Description |
|  | set autotrace off | Shows query result | Disables all autotrace |
|  | set autotrace on | Shows query result, execution plan, statistics of the statement. | Shows the execution plan as well as statistics of the statement. |
|  | set autotrace traceonly | Shows execution plan, statistics of the statement. |  |
|  | set autotrace on explain | Displays the execution plan only. |  |
|  | set autotrace on statistics | Displays the statistics only |  |
|  | set autotrace on explain statistics | Shows results, execution plan, statistic |  |
|  | set autotrace traceonly explain | Shows execution plan |  |
|  | set autotrace traceonly statistics | Shows statistic |  |
|  | set autotrace traceonly explain statistics | Shows execution plan and statistic |  |
|  | set autotrace off explain | Displays query result |  |
|  | set autotrace off statistics | Displays query result |  |
|  | set autotrace off explain statistics | Displays query result |  |

# Join Methods

## Task 2: Nested Loops Joins

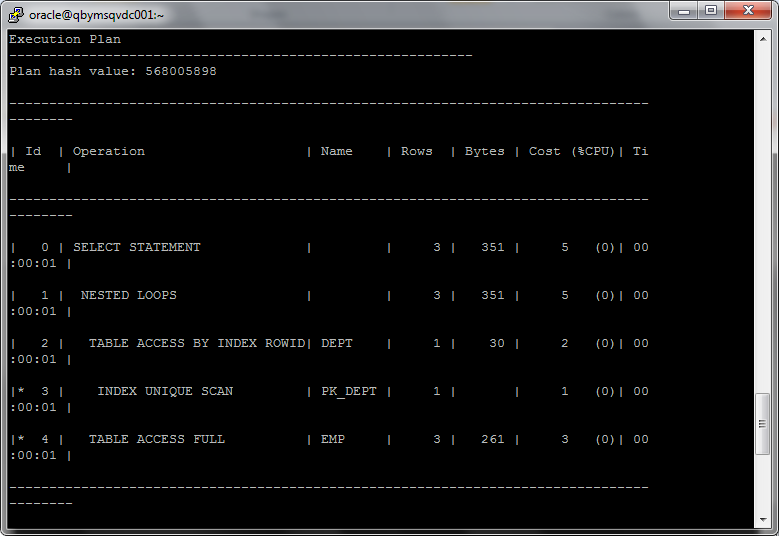
**Example:**

# SELECT \*

FROM scott.emp e, scott.dept d

WHERE e.deptno = d.deptno

AND d.deptno = 10



## Task 3: Sort-Merge Joins

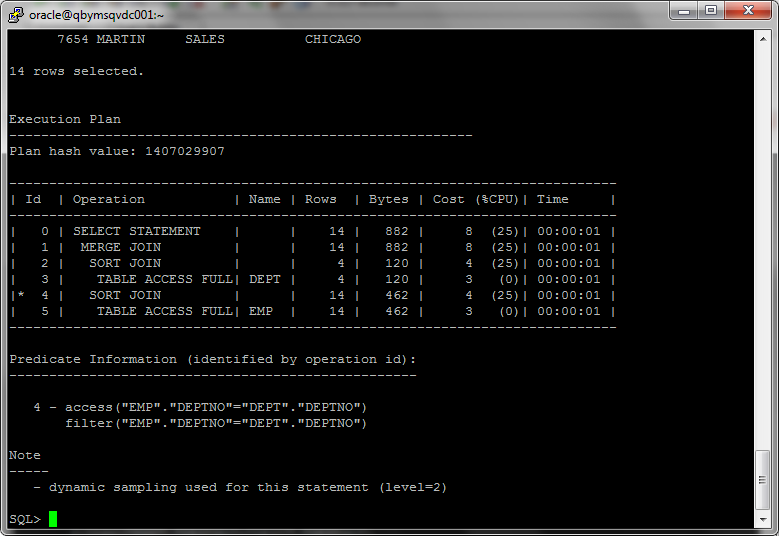
**Task:** Prepare SQL **explain plan** using: SQL plus Auto Trace Utility.

**Note:**  If you would like change in execution plan the type of join method use oracle performance hints. (USE\_MERGE)

select /\*+ USE\_MERGE (dept emp) \*/ empno, ename, dname, loc

from scott.dept, scott.emp

where emp.deptno = dept.deptno;



## Task 4: Hash Joins

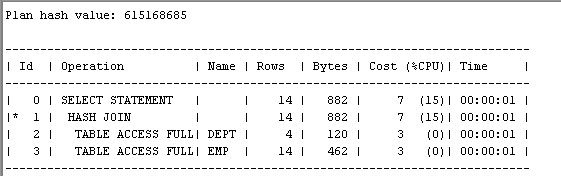
**Task:** Prepare SQL **explain plan** using software: Oracle SQL Developer.

**Note:**  If you would like change in execution plan the type of join method use oracle performance hints. (USE\_HASH)

select /\*+ USE\_HASH \*/ empno, ename, dname, loc

from scott.dept, scott.emp

where emp.deptno = dept.deptno;

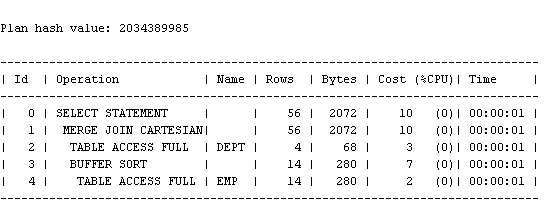


## Task 5: Cartesian Joins

**Task:** Prepare SQL **explain plan** using software: Oracle SQL Developer.

select empno, ename, dname, loc

from scott.dept, scott.emp;



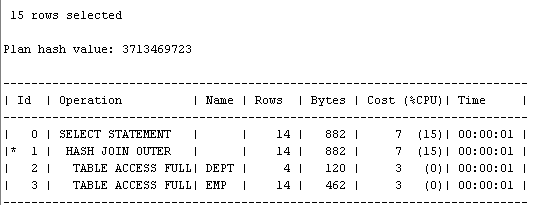
## Task 6: Left/Right Outer Joins

**Tasks:**

* Prepare SQL **trace protocol** using software: Oracle SQL Developer.
* Create SQL left outer join

select empno, ename, dname, loc

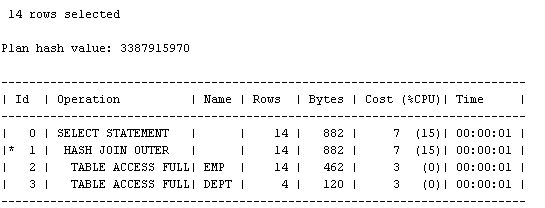
from scott.dept left join scott.emp on emp.deptno = dept.deptno;



* Create SQL right outer join

select empno, ename, dname, loc

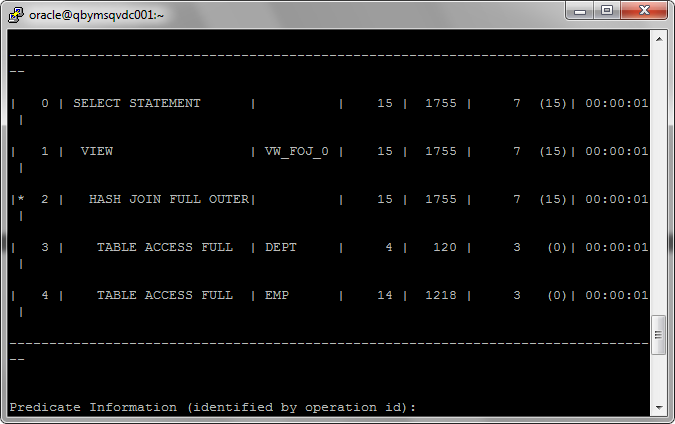
from scott.dept right join scott.emp on emp.deptno = dept.deptno;



## Task 7: Full Outer Join

**Task:** Prepare SQL **explain plan** using: SQL plus Auto Trace Utility.

select \* from scott.emp full outer join scott.dept on (emp.deptno = dept.deptno);



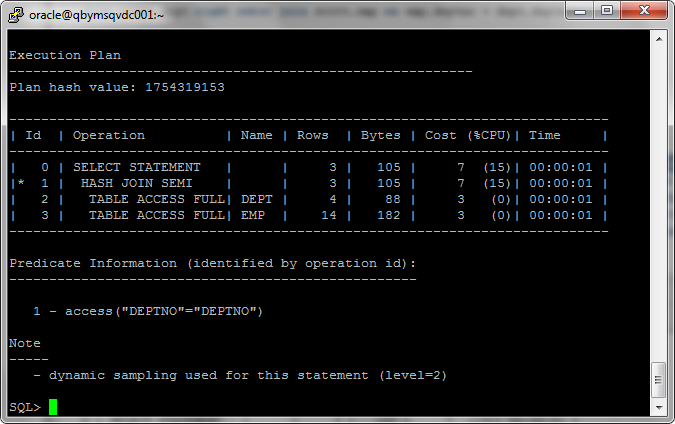
## Task 8: Semi Joins

**Task:** Prepare All possible variants of SEMI JOIN SQL **explain plan** using: SQL plus Auto Trace Utility.

**Note:**  If you would like change in execution plan the type of join method use oracle performance hints.

1. SEMIJOIN – perform a semi-join (the optimizer gets to pick which kind)
2. NO\_SEMIJOIN – obviously means don’t perform a semi-join
3. NL\_SJ – perform a nested loops semi-join (deprecated as of 10g)
4. HASH\_SJ – perform a hash semi-join (deprecated as of 10g)
5. MERGE\_SJ – perform a merge semi-join (deprecated as of 10g)

SELECT /\*+ SEMIJOIN (dept emp)\*/DName FROM SCOTT.dept dept WHERE deptno IN (SELECT deptno FROM scott.emp );



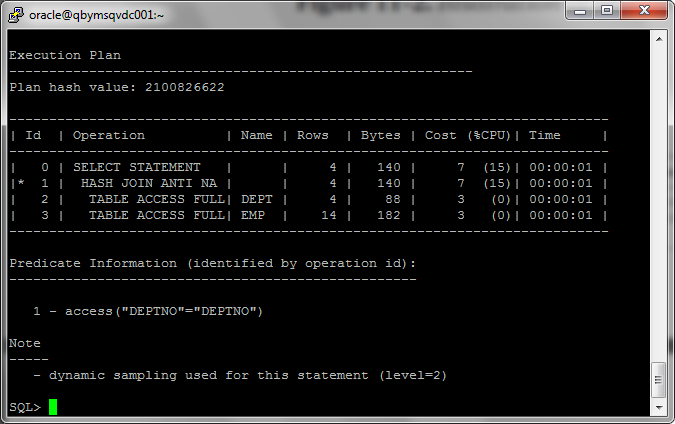
## Task 9: Anti Joins

**Task:** Prepare All possible variants of ANTI JOIN SQL **explain plan** using: SQL plus Auto Trace Utility.

**Note:**  If you would like change in execution plan the type of join method use oracle performance hints.

1. ANTIJOIN – perform an anti-join (the optimizer gets to pick which kind)
2. USE\_ANTI – older version of ANTIJOIN hint
3. NL\_AJ – perform a NESTED LOOPS anti-join (deprecated as of 10g)
4. HASH\_AJ – perform a HASH anti-join (deprecated as of 10g)
5. MERGE\_AJ – perform a MERGE anti-join (deprecated as of 10g)

SELECT/\*+ ANTIJOIN (dept emp)\*/ DName FROM SCOTT.dept dept WHERE deptno NOT IN ( SELECT deptno FROM scott.emp );



## Task 10: Prepare summary table

**Task:** Make comparison of all possible variant of join methods and join access methods and fill the table below:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Join Access “A” | Join Access “B” | Nested Loop | Hash Join | Sort-Merge Join | Anti-Join | Semi-Join |
| Small Table | Small Table | Хорошо подходит для небольших таблиц |  |  |  |  |
| Small Table | Indexed Small Table | Ещё лучше подходит для таблиц с индексом |  | Индекс по сортируемому полю значительно ускоряет выполнение запроса. |  |  |
| Large Table | Large Table | Низкая производительность для больших таблиц |  |  |  |  |
| Large Table | Indexed Large Table |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

(Заполнять в таблице по-моему крайне не удобно. Поэтому пишу так)

Nested Loop – отличный вариант для небольшого набора данных. С большими таблицами работает очень плохо. Желательно чтобы таблицы были проиндексированы, иначе работает слабенько.

Hash Join – требует много памяти для хранения хэш таблиц. Но оправдывает себя на больших объёмах данных, хорошо работает с большими, не отсортированными, не проиндексированными таблицами. Кроме того распараллеливается лучше остальных.

Sort Merge Join – требует индекса по полю соединения. Сортировка больших объёмов данных занимает много места в памяти, что может привести к свопу. Лучше всего будет работать в кластерном индексе.

Крайне желателен уникальный индекс.

Как я понял эти три основные метода «обращения» к соединяемым данным. А Outer, Inner, Semi, Anti и т.д. реализуют логику по которой будут отбираться данные . Например “HASH JOIN ANTI” и т.п.  
Следовательно все эти виды наследуют тип работы с данными от Nested Loop/ Hash Join/ Merge Join.