Data Science and Database Technology

Practice #5 – Oracle Optimizer

Practice objective

Generate the execution plan for some SQL statements analyzing the following issues:

1. access paths
2. join orders and join methods
3. operation orders
4. exploitation of indexes defined by the user.

The evaluation will be performed using Oracle Database 10g Express Edition (Oracle XE).

Database schema

The database consists of 3 tables: (EMP, DEPT e SALGRADE). The table schema and some records are shown in the following.

Table ***EMP***

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT NO
1	COZZA MARIA	PROFESSOR	0	09-JUN-81	1181		126
2	ECO LUIGI	PHDSTUDENT	0	09-JUN-81	1360		189
3	CORONA CLARA	PHDSTUDENT	2	09-JUN-81	624		15

Table ***DEPT***

DEPTNO	DNAME
1	*INFORMATION*
2	*CHAIRMANSHIP*
3	*ENVIRONMENT*
4	*PHYSICS*

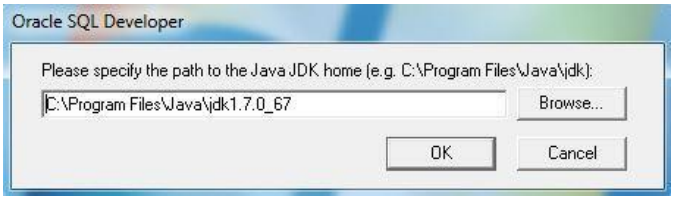
Table ***SALGRADE***

GRADE	LOSAL	HISAL
1	*478*	*1503*
2	*661*	*1346*
3	*489*	*1358*
4	*942*	*1320*

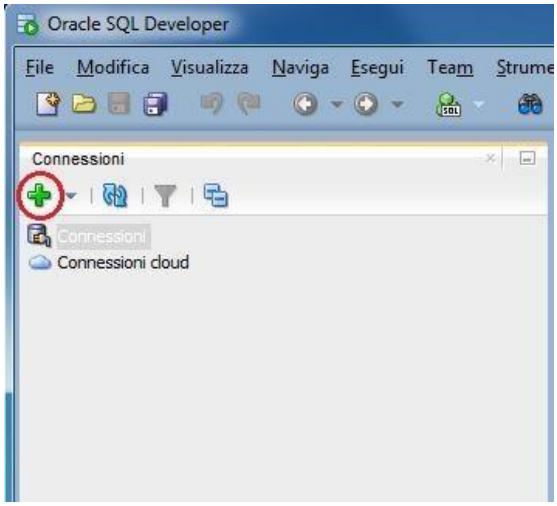
Preliminary steps to perform the practice:

Connection to the database

Open the Oracle SQL Developer program (from Start Menu-All programs) Select the Java SDK path



Click on the green "plus" botton on the left to create a new connection

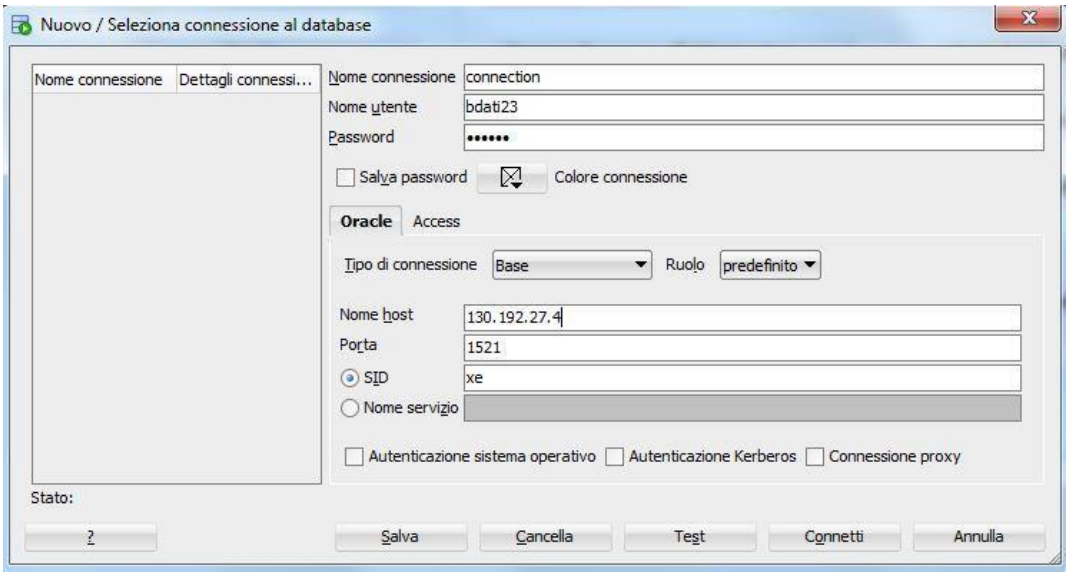


Login

To logon through the Web interface, you have to insert the following parameters:

- Nome utente (username): bdati[choose a number between 1-100]
- Password: orac[choose a number between 1-100]
- Nome host (host name): 130.192.27.4
- Port: 1521
- SID: xe

For example, if you are working on pc number 23, the corresponding username is **`bdati23`** and the password is **`orac23`**.



Available materials

Some scripts with SQL statements are available to perform the following operations:

1. create an index on a table column
2. compute statistics for the database

The scripts are available at the course website in the `Scripts.zip` archive

http://dbdmg.polito.it/wordpress/teaching/database-management-systems/

The scripts can be loaded clicking on "Open" in the File Menu and selecting the .sql file. To execute the script click on the "Esegui Script" button as shown in the following figure.

To view the index statistics, execute the script show_indexes.sql (or copy the script content and paste it as SQL command).

Setting up the optimizer environment

At the beginning of working session you need to perform the following steps:

1. compute statistics on tables by means of the Web Interface or by the following script `comp_statistics_tables.sql`
2. check if there exist secondary indexes by means of the following SQL query `select INDEX_NAME from USER_INDEXES;`
   if secondary indexes (without considering system indexes, e.g., SYS_#) have been created, please, drop them by means of the following SQL statement `DROP INDEX IndexName;`

Execution plan computation for a query

To obtain the execution plan for a query through Web interface, it is necessary to execute the query and then to click the "Piano di esecuzione" button (as shown in Fig.1) to display the execution plan of the query.

Fig.1

Useful SQL statements

- To view the table schema with all attributes:
  ```
  DESCRIBE TableName;
  ```

- To create an index:
  ```
  CREATE INDEX IndexName ON TableName(ColumnName);
  ```

- To compute statistics related to indexes:
  ```
  ANALYZE INDEX IndexName COMPUTE STATISTICS;
  ```
- To remove an index:
  ```
  DROP INDEX IndexName;
  ```

- To view the indexes related to a table:
  ```
  SELECT INDEX_NAME FROM USER_INDEXES
  WHERE table_name='Table Name needs to be written in capital letters';
  ```

- Display statistics related to indexes:
  ```
  SELECT USER_INDEXES.INDEX_NAME as INDEX_NAME, INDEX_TYPE,
  USER_INDEXES.TABLE_NAME, COLUMN_NAME||'('||COLUMN_POSITION||')' as
  COLUMN_NAME, BLEVEL, LEAF_BLOCKS, DISTINCT_KEYS, AVG_LEAF_BLOCKS_PER_KEY,
  AVG_DATA_BLOCKS_PER_KEY, CLUSTERING_FACTOR
  FROM user_indexes, user_ind_columns
  WHERE user_indexes.index_name=user_ind_columns.index_name and
  user_indexes.table_name=user_ind_columns.table_name;
  ```

- Display statistics related to tables:
  ```
  SELECT TABLE_NAME, NUM_ROWS, BLOCKS, EMPTY_BLOCKS, AVG_SPACE, CHAIN_CNT,
  AVG_ROW_LEN
  ```

```
FROM USER_TABLES;
```

- Display statistics related to table columns:

```
SELECT COLUMN_NAME, NUM_DISTINCT, NUM_NULLS, NUM_BUCKETS, DENSITY
FROM USER_TAB_COL_STATISTICS
WHERE TABLE_NAME = 'TableName' ORDER BY COLUMN_NAME;
```

- Display histograms:

```
SELECT *
FROM USER_HISTOGRAMS;
```

Queries

The following queries should be analyzed during the practice performing the following steps:

1. algebraic expression represented like a tree structure of the query
2. execution plan selected by Oracle optimizer when no physical secondary structures are defined
3. **Only** for queries from #4 to #7, Select one or more secondary physical structures to increase query performance.

Resume of table structures

EMP (EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO)
DEPT (DEPTNO, DNAME, LOC)
SALGRADE (GRADE, LOSAL, HISAL)

Query #1

```
SELECT *
FROM emp, dept
WHERE emp.deptno = dept.deptno AND emp.job = 'ENGINEER';
```

Change the optimizer goal from ALL ROWS (best throughput) to FIRST_ROWS (best response time) by means of the following hint. Set different values for n.

```
SELECT /*+ FIRST_ROWS(n) */ *
FROM emp, dept
WHERE emp.deptno = dept.deptno AND emp.job = 'ENGINEER';
```

Query #2

Disable the hash join method by means of the following hint: (/*+ NO_USE_HASH(e d) */)

```
SELECT /*+ NO_USE_HASH(e d) */ d.deptno, AVG(e.sal)
FROM emp e, dept d
WHERE d.deptno = e.deptno
GROUP BY d.deptno;
```

Query #3

Disable the hash join method by means of the following hint: (/*+ NO_USE_HASH(e d) */)

```
SELECT /*+ NO_USE_HASH(e d) */ ename, job, sal, dname
FROM emp e, dept d
WHERE e.deptno = d.deptno
AND NOT EXISTS
   (SELECT * FROM salgrade WHERE e.sal = hisal);
```

Queries #4

Select one or more secondary structures to optimize the following query:

```
select avg(e.sal) from
emp e where e.deptno <
10 and
e.sal > 100 and e.sal < 200;
```

Compare query performance using distinct secondary structures on different attributes with the one achieved by a unique secondary structure on multiple attributes.

Query #5

Select one or more secondary structures to optimize the following query:

```
select dname      from
dept
where deptno in (select deptno
from emp
                 where job = 'PHILOSOPHER');
```

Query #6

Select one or more secondary structures to optimize the following query (remove already existing indexes to compare query performance with and without indexes):

```
select e1.ename, e1.empno, e1.sal, e2.ename, e2.empno, e2.sal          from
emp e1, emp e2
where e1.ename <> e2.ename and e1.sal < e2.sal
and e1.job = 'PHILOSOPHER' and e2.job = 'ENGINEER';
```

Query #7

Select one or more secondary structures to optimize the following query:

```
select *
from emp e, dept d
where e.deptno = d.deptno
and e.sal not in (select hisal from salgrade  where
                   hisal > 500 and hisal < 1900);
```