|  |
| --- |
|  |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REVISION HISTORY | | | | | |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.0 | Initial status | [Kiryl Bucha](mailto:Kiryl_Bucha@epam.com) | 12-JAN-2012 |  |  |
| 2.0 | Updated in accordance with renewed content | [Elias Nema](mailto:Elias_Nema@epam.com) | 20-JAN-2014 |  |  |

Contents

[1. Table access 3](#_Toc443488391)

[1.1. Task 1: Full Scan, High-Water Mark and Consistent Gets 3](#_Toc443488392)

[1.1.1. Step 1 3](#_Toc443488393)

[1.1.2. Step 2 3](#_Toc443488394)

[1.1.3. Step 3 3](#_Toc443488395)

[1.1.4. Step 4 4](#_Toc443488396)

[1.1.5. Step 5 4](#_Toc443488397)

[1.1.6. Step 6 4](#_Toc443488398)

[1.1.7. Step 7 4](#_Toc443488399)

[1.1.8. Step 8 4](#_Toc443488400)

[1.1.9. Step 9 4](#_Toc443488401)

[1.1.10. Task Result 4](#_Toc443488402)

[2. Index Scan types 4](#_Toc443488403)

[2.1. Task 2: Index Clustering Factor 4](#_Toc443488404)

[2.1.1. Step 1 4](#_Toc443488405)

[2.1.2. Step 2 4](#_Toc443488406)

[2.1.3. Step 3 5](#_Toc443488407)

[2.1.4. Step 4 5](#_Toc443488408)

[2.1.5. Step 5 5](#_Toc443488409)

[2.1.6. Task Result 5](#_Toc443488410)

[2.2. Task 3: Index Unique Scan 5](#_Toc443488411)

[2.2.1. Step 1 5](#_Toc443488412)

[2.2.2. Step 2 5](#_Toc443488413)

[2.2.3. Task Result 5](#_Toc443488414)

[2.3. Task 4: Index Range Scan 5](#_Toc443488415)

[2.3.1. Step 1 5](#_Toc443488416)

[2.3.2. Task Result 5](#_Toc443488417)

[2.4. Task 5: Index Skip Scan 5](#_Toc443488418)

[2.4.1. Step 1 5](#_Toc443488419)

[2.4.2. Step 2: 6](#_Toc443488420)

[2.4.3. Step 3 6](#_Toc443488421)

[2.4.4. Task Result 6](#_Toc443488422)

# Table access

## Task 1: Full Scan, High-Water Mark and Consistent Gets

### Step 1

CREATE TABLE t2 AS

SELECT TRUNC( rownum / 100 ) id, rpad( rownum,100 ) t\_pad

FROM dual

CONNECT BY rownum < 100000;

### Step 2

CREATE INDEX t2\_idx1 ON t2 ( id );

### Step 3

Number of blocks used by segment:

SELECT blocks FROM user\_segments WHERE segment\_name = 'T2';

Number of blocks used by real data:

SELECT COUNT(DISTINCT (dbms\_rowid.rowid\_block\_number(rowid))) block\_ct

FROM t2 ;

Use autotrace feature of SQL Developer (F6):

SELECT COUNT( \* ) FROM t2;

|  |  |
| --- | --- |
| consistent gets | 1539 |
| consistent gets direct | 1536 |
| consistent gets from cache | 3 |
| consistent gets pin | 3 |
| consistent gets pin (fastpath) | 3 |
| CPU used by this session | 1 |
| CPU used when call started | 2 |
| logical read bytes from cache | 24576 |
| no work - consistent read gets | 1536 |
| Number of read IOs issued | 53 |
| opened cursors cumulative | 2 |
| parse count (total) | 2 |
| physical read bytes | 12582912 |
| physical read IO requests | 53 |
| physical read total bytes | 12582912 |
| physical read total IO requests | 53 |
| physical read total multi block requests | 45 |
| physical reads | 1536 |
| physical reads direct | 1536 |
| Requests to/from client | 25 |
| session logical reads | 1539 |
| session pga memory | -393216 |
| sorts (memory) | 2 |
| sorts (rows) | 2356 |
| SQL\*Net roundtrips to/from client | 25 |
| table scan blocks gotten | 1536 |
| table scan disk non-IMC rows gotten | 99999 |
| table scan rows gotten | 99999 |
| table scans (direct read) | 1 |
| table scans (short tables) | 1 |
| user calls | 27 |
| workarea memory allocated | 261 |

### Step 4

Delete all rows from the table:

DELETE FROM t2;

### Step 5

Repeat Step 3 and collect results.

### Step 6

Insert 1 row:

INSERT INTO t2

( ID, T\_PAD )

VALUES

( 1,'1' );

COMMIT;

### Step 7

Repeat Step 3 and collect results.

### Step 8

Truncate table:

TRUNCATE TABLE t2;

### Step 9

Repeat Step 3 and collect results.

### Task Result

Expecting:

* Tasks’ results and your analysis of these results in the description column.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Count of Blocks | Count of Used Blocks | Count of Rows | Consistent gets | Description |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# Index Scan types

## Task 2: Index Clustering Factor

### Step 1

Create table t2 as in task 1 steps 1-2

### Step 2

Create table t1 as listed below:

CREATE TABLE t1 AS

SELECT MOD( rownum, 100 ) id, rpad( rownum,100 ) t\_pad

FROM dual

CONNECT BY rownum < 100000;

### Step 3

CREATE INDEX t1\_idx1 ON t1

( id );

### Step 4

Don’t forget to gather statistics for both tables:

EXEC dbms\_stats.gather\_table\_stats( USER,'t1',method\_opt=>'FOR ALL COLUMNS SIZE 1',CASCADE=>TRUE );

EXEC dbms\_stats.gather\_table\_stats( USER,'t2',method\_opt=>'FOR ALL COLUMNS SIZE 1',CASCADE=>TRUE );

### Step 5

View index clustering factor:

SELECT t.table\_name||'.'||i.index\_name idx\_name,

i.clustering\_factor,

t.blocks,

t.num\_rows

FROM user\_indexes i, user\_tables t

WHERE i.table\_name = t.table\_name

AND t.table\_name IN( 'T1','T2' );

### Task Result

Expecting:

* Screenshot of step 5.
* Description of the index clustering factor.
* Explanation: why do we have different factors for t1\_idx1 and t2\_idx.
* Which Index has better performance while executing **select** clause filtered by **IN ( list of values ).**

## Task 3: Index Unique Scan

### Step 1

CREATE UNIQUE INDEX udx\_t1 ON t1( t\_pad );

### Step 2

SELECT t1.\* FROM t1 where t1.t\_pad = '1';

### Task Result

Expecting:

* Screenshot of step 2.
* Process description: How does oracle read the block on step#2?

## Task 4: Index Range Scan

### Step 1

SELECT t2.\* FROM t2 where t2.id = '1';

### Task Result

Expecting:

* Screenshot of the step 1.
* Description of process: How does oracle read the block on step#1?

## Task 5: Index Skip Scan

### Step 1

CREATE TABLE employees AS

SELECT \*

FROM scott.emp;

### Step 2:

CREATE INDEX idx\_emp01 ON employees

( empno, ename, job );

### Step 3

Collect trace and statistic for the queries below:

SELECT /\*+INDEX\_SS(emp idx\_emp01)\*/ emp.\* FROM employees emp where ename = 'SCOTT';

SELECT /\*+FULL\*/ emp.\* FROM employees emp WHERE ename = 'SCOTT';

### Task Result

Expecting:

* Step 3 screens.
* Process description: How does oracle uses the index created on the step#2?;
* Tasks’ results and your analysis of these results in the description column.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Count of Blocks | Count of Used Blocks | Count of Rows | Consistent gets | Description |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |