1Z0-117Q&As

Oracle Database 11g Release 2: SQL Tuning Exam

Pass home 1Z0-117 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.geekcert.com/1z0-117.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by home Official Exam Center

QUESTION 1

Exhibit

```
SQL SELECT id "id", parent_id, position "pos"
lpad(' ', "level )||operation||decode (id, 0, cost||POSITION" operation),
Operations "option" object_name "onject", object_node "table_queue",
Other_tag parallel oper type, distribution "row dist", other "slave SQL"
FROM plan_table
Connect by prior id=parent_id START WITH id=0
ORDER By id;
```

Id	par	pos	operations	option	object
0		4	SELECT STATEMENT cost=4		
1	0	1	HASH	GROUP BY	
2	1	1	NESTED LOOPS		
3	2	1	TABLE ACCESS	FULL	DEPARTMENTS
4	2	2	INDEX	RANGE SCAN	EMP_DEPARTMENT_IX

Examine the following SQL statement:

```
SQL> EXPLAIN PLAN FOR
SELECT department_name, count (')
FROM hr. employees e, hr.departments d
WHERE e.department_id=d.department_id
Group by.ddepartment_name;
```

Examine the exhibit to view the execution plan. Which statement is true about the execution plan?

A. The EXPLAIN PLAN generates the execution plan and stores it in c$SQL_PLAN after executing the query. Subsequent executions will use the same plan.

B. The EXPLAIN PLAN generates the execution plan and stores it in PLAN_TABLE without executing the query. Subsequent executions will always use the same plan.

C. The row with the ID 3 is the first step executed in the execution plan.

D. The row with the ID 0 is the first step executed in the execution plan.

E. The rows with the ID 3 and 4 are executed simultaneously.

Correct Answer: E

Note the other_tag parallel in the execution plan.

Note:

Within the Oracle plan_table, we see that Oracle keeps the parallelism in a column called other_tag. The other_tag column will tell you the type of parallel

operation that is being performed within your query.

For parallel queries, it is important to display the contents of the other_tag in the execution.

QUESTION 2

An application user complains about statement execution taking longer than usual. You find that the query uses a bind variable in the WHERE clause as follows:

```
SELECT prod_category, AVG(amount_sold)
FROM sales s, products p
WHERE p.prod_id = s.prod_id AND prod_category != :pcat
GROUP BY prod_category
```

You want to view the execution plan of the query that takes into account the value in the bind variable PCAT. Which two methods can you use to view the required execution plan?

A. Use the DBMS_XPLAN.DISPLAY function to view the execution plan.

B. Identify the SQL_ID for the statementsand use DBMS_XPLAN.DISPLAY_CURSOR for that SQL_ID to view the execution plan.

C. Identify the SQL_ID for the statement and fetch the execution plan PLAN_TABLE.

D. View the execution plan for the statement from V$SQL_PLAN.

E. Execute the statement with different bind values and set AUTOTRACE enabled for session.

Correct Answer: BD

D: V$SQL_PLAN contains the execution plan information for each child cursor loaded in the library cache.

B: The DBMS_XPLAN package supplies five table functions:

DISPLAY_SQL_PLAN_BASELINE - to display one or more execution plans for the SQL statement identified by SQL handle

DISPLAY - to format and display the contents of a plan table.

DISPLAY_AWR - to format and display the contents of the execution plan of a stored SQL statement in the AWR.

DISPLAY_CURSOR - to format and display the contents of the execution plan of any loaded cursor.

DISPLAY_SQLSET - to format and display the contents of the execution plan of statements stored in a SQL tuning set.

QUESTION 3

Which three are tasks performed in the hard parse stage of a SQL statement executions?

A. Semantics of the SQL statement are checked.

B. The library cache is checked to find whether an existing statement has the same hash value.

C. The syntax of the SQL statement is checked.

D. Information about location, size, and data type is defined, which is required to store fetched values in variables.

E. Locks are acquired on the required objects.

Correct Answer: BDE

Parse operations fall into the following categories, depending on the type of statement submitted and the result of the hash check: A) Hard parse

If Oracle Database cannot reuse existing code, then it must build a new executable version of the application code. This operation is known as a hard parse, or a

library cache miss. The database always perform a hard parse of DDL.

During the hard parse, the database accesses the library cache and data dictionary cache numerous times to check the data dictionary. When the database

accesses these areas, it uses a serialization device called a latch on required objects so that their definition does not change (see "Latches"). Latch contention

increases statement execution time and decreases concurrency.

B) Soft parse

A soft parse is any parse that is not a hard parse. If the submitted statement is the same as a reusable SQL statement in the shared pool, then Oracle Database

reuses the existing code. This reuse of code is also called a library cache hit.

Soft parses can vary in the amount of work they perform. For example, configuring the session cursor cache can sometimes reduce the amount of latching in the

soft parses, making them "softer."

In general, a soft parse is preferable to a hard parse because the database skips the optimization and row source generation steps, proceeding straight to

execution.

Incorrect: A, C: During the parse call, the database performs the following checks: Syntax Check Semantic Check Shared Pool Check The hard parse is within Shared Pool check. Reference: Oracle Database Concepts 11g, SQL Parsing

QUESTION 4

See the code fragment:

```
PARALLEL_DEGREE_POLICY=MANUAL
PARALLEL_MIN_PERCENT = 50
PARALLEL_MAX_SERVERS = 128
PARALLEL_MIN_SERVERS = 0
PARALLEL_DEGREE_LIMIT = 8
```

You execute the following query:

```
SELECT /*+ FULL (c, 8) */ c.cust_last_name, s.time_id, s.quantity_id,
s.quality_sold
FROM oe.customers c, sh.sales s
WHERE s.cust_id = c.customer_id;
```

You receive the following error message:

ORA-12827: insufficient parallel query slaves available

Which three parameter settings could you change to avoid this error?

A. Decrease the value of PARALLEL_MIN_PERCENT

B. Increase the value of PARALLEL_MAX_SERVERS

C. Increase the value of PARALLEL_MIN_SERVERS

D. Reduce the value of PARALLEL_MIN_TIME_THRESHOLF

E. Increase the value of PARALLEL_DEGREE_LIMIT

F. Set the PARALLEL_DEGREE_POLICY = AUTO

G. Set the PARALLEL_DEGREE_POLICY = LIMITED

Correct Answer: ABG

A: ORA-12827: insufficient parallel query slaves available Cause: PARALLEL_MIN_PERCENT parameter was specified and fewer than minimum slaves were acquired Action: either re-execute query with lower PARALLEL_MIN_PERCENT or wait until some running queries are completed, thus freeing up slaves

B: Your query doesn\\'t run because you\\'ve told Oracle not to run it unless at least 5% of the parallel execution processes are available for your query.

Set PARALLEL_MIN_PERCENT=0 or increase the number of parallel execution processes by increasing the PARALLEL_MAX_SERVERS parameter.

G: PARALLEL_DEGREE_POLICY

PARALLEL_DEGREE_POLICY specifies whether or not automatic degree of Parallelism, statement queuing, and in-memory parallel execution will be enabled.

LIMITED

Enables automatic degree of parallelism for some statements but statement queuing and in- memory Parallel Execution are disabled. Automatic degree of parallelism is only applied to those statements that access tables or indexes decorated explicitly with the PARALLEL clause. Tables and indexes that have a degree of parallelism specified will use that degree of parallelism.

Note: PARALLEL_MIN_PERCENT operates in conjunction with PARALLEL_MAX_SERVERS and PARALLEL_MIN_SERVERS. It lets you specify the minimum percentage of parallel execution processes (of the value of PARALLEL_MAX_SERVERS) required for parallel execution. Setting this parameter ensures that parallel operations will not execute sequentially unless adequate resources are available. The default value of 0 means that no minimum percentage of processes has been set.

Consider the following settings:

PARALLEL_MIN_PERCENT = 50 PARALLEL_MIN_SERVERS = 5 PARALLEL_MAX_SERVERS = 10

If 8 of the 10 parallel execution processes are busy, only 2 processes are available. If you then request a query with a degree of parallelism of 8, the minimum 50% will not be met.

You can use this parameter in conjunction with PARALLEL_ADAPTIVE_MULTI_USER. In a multi-user environment, an individual user or application can set PARALLEL_MIN_PERCENT to a minimum value until sufficient resources are available on the system and an acceptable degree of parallelism is returned.

QUESTION 5

Examine Exhibit 1 to view the query and its execution plan.

```
SQL> select
        First_name, e.department_id, d.department_id, d,department_name from employees e, departments d
        Where e.department_id = d.department_id and last_name like '%a%'

106 rows selected.

Execution plan
----------------------------
Plan hash value: 1473400139
```

Id	Operation	Name	Rows	Bytes	Cost	(%CPU)	Time
0	SELECT STATEMENT		106	2576	6	(34)	00:00:01
1	MERGE JOIN		106	2756	6	(34)	00:00:01
2	TABLE ACCESS BY INDEX ROWID	DEPARTMENTS	27	432	2	(0)	00:00:01
3	INDEX FULL SCAN	DEPT_IDPK	27		1	(0)	00:00:01
*4	SORT JOIN		107	1070	4	(50)	00:00:01
5	VIEW	indes$_join$_001	107	7070	3	(34)	00:00:01
*6	HASH JOIN						
7	INDEX FAST FULL SCAN	EMP_DEPARTMENT_IX	107	1070	1	(0)	00:00:01
*8	INDEX FAST FULL SCAN	EMP_NAME_IX	107	1070	1	(0)	00:00:01

```
Predicate information (Identified by Operation id):
---------------------------------------------------
4 – access ("E", "DEPARTMENT_ID" = "D", DEPARTMENT_ID)
  – filter ("E", "DEPARTMENT_ID" = "D", DEPARTMENT_ID)
6 – access (ROWID=ROWID)
8 – filter ("LAST_NAME" LIKE '%A%')

Statistics
----------------------------------------------------------
1          recursive calls
0          db block gets
20         consistent gets
0          physical reads
0          redo size
4034       bytes sent via SQL*NET to client
596        bytes received via SQL*NET from client
9          SQL*NET roundtrips to/from client
1          sorts (memory)
0          sorts (disk)
106        rows proceed
```

Examine Exhibit 2 to view the structure and indexes for the EMPLOYEES and DEPARTMENTS tables. Examine Exhibit 3 to view the initialization parameters for the instance.

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR(20)
LAST_NAME	NOT NULL	VARCHAR(25)
EMAIL	NOT NULL	VARCHAR (25)
PHONE_NUMBER		VARCHAR(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR (10)
SALARY		NUMBER (8, 2)
COMISSION_PCT		NUMBER (2, 2)
MANAGER_ID		NUMBER (6)
DEPARTMENT_ID		NUMBER (4)

INDEX_NAME	INDEX_TYPE	COLUMN_NAME
EMP_NAME_IX	NORMAL	LAST_NAME
EMP_MANAGER_IX	NORMAL	MANAGER_ID
EMP_JOB_IX	NORMAL	JOB_ID
EMP_DEPARTMENT_IX	NORMAL	DEPARTMENT_ID
EMP_EMP_ID_PK	NORMAL	EMPLOYEE_ID
EMP_EMAIL_UK	NORMAL	EMAIL

Departments

Name	Null?	Type
DEPARTMENT	NOT NULL	NUMBER (4)
DEPARTMENT_NAME	NOT NULL	VARCHAR (30)
MANAGER_ID		NUMBER (6)
LOCATION_ID		NUMBER (4)

INDEX_NAME	INDEX_TYPE	COLUMN_NAME
DEPT_LOCATION_IX	NORMAL	LOCATION_ID
DEPT_ID_PK	NORMAL	DEPARTMENT_ID

NAME	TYPE	VALUE
optimizer_capture_sql_plan_baselines	boolean	FALSE
optimizer_dynamic_sampling	integer	2
optimizer_features_sampling	string	11.2.0.1
optimizer_index_catching	integer	0
optimizer_index_cost_adj	integer	100
optimizer_mode	string	ALL_ROWS
optimizer_secure_view_merging	boolean	TRUE
optimizer_use_invisible_indexes	boolean	FALSE
optimizer_use_pending_statistics	boolean	FALSE
optimizer_use_sql_plan_baselines	boolean	TRUE

Why is sort-merge join chosen as the access method?

A. Because the OPTIMIZER_MODE parameter is set to ALL_ROWS.

B. Because of an inequality condition.

C. Because the data is not sorted in the LAST_NAME column of the EMPLOYEES table

D. Because of the LIKE operator used in the query to filter out records

Correct Answer: A

Incorrect:

B: There is not an inequality condition in the statement.

C: Merge joins are beneficial if the columns are sorted.

D: All regular joins should be able to use Hash or Sort Merge, except LIKE, !=, and NOT ... joins.

Note:

*

A sort merge join is a join optimization method where two tables are sorted and then joined.

*

A "sort merge" join is performed by sorting the two data sets to be joined according to the join keys and then merging them together. The merge is very cheap, but the sort can be prohibitively expensive especially if the sort spills to disk. The cost of the sort can be lowered if one of the data sets can be accessed in sorted order via an index, although accessing a high proportion of blocks of a table via an index scan can also be very expensive in comparison to a full table scan.

*

Sort merge joins are useful when the join condition between two tables is an inequality condition (but not a nonequality) like =. Sort merge joins

perform better than nested loop joins for large data sets. You cannot use hash joins unless there is an equality

condition.

*

When the Optimizer Uses Sort Merge Joins

The optimizer can choose a sort merge join over a hash join for joining large amounts of data if any of the following conditions are true:

/ The join condition between two tables is not an equi-join.

/ Because of sorts already required by other operations, the optimizer finds it is cheaper to use a sort merge than a hash join. Reference: Oracle Database Performance Tuning Guide , Sort Merge Joins