Enter user-name: system

Enter password:

Connected to:

Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production

With the Partitioning, OLAP, Data Mining and Real Application Testing options

/\*

/\*Q1. Create individual b-tree indexes on the following columns of the table:

customers\_copy\_btree

(a) cust\_gender

(b) cust\_year\_of\_birth

(c) cust\_last\_name

(d) cust\_street\_address

How long does it take to create the indexes?\*/

create table customer\_copy\_btree as select \* from sh.customers;

create index customer\_copy\_btree\_gender\_idx on customer\_copy\_btree(cust\_gender);

Elapsed: 00:00:00.20

create index customer\_copy\_btree\_yob\_idx on customer\_copy\_btree(cust\_year\_of\_birth);

Elapsed: 00:00:00.03

create index customer\_copy\_btree\_lname\_idx on customer\_copy\_btree(cust\_last\_name);

Elapsed: 00:00:00.06

create index customer\_copy\_btree\_stra\_idx on customer\_copy\_btree(cust\_street\_address);

Elapsed: 00:00:00.04

COLUMN TIME INDEX BTREE

CREATION TIME Elapsed: 00:00:00.29

GENDER Elapsed: 00:00:00.29

YEAR OF BIRTH Elapsed: 00:00:00.23

LAST NAME Elapsed: 00:00:00.04

STREET ADDRESS Elapsed: 00:00:00.04

/\*Q2. Create bitmap indexes on the above columns. How long does it take to create bitmap

indexes? Compare it with the results of btree index creation.\*/

SQL> CREATE TABLE CUSTOMER\_BITMAP AS ( SELECT \* FROM SH.CUSTOMERS);

Table created.

Elapsed: 00:00:00.29

SQL> create bitmap index cus\_gender\_bitmap\_idx on customer\_bitmap(cust\_gender);

Index created.

Elapsed: 00:00:00.23

SQL> create bitmap index cus\_year\_of\_birth\_bitmap\_idx on

customer\_bitmap(cust\_year\_of\_birth);

Index created.

Elapsed: 00:00:00.04

SQL> create bitmap index cus\_last\_name\_bitmap\_idx on customer\_bitmap(cust\_last\_name);

Index created.

Elapsed: 00:00:00.03

SQL> create bitmap index cus\_street\_address\_bitmap\_idx on

customer\_bitmap(cust\_street\_address);

Index created.

Elapsed: 00:00:00.04

COMPARISON CHART

COLUMN TIME INDEX BTREE TIME INDEX

BITMAP

CREATION TIME Elapsed: 00:00:03.06 Elapsed: 00:00:02.27

GENDER Elapsed: 00:00:00.38 Elapsed: 00:00:00.34

YEAR OF BIRTH Elapsed: 00:00:00.64 Elapsed: 00:00:00.03

LAST NAME Elapsed: 00:00:00.32 Elapsed: 00:00:00.05

STREET ADDRESS Elapsed: 00:00:01.03 Elapsed: 00:00:00.14

/\*Q3. Do as directed:

(a) Find the size of each segment: customers\_copy\_bitmap and customers\_copy\_btree

(b) The b-tree index range for high and low cardinality address index.

(c) The bitmap index range for high and low cardinality address index.

\*/

--A

select segment\_name,

bytes/1024/1024 "Size in MB"

from user\_segments

where segment\_name like '%CUSTOMER\_COPY%';

--B

select segment\_name,

bytes/1024/1024 "Size in MB"

from user\_segments

where segment\_name like '%BTREE%';

SEGMENT\_NAME Size in MB

CUS\_GENDER\_BTREE\_IDX

CUS\_YEAR\_OF\_BIRTH\_BTREE\_IDX

CUS\_LAST\_NAME\_BTREE\_IDX

.875

2

1

CUS\_GENDER\_BTREE\_IDXX

CUS\_YEAR\_OF\_BIRTH\_BTREE\_IDXX

CUS\_LAST\_NAME\_BTREE\_IDXX

.875

2

1

--C

select segment\_name,

bytes/1024/1024 "Size in MB"

from user\_segments

where segment\_name like '%BITMAP%';

SEGMENT\_NAME Size in MB

CUSTOMER\_BITMAPS 12

CUS\_GENDER\_BITMAP\_IDX .0625

CUS\_YEAR\_OF\_BIRTH\_BITMAP\_IDX .1875

CUS\_LAST\_NAME\_BITMAP\_IDX .125

CUS\_STREET\_ADDRESS\_BITMAP\_IDX 3

Elapsed: 00:00:00.06

/\* Q4. Do as directed:

a. Create function based index on Employee table of HR schema. Function should be on salary

attribute based on commission percentage.

b. Find out list of employees having commission percentage less than 50000.

c. Create function based index on employee name for Upper and lower function.

d. Create user table with attributes (UserId, UserName, Gender)

e. Insert 10000 records in user table

f. Build regular index on Username

g. Build function based index on user name based on Upper function

h. Compare the response time and comment.

\*/

-- A

CREATE INDEX idx\_emp\_salary\_comm\_pct

ON hr.employees ((salary \* commission\_pct));

-- B

SELECT \*

FROM hr.employees

WHERE commission\_pct < 0.5;

-- C

CREATE INDEX idx\_emp\_name\_upper\_lower

ON hr.employees (UPPER(LOWER(name)));

SELECT \*

FROM hr.employees

WHERE UPPER(LOWER(name)) = UPPER(LOWER('John Smith'));

--D

CREATE TABLE user (

UserId INT PRIMARY KEY,

UserName VARCHAR(50) NOT NULL,

Gender CHAR(1) NOT NULL

);

-- E

DECLARE @i INT = 1;

WHILE (@i <= 10000)

BEGIN

INSERT INTO user (UserId, UserName, Gender)

VALUES (@i, 'User' + CAST(@i AS VARCHAR), CASE WHEN @i%2 = 0 THEN 'F' ELSE

'M' END);

SET @i = @i + 1;

END;

-- F

CREATE INDEX idx\_user\_username ON user (UserName);

SELECT \* FROM user WHERE UserName = 'JohnDoe';

-- G

CREATE INDEX idx\_user\_username\_upper ON user (UPPER(UserName));

SELECT \* FROM user WHERE UPPER(UserName) = 'JOHNDOE';

-- H

/\*Regular Indexes:

More efficient for exact matches on indexed columns

Better for range queries that can be satisfied by the index

Generally more efficient for most queries

Function-Based Indexes:

More useful for queries that involve functions applied to indexed columns, such as UPPER or

LOWER

Better for case-insensitive matching

Useful for certain types of queries that regular indexes cannot optimize as well

May have higher maintenance overhead due to the additional processing required to maintain the

index

\*/

--INDEX ORGANIZED TABLE

-- Q5. Do as directed :

/\*1. Create an IOT look\_ups with the attributes (lookup\_code, lookup\_value,

lookup\_description) in tablespace ts\_lookup.

Constraint: lookup\_code should be primary key

PctThreshold is 20 and and lookup\_description should be in overflow area.

Overflow should be in ts\_overflow tablespace.

\*/

--1

CREATE TABLESPACE LOOKUPS DATAFILE 'C:\Users\admin\Desktop\DWM' SIZE 10M;

SQL> CREATE TABLESPACE LOOKUPTEST DATAFILE

'C:\Users\admin\Desktop\DWM\lookuptest.dbf' SIZE 10M;

Tablespace created.

Elapsed: 00:00:01.51

SQL>

CREATE TABLE IOT\_LOOKUPS(

lookup\_code NUMBER(10),

lookup\_value NUMBER(10),

lookup\_description VARCHAR2(40),

CONSTRAINT LOOKUP\_PK PRIMARY KEY(LOOKUP\_CODE))

ORGANIZATION INDEX

PCTTHRESHOLD 20

INCLUDING LOOKUP\_DESCRIPTION

OVERFLOW TABLESPACE LOOKUPTEST;

Table created.

BEGIN

FOR lookup\_code IN 1 .. 100000 LOOP

IF MOD(lookup\_code, 2) = 0 THEN

INSERT INTO IOT\_LOOKUPS

VALUES (lookup\_code, lookup\_code+1, 'String');

ELSE

INSERT INTO IOT\_LOOKUPS

VALUES (lookup\_code, lookup\_code+2, 'String\_2');

END IF;

COMMIT;

END LOOP;

END;

/

PL/SQL procedure successfully completed.

Elapsed: 00:00:06.98

--Q6. Do as directed

/\*

1. Create a Index Organized Table(IOT) emp\_iot based on hr.employees

2. Create a Index Organized Table(IOT) emp101\_emp based on hr.employees. Place the

column hiredate in overflow area.

3. Compare the timings of executing select all from employees,emp\_iot, and emp101\_iot.

Comment on your observations.\*/

-- 1

CREATE TABLE EMP\_IOT

(EMP\_NO NUMBER,

EMP\_NAME VARCHAR2(20),

EMP\_DEPT NUMBER,

EMP\_ADDRESS VARCHAR2(500),

EMP\_HIST VARCHAR2(1000),

CONSTRAINT EMP\_PK PRIMARY KEY(EMP\_NO))

ORGANIZATION INDEX

INCLUDING EMP\_NAME

TABLESPACE LOOKUP\_DESCRIPTION

OVERFLOW TABLESPACE LOOKUPTEST;

Tablespace Created.

BEGIN

FOR EMP\_NO IN 1 .. 100000 LOOP

IF MOD(EMP\_NO, 2) = 0 THEN

INSERT INTO EMP\_IOT

VALUES (EMP\_NO,'SAM', 101, 'MYADDRESS', 'OLDTEXT');

ELSE

INSERT INTO EMP\_IOT

VALUES (EMP\_NO,'JAM', 501, 'MYADDRESS', 'OLDTEXT');

END IF;

COMMIT;

END LOOP;

END;

/

PL/SQL procedure successfully completed.

Elapsed: 00:00:13.96

--2

CREATE TABLE EMP\_IOT101

(EMP\_NO NUMBER,

EMP\_NAME VARCHAR2(20),

EMP\_DEPT NUMBER,

EMP\_ADDRESS VARCHAR2(500),

EMP\_HIST VARCHAR2(1000),

HIREDATE DATE,

CONSTRAINT EMP\_PUK PRIMARY KEY(EMP\_NO))

ORGANIZATION INDEX

INCLUDING HIREDATE

TABLESPACE LOOKUPTEST

OVERFLOW TABLESPACE LOOKUPTEST;

Table created.

BEGIN

FOR EMP\_NO IN 1 .. 100000 LOOP

IF MOD(EMP\_NO, 2) = 0 THEN

INSERT INTO EMP\_IOT101

VALUES (EMP\_NO,'SAM', 101, 'MYADDRESS', 'OLDTEXT', '1-4-2005');

ELSE

INSERT INTO EMP\_IOT101

VALUES (EMP\_NO,'JAM', 501, 'MYADDRESS', 'OLDTEXT','1-4-2005');

END IF;

COMMIT;

END LOOP;

END;

/

PL/SQL procedure successfully completed.

Elapsed: 00:00:20.16

/\* Q.7 Do as directed :

a. Create a cluster PERSONNEL containing copy\_emp and copy\_dept tables. Cluster key is

deptno.

b. create a index on cluster PERSONNEL.

c. Populate copy\_emp and copy\_dept with data from emp and dept tables of scott

respectively.

d. Create a dumkmy table &quot;dummy&quot; which references empno of copy\_emp.

e. Drop cluster PERSONNEL.

f. Create a hash cluster named hash\_emp containing table copy101\_emp. Create 10 hashkeys

and use the hash function (empno mod 100).\*/

-- A

CREATE CLUSTER personnel (deptno NUMBER);

CREATE TABLE copy\_emp (

empno NUMBER,

ename VARCHAR2(50),

job VARCHAR2(50),

mgr NUMBER,

hiredate DATE,

sal NUMBER,

comm NUMBER,

deptno NUMBER

)

CLUSTER personnel (deptno);

CREATE TABLE copy\_dept (

deptno NUMBER,

dname VARCHAR2(50),

loc VARCHAR2(50)

)

CLUSTER personnel (deptno);

--B

CREATE INDEX personnel\_deptno\_idx

ON CLUSTER personnel (deptno);

-- C

INSERT INTO copy\_dept (deptno, dname, loc)

SELECT deptno, dname, loc FROM scott.dept;

INSERT INTO copy\_emp (empno, ename, job, mgr, hiredate, sal, comm, deptno)

SELECT empno, ename, job, mgr, hiredate, sal, comm, deptno FROM scott.emp;

--D

CREATE TABLE dummy (

id NUMBER,

empno NUMBER,

CONSTRAINT dummy\_empno\_fk FOREIGN KEY (empno)

REFERENCES copy\_emp(empno)

);

-- E

DROP CLUSTER personnel;

-- F

CREATE CLUSTER hash\_emp (

empno NUMBER,

ename VARCHAR2(30),

job VARCHAR2(20),

mgr NUMBER,

hiredate DATE,

sal NUMBER,

comm NUMBER,

deptno NUMBER

)

HASHKEYS 10

HASH IS empno MOD 100;

CREATE TABLE copy101\_emp (

empno NUMBER,

ename VARCHAR2(30),

job VARCHAR2(20),

mgr NUMBER,

hiredate DATE,

sal NUMBER,

comm NUMBER,

deptno NUMBER

)

CLUSTER hash\_emp(empno);