Name: Shaikh Inamul Hasan

Roll No: 100

Lab 10: ADMS Assignment on Range, List and ORDBMS

Implementation of Data partitioning through Range and List partitioning.

Range:

1. Create a table Customer with following schema & partition the cid column by using range partitioning:

cid number not null

fname varchar2(20)

mname varchar2(20)

lname varchar2(20)

address varchar2(20)

a) Make 3 partitions which will contain the values from 1 to 99, 100 to 199, and 200 onwards.

Code:

```
CREATE TABLE cust1 (
  cid NUMBER NOT NULL,
  fname VARCHAR2(20),
  mname VARCHAR2(20),
  lname VARCHAR2(20),
  address VARCHAR2(20)
)
PARTITION BY RANGE(cid) (
  PARTITION c100 VALUES LESS THAN (100),
  PARTITION c200 VALUES LESS THAN (200),
```

```
  PARTITION c300 VALUES LESS THAN (300),
  PARTITION c400 VALUES LESS THAN (400),
  PARTITION cother VALUES LESS THAN (MAXVALUE)
);
```

Output:

```
Table created.
```

b) Insert the appropriate records into the table (Also insert some ids with value more than 400)

Code:

```
-- Insert records with IDs less than 100
INSERT INTO cust1 (cid, fname, mname, lname, address) VALUES (1, 'Inam', 'A', 'S', 'Mira Road');
INSERT INTO cust1 (cid, fname, mname, lname, address) VALUES (50, 'Inamul', 'B', 'Shaikh', 'Bhayander');
INSERT INTO cust1 (cid, fname, mname, lname, address) VALUES (90, 'Hasan', 'C', 'SH', 'Kashimira');
-- Insert records with IDs between 100 and 199
INSERT INTO cust1 (cid, fname, mname, lname, address) VALUES (120, 'Shaikh', 'D', 'Inam', 'Kandivali');
INSERT INTO cust1 (cid, fname, mname, lname, address) VALUES (150, 'I', 'E', 'S', 'Malad');
INSERT INTO cust1 (cid, fname, mname, lname, address) VALUES (180, 'Inamul Hasan', 'F', 'Shaikh', 'Andheri');
-- Insert records with IDs greater than 200
INSERT INTO cust1 (cid, fname, mname, lname, address) VALUES (250, 'Abdul', 'G', 'Shaikh', 'Mahim');
```

```
INSERT INTO cust1 (cid, fname, mname, lname, address) VALUES (300, 'Mujeeb', 'H', 'Shaikh', 'Bandra');

-- Insert records with IDs greater than 400

INSERT INTO cust1 (cid, fname, mname, lname, address) VALUES (450, 'Rahman', 'I', 'S', 'Borivali');

INSERT INTO cust1 (cid, fname, mname, lname, address) VALUES (500, 'Kadar', 'J', 'Khan', 'Dahisar');
```

Output:

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

c) Retrieve the values from the table

Code:

```
SELECT * FROM cust1;
```

Output:

CID	FNAME	MNAME	LNAME	ADDRESS
1	Inam	A	S	Mira Road
50	Inamul	B	Shaikh	Bhayander
90	Hasan	C	SH	Kashimira
120	Shaikh	D	Inam	Kandivali
150	I	E	S	Malad
180	Inamul Hasan	F	Shaikh	Andheri
250	Abdul	G	Shaikh	Mahim
300	Mujeeb	H	Shaikh	Bandra
450	Rahman	I	S	Borivali
500	Kadar	J	Khan	Dahisar

d) Retrieve the values from individual partitions

Code:

```
-- Retrieve values from the c1_to_99 partition (IDs less than 100)

SELECT * FROM cust1 WHERE cid < 100;

-- Retrieve values from the c100_to_199 partition (IDs between 100 and 199)

SELECT * FROM cust1 WHERE cid >= 100 AND cid < 200;

-- Retrieve values from the c200_and_above partition (IDs greater than or equal to 200)

SELECT * FROM cust1 WHERE cid >= 200;
```

Output:

CID	FNAME	MNAME	LNAME	ADDRESS
1	Inam	A	S	Mira Road
50	Inamul	B	Shaikh	Bhayander
90	Hasan	C	SH	Kashimira

CID	FNAME	MNAME	LNAME	ADDRESS
120	Shaikh	D	Inam	Kandivali
150	I	E	S	Malad
180	Inamul Hasan	F	Shaikh	Andheri

CID	FNAME	MNAME	LNAME	ADDRESS
250	Abdul	G	Shaikh	Mahim
300	Mujeeb	H	Shaikh	Bandra
450	Rahman	I	S	Borivali
500	Kadar	J	Khan	Dahisar

e) Retrieve the partition details from the system table.

Code:

```
SELECT table_name, partition_name, high_value

FROM user_tab_partitions

WHERE table_name = 'CUST1';
```

Output:

TABLE_NAME	PARTITION_NAME	HIGH_VALUE
CUST1	C100	100
CUST1	C200	200
CUST1	C300	300
CUST1	C400	400
CUST1	COTHER	MAXVALUE

2. Create a table with following schema

Table name: Purchase

transid number not null

cust_id number

inv_date date

cust_name varchar2(30)

- Partition the table according to inv_date such that it has 4 partitions having:

Data of 2008 & previous years,

Data of 2009

Data of 2010

Data of 2011 & onwards.

Code:

```
CREATE TABLE Purchase (
  transid NUMBER NOT NULL,
  cust_id NUMBER,
  inv_date DATE,
  cust_name VARCHAR2(30)
```

```
)

PARTITION BY RANGE(inv_date) (

  PARTITION sales2008 VALUES LESS THAN (TO_DATE('31/12/2008',
'DD/MM/YYYY')),

  PARTITION sales2009 VALUES LESS THAN (TO_DATE('31/12/2009',
'DD/MM/YYYY')),

  PARTITION sales2010 VALUES LESS THAN (TO_DATE('31/12/2010',
'DD/MM/YYYY')),

  PARTITION sales2011 VALUES LESS THAN (MAXVALUE)

);
```

Output:

```
Table created.
```

a) Insert the appropriate records into the table.

Code:

```
-- Insert records for sales in 2008

INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (1, 101,
TO_DATE('15/03/2008', 'DD/MM/YYYY'), 'I');

INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (2, 102,
TO_DATE('20/06/2008', 'DD/MM/YYYY'), 'Inam');

INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (3, 103,
TO_DATE('10/12/2008', 'DD/MM/YYYY'), 'Shaikh');

-- Insert records for sales in 2009

INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (4, 104,
TO_DATE('05/02/2009', 'DD/MM/YYYY'), 'Inamul');

INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (5, 105,
TO_DATE('18/09/2009', 'DD/MM/YYYY'), 'Hasan');

INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (6, 106,
TO_DATE('22/11/2009', 'DD/MM/YYYY'), 'Inamul Hasan');
```

-- Insert records for sales in 2010

```
INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (7, 107, TO_DATE('03/04/2010', 'DD/MM/YYYY'), 'Shaikh Inamul Hasan');

INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (8, 108, TO_DATE('14/08/2010', 'DD/MM/YYYY'), 'Mujeeb');

INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (9, 109, TO_DATE('27/12/2010', 'DD/MM/YYYY'), 'Rahman');
```

-- Insert records for sales in 2011

```
INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (10, 110, TO_DATE('08/05/2011', 'DD/MM/YYYY'), 'Abdul');

INSERT INTO Purchase (transid, cust_id, inv_date, cust_name) VALUES (11, 111, TO_DATE('19/09/2011', 'DD/MM/YYYY'), 'Kadar');
```

Output:

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

b) Retrieve the values from the table.

Code:

```
SELECT * FROM Purchase;
```

Output:

TRANSID	CUST_ID	INV_DATE	CUST_NAME
1	101	15-MAR-08	I
2	102	20-JUN-08	Inam
3	103	10-DEC-08	Shaikh
4	104	05-FEB-09	Inamul
5	105	18-SEP-09	Hasan
6	106	22-NOV-09	Inamul Hasan
7	107	03-APR-10	Shaikh Inamul Hasan
8	108	14-AUG-10	Mujeeb
9	109	27-DEC-10	Rahman
10	110	08-MAY-11	Abdul
11	111	19-SEP-11	Kadar

c) Split the last partition so that we have a separate partition for 2011; check the entries of system table.

Code:

```
-- Split the last partition for 2011

ALTER TABLE Purchase

SPLIT PARTITION sales2011

AT (TO_DATE('01/01/2011', 'DD/MM/YYYY'))

INTO (PARTITION sales2011, PARTITION sales2012);
```

Output:

```
Table altered.
```

d) Retrieve the values from individual partitions.

Code:

```
-- Retrieve values from the sales2008 partition

SELECT * FROM Purchase PARTITION (sales2008);

-- Retrieve values from the sales2009 partition

SELECT * FROM Purchase PARTITION (sales2009);

-- Retrieve values from the sales2010 partition

SELECT * FROM Purchase PARTITION (sales2010);

-- Retrieve values from the sales2011 partition

SELECT * FROM Purchase PARTITION (sales2011);

-- Retrieve values from the sales2012 partition

SELECT * FROM Purchase PARTITION (sales2012);
```

Output:

TRANSID	CUST_ID	INV_DATE	CUST_NAME
1	101	15-MAR-08	I
2	102	20-JUN-08	Inam
3	103	10-DEC-08	Shaikh

TRANSID	CUST_ID	INV_DATE	CUST_NAME
4	104	05-FEB-09	Inamul
5	105	18-SEP-09	Hasan
6	106	22-NOV-09	Inamul Hasan

TRANSID	CUST_ID	INV_DATE	CUST_NAME
7	107	03-APR-10	Shaikh Inamul Hasan
8	108	14-AUG-10	Mujeeb
9	109	27-DEC-10	Rahman

TRANSID	CUST_ID	INV_DATE	CUST_NAME
10	110	08-MAY-11	Abdul
11	111	19-SEP-11	Kadar

e) Find the number of transactions done in the year 2009.

Code:

```
-- Find the number of transactions done in the year 2009

SELECT COUNT(*) AS num_transactions

FROM Purchase

WHERE EXTRACT(YEAR FROM inv_date) = 2009;
```

Output:

NUM_TRANSACTIONS
3

List:

1. Create a table Bookshelf with following schema & partition the Category column by using list partitioning:

Table name: Bookshelf

Title varchar2(60) not null

Publisher varchar2(40) not null

Category varchar2(30)

Rating number not null;

a) Divide the data into 4 partitions using list partitioning on column Category with values 'TECHNOLOGY','QUANTITATIVE', 'LOGICAL', 'MYTHOLOGY'.

Code:

```
CREATE TABLE Bookshelf (
   Title VARCHAR2(60) NOT NULL,
   Publisher VARCHAR2(40) NOT NULL,
   Category VARCHAR2(30),
   Rating NUMBER NOT NULL
)
PARTITION BY LIST (Category) (
   PARTITION cat_technology VALUES ('TECHNOLOGY'),
   PARTITION cat_quantitative VALUES ('QUANTITATIVE'),
   PARTITION cat_logical VALUES ('LOGICAL'),
   PARTITION cat_mythology VALUES ('MYTHOLOGY'),
   PARTITION cat_other VALUES (DEFAULT)
);
```

Output:

```
Table created.
```

b) Insert the appropriate records into the table.

Code:

```
-- Insert records into the 'TECHNOLOGY' partition
INSERT INTO Bookshelf (Title, Publisher, Category, Rating)
VALUES ('Book1', 'TechPublisher1', 'TECHNOLOGY', 4.5);
INSERT INTO Bookshelf (Title, Publisher, Category, Rating)
VALUES ('Book2', 'TechPublisher2', 'TECHNOLOGY', 3.8);
-- Insert records into the 'QUANTITATIVE' partition
INSERT INTO Bookshelf (Title, Publisher, Category, Rating)
VALUES ('Book3', 'MathPublisher1', 'QUANTITATIVE', 4.2);
INSERT INTO Bookshelf (Title, Publisher, Category, Rating)
VALUES ('Book4', 'MathPublisher2', 'QUANTITATIVE', 4.8);
-- Insert records into the 'LOGICAL' partition
INSERT INTO Bookshelf (Title, Publisher, Category, Rating)
VALUES ('Book5', 'LogicPublisher1', 'LOGICAL', 4.0);
INSERT INTO Bookshelf (Title, Publisher, Category, Rating)
VALUES ('Book6', 'LogicPublisher2', 'LOGICAL', 3.5);
-- Insert records into the 'MYTHOLOGY' partition
INSERT INTO Bookshelf (Title, Publisher, Category, Rating)
VALUES ('Book7', 'MythPublisher1', 'MYTHOLOGY', 4.7);
INSERT INTO Bookshelf (Title, Publisher, Category, Rating)
VALUES ('Book8', 'MythPublisher2', 'MYTHOLOGY', 4.9);
-- Insert records into the default partition (cat_other)
```

```
INSERT INTO Bookshelf (Title, Publisher, Category, Rating)
VALUES ('Book9', 'OtherPublisher1', 'OTHER_CATEGORY', 3.2);
INSERT INTO Bookshelf (Title, Publisher, Category, Rating)
VALUES ('Book10', 'OtherPublisher2', NULL, 4.1);
```

Output:

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

c) retrieve the values from the table and from individual partitions.

Code:

```
-- Retrieve all values from the entire table
SELECT * FROM Bookshelf;
-- Retrieve values from the 'TECHNOLOGY' partition
SELECT * FROM Bookshelf PARTITION (cat_technology);
-- Retrieve values from the 'QUANTITATIVE' partition
SELECT * FROM Bookshelf PARTITION (cat_quantitative);
-- Retrieve values from the 'LOGICAL' partition
SELECT * FROM Bookshelf PARTITION (cat_logical);
-- Retrieve values from the 'MYTHOLOGY' partition
SELECT * FROM Bookshelf PARTITION (cat_mythology);
-- Retrieve values from the default partition (cat_other)
SELECT * FROM Bookshelf PARTITION (cat_other);
```

Output:

TITLE	PUBLISHER	CATEGORY	RATING
Book1	TechPublisher1	TECHNOLOGY	4.5
Book2	TechPublisher2	TECHNOLOGY	3.8
Book3	MathPublisher1	QUANTITATIVE	4.2
Book4	MathPublisher2	QUANTITATIVE	4.8
Book5	LogicPublisher1	LOGICAL	4
Book6	LogicPublisher2	LOGICAL	3.5
Book7	MythPublisher1	MYTHOLOGY	4.7
Book8	MythPublisher2	MYTHOLOGY	4.9
Book9	OtherPublisher1	OTHER_CATEGORY	3.2
Book10	OtherPublisher2	-	4.1

TITLE	PUBLISHER	CATEGORY	RATING
Book1	TechPublisher1	TECHNOLOGY	4.5
Book2	TechPublisher2	TECHNOLOGY	3.8

TITLE	PUBLISHER	CATEGORY	RATING
Book3	MathPublisher1	QUANTITATIVE	4.2
Book4	MathPublisher2	QUANTITATIVE	4.8

TITLE	PUBLISHER	CATEGORY	RATING
Book5	LogicPublisher1	LOGICAL	4
Book6	LogicPublisher2	LOGICAL	3.5

TITLE	PUBLISHER	CATEGORY	RATING
Book7	MythPublisher1	MYTHOLOGY	4.7
Book8	MythPublisher2	MYTHOLOGY	4.9

TITLE	PUBLISHER	CATEGORY	RATING
Book9	OtherPublisher1	OTHER_CATEGORY	3.2
Book10	OtherPublisher2	-	4.1

d) Retrieve the partition details from system table.

Code:

```
-- Retrieve partition details for the 'Bookshelf' table

SELECT TABLE_NAME, PARTITION_NAME, HIGH_VALUE

FROM USER_TAB_PARTITIONS

WHERE TABLE_NAME = 'BOOKSHELF';
```

Output:

TABLE_NAME	PARTITION_NAME	HIGH_VALUE
BOOKSHELF	CAT_LOGICAL	'LOGICAL'
BOOKSHELF	CAT_MYTHOLOGY	'MYTHOLOGY'
BOOKSHELF	CAT_OTHER	DEFAULT
BOOKSHELF	CAT_QUANTITATIVE	'QUANTITATIVE'
BOOKSHELF	CAT_TECHNOLOGY	'TECHNOLOGY'

Implementation of ORDBMS using ADT (Abstract Data Types)

1. Create type Address having the specified columns (address1, address2, state, city, pincode).

Create Customer table having the specified columns (Customer_id, Customer_name and

Address type).

Code:

```
-- Create type Address using ADT within a PL/SQL block
BEGIN
   EXECUTE IMMEDIATE 'CREATE TYPE Address AS OBJECT (
      address1 VARCHAR2(100),
      address2 VARCHAR2(100),
      state VARCHAR2(50),
      city VARCHAR2(50),
      pincode VARCHAR2(10)
   )';
END;
/
-- Create Customer table with Address type
CREATE TABLE Customer (
   Customer_id NUMBER PRIMARY KEY,
   Customer_name VARCHAR2(100),
   Customer_address Address
);
```

Output:

```
Statement processed.

Table created.
```

a) Insert records into customer table.

Code:

```
-- Insert records into the 'Customer' table
INSERT INTO Customer (Customer_id, Customer_name, Customer_address)
VALUES (
   100,
   'Inam',
   Address('Kashimira', 'India','Maharashtra', 'Thane', '401107')
);
INSERT INTO Customer (Customer_id, Customer_name, Customer_address)
VALUES (
   101,
   'Inamul Hasan',
   Address('Bhayander','India','Maharashtra', 'Thane', '401107')
);
```

Output:

```
1 row(s) inserted.

1 row(s) inserted.
```

b) Display the details customer.

Code:

```
-- Display details of all customers with individual attributes of the CUSTOMER_ADDRESS
SELECT
   Customer_id,
   Customer_name,
   TREAT(Customer_address AS Address).address1 AS Address1,
```

```
    TREAT(Customer_address AS Address).address2 AS Address2,
    TREAT(Customer_address AS Address).state AS State,
    TREAT(Customer_address AS Address).city AS City,
    TREAT(Customer_address AS Address).pincode AS Pincode
FROM Customer;
```

Output:

CUSTOMER_ID	CUSTOMER_NAME	ADDRESS1	ADDRESS2	STATE	CITY	PINCODE
100	Inam	Kashimira	India	Maharashtra	Thane	401107
101	Inamul Hasan	Bhayander	India	Maharashtra	Thane	401107

c) display the description/structure of the customer table.

Code:

```
DESCRIBE Customer;
```

Output:

TABLE CUSTOMER

Column	Null?	Type
CUSTOMER_ID	NOT NULL	NUMBER
CUSTOMER_NAME	-	VARCHAR2(100)
CUSTOMER_ADDRESS	-	ADDRESS

d) List the customers from Mumbai.

Code:

```
-- List customers from Mumbai
SELECT
    Customer_id,
    Customer_name,
    TREAT(Customer_address AS Address).address1 AS Address1,
```

```
    TREAT(Customer_address AS Address).address2 AS Address2,
    TREAT(Customer_address AS Address).state AS State,
    TREAT(Customer_address AS Address).city AS City,
    TREAT(Customer_address AS Address).pincode AS Pincode
FROM
    Customer
WHERE
    TREAT(Customer_address AS Address).city = 'Mumbai';
```

Output:

```
no data found
```

e) Count the number of customers' state wise.

Code:

```
-- Count the number of customers state-wise
SELECT
    TREAT(Customer_address AS Address).state AS State,
    COUNT(*) AS CustomerCount
FROM
    Customer
GROUP BY
    TREAT(Customer_address AS Address).state;
```

Output:

STATE	CUSTOMERCOUNT
Maharashtra	2