PRACTICAL-1

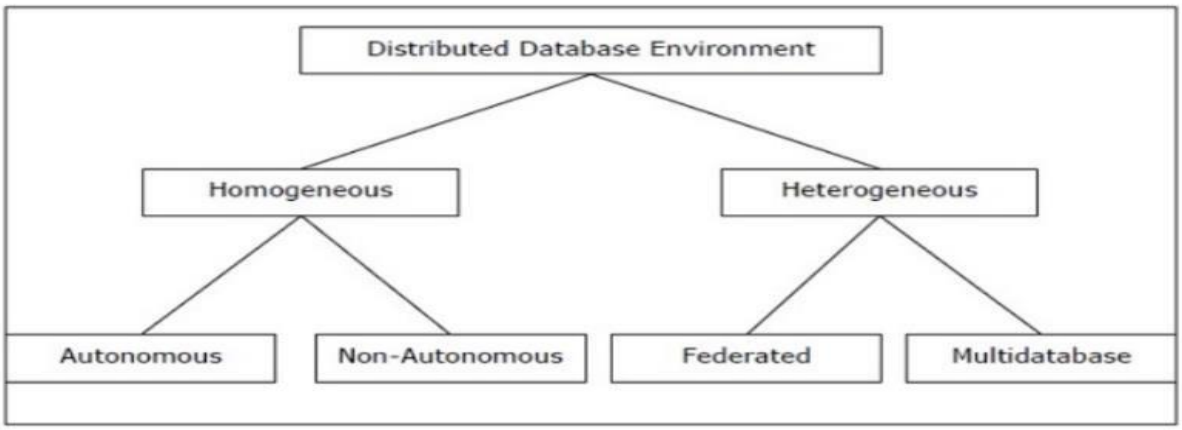
Aim: Implementation of Data Partitioning through Range and List Partitioning

Distributed Databases:

A distributed database (DDB) is an integrated collection of databases that is physically distributed across sites in a computer network. A distributed database management system (DDBMS) is the software system that manages a distributed database such that the distribution aspects are transparent to the users. To form a distributed database system (DDBS), the files must be structured, logically interrelated, and physically distributed across multiple sites. In addition, there must be a common interface to access the distributed data.

Types of Distributed Databases:

Distributed databases can be broadly classified into homogeneous and heterogeneous distributed database environments, each with further subdivisions, as shown in the following illustration.



Homogeneous Distributed Databases:

In a homogeneous distributed database, all the sites use identical DBMS and operating systems. Its properties are −

- The sites use very similar software.
- The sites use identical DBMS or DBMS from the same vendor.
- The database is accessed through a single interface as if it is a single database.

Types of Heterogeneous Distributed Databases:

- Autonomous − Each database is independent and functions on its own. They are integrated by a controlling application and use message passing to share data updates.
- Non-autonomous − Data is distributed across the homogeneous nodes and a central or master DBMS coordinates data updates across the sites.

Heterogeneous Distributed Databases:

In a heterogeneous distributed database, different sites have different operating systems, DBMS products and data models. Its properties are −

- Different sites use dissimilar schemas and software.
- The system may be composed of a variety of DBMSs like relational, network, hierarchical or object oriented.
- Query processing is complex due to dissimilar schemas.
- Transaction processing is complex due to dissimilar software.
- A site may not be aware of other sites and so there is limited co-operation in processing user requests.

Types of Heterogeneous Distributed Databases:

- Federated − The heterogeneous database systems are independent in nature and integrated together so that they function as a single database system.
- Un-federated − The database systems employ a central coordinating module through which the databases are accessed.

Data Partitioning:

Data partitioning is the technique of distributing data across multiple tables, disks, or sites in order to improve query processing performance and increase database manageability Query processing performance can be improved in one of two ways.

First, depending on how the data is partitioned, in some cases it can be determined a prior that a partition does not have to be accessed to process the query. Second, when data is partitioned across multiple disks or sites, I/O parallelism and in some cases query parallelism can be attained as different partitions can be accessed in parallel.

Range Partitioning:

Range partitioning is a type of relational database partitioning wherein the partition is based on a predefined range for a specific data field such as uniquely numbered IDs, dates or simple values like currency. A partitioning key column is assigned with a specific range, and when a data entry fits this range, it is assigned to this partition; other-wise it is placed in another partition where it fits.

List Partitioning:

List partitioning enables you to explicitly control how rows map to partitions by specifying a list of discrete values for the partitioning key in the description for each partition. The advantage of list partitioning is that you can group and organize unordered and unrelated sets of data in a natural way.

Hash Partitioning:

Hash partitioning is a partitioning technique where a hash key is used to distribute rows evenly across the different partitions. Hash partitioning is used where ranges aren't appropriate Ex: employee number, customer ID, etc. Using this approach, data is randomly distributed across the partitions rather than grouped.

1] Range Partitioning:

Code:

```
CREATE TABLE employee_siddhartha(emp_id NUMBER(3),emp_name VARCHAR2(30),emp_add VARCHAR2(50),emp_salary NUMBER(5))
PARTITION BY RANGE(emp_salary)
(
PARTITION s1 VALUES LESS THAN (2000),
PARTITION s2 VALUES LESS THAN (4000),
PARTITION s3 VALUES LESS THAN (6000),
PARTITION s4 VALUES LESS THAN (8000),
PARTITION s5 VALUES LESS THAN (10000)
);
INSERT INTO employee_siddhartha VALUES (1,'Virat','Banglore',9500);
INSERT INTO employee_siddhartha VALUES (2,'Rohit','Mumbai',8500);
INSERT INTO employee_siddhartha VALUES (3,'Rahul','Lucknow',7000);
INSERT INTO employee_siddhartha VALUES (4,'Sachin','Mumbai',6500);
INSERT INTO employee_siddhartha VALUES (5,'Sehvag','Delhi',4200);


SELECT * FROM employee_siddhartha;
SELECT * FROM employee_siddhartha PARTITION (s4);
```

Output:

```
Table created.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

EMP_ID	EMP_NAME	EMP_ADD	EMP_SALARY
5	Sehvag	Delhi	4200
3	Rahul	Lucknow	7000
4	Sachin	Mumbai	6500
1	Virat	Banglore	9500
2	Rohit	Mumbai	8500

Download CSV
5 rows selected.

EMP_ID	EMP_NAME	EMP_ADD	EMP_SALARY
3	Rahul	Lucknow	7000
4	Sachin	Mumbai	6500

Download CSV

2] List Partitioning:

Code:

```
CREATE TABLE employee_siddhartha(emp_id NUMBER(3),emp_name VARCHAR2(30),emp_add VARCHAR2(50),emp_salary NUMBER(5))
PARTITION BY LIST(emp_add)
(
PARTITION a1 VALUES ('Thane','Mulund'),
PARTITION a2 VALUES ('Kurla'),
PARTITION a3 VALUES ('Ghatkopar'),
PARTITION a4 VALUES ('Kalyan','Badlapur'),
PARTITION a5 VALUES ('Chembur','Nerul','Panvel')
)ENABLE ROW MOVEMENT;
-- SELECT TABLE_NAME, PARTITION_NAME FROM USER_TAB_PARTITION;
INSERT INTO employee_siddhartha VALUES (1,'Virat','Mulund',9500);
INSERT INTO employee_siddhartha VALUES (2,'Rohit','Thane',8500);
INSERT INTO employee_siddhartha VALUES (3,'Rahul','Badlapur',7000);
INSERT INTO employee_siddhartha VALUES (4,'Sachin','Kurla',6500);
INSERT INTO employee_siddhartha VALUES (5,'Sehvag','Kalyan',4200);
INSERT INTO employee_siddhartha VALUES (6,'Gambhir','Chembur',2200);
INSERT INTO employee_siddhartha VALUES (7,'Bumrah','Panvel',3800);


SELECT * FROM employee_siddhartha;
SELECT * FROM employee_siddhartha PARTITION (a1);
```

Output:

```
Table created.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

EMP_ID	EMP_NAME	EMP_ADD	EMP_SALARY
1	Virat	Mulund	9500
2	Rohit	Thane	8500
4	Sachin	Kurla	6500
3	Rahul	Badlapur	7000
5	Sehvag	Kalyan	4200
6	Gambhir	Chembur	2200
7	Bumrah	Panvel	3800

Download CSV
7 rows selected.

EMP_ID	EMP_NAME	EMP_ADD	EMP_SALARY
1	Virat	Mulund	9500
2	Rohit	Thane	8500

Download CSV
2 rows selected.

3] Hash Partitioning:

Code:

```
Select SQL Plus
1 row created.

SQL> drop table employee_siddhartha;

Table dropped.

SQL> create table employee_siddhartha(
  2  emp_id NUMERIC(10) NOT NULL,
  3  emp_name VARCHAR2(50) NOT NULL,
  4  dept VARCHAR2(25) NOT NULL,
  5  emp_loc VARCHAR2(25) NOT NULL,
  6  doj DATE)
  7  PARTITION BY HASH (emp_id)
  8  (
  9  PARTITION A1,
 10  PARTITION A2,
 11  PARTITION A3
 12  );

Table created.

SQL> insert into employee_siddhartha values(1,'josh','cric','eng','1-mar-2000');

1 row created.

SQL> insert into employee_siddhartha values(2,'virat','cric','ind','31-may-2000');

1 row created.

SQL> insert into employee_siddhartha values(3,'ronaldo','foot','port','11-july-1999');

1 row created.

SQL> insert into employee_siddhartha values(4,'david','tenis','aus','17-sept-1999');
insert into employee_siddhartha values(4,'david','tenis','aus','17-sept-1999')
                                        *
ERROR at line 1:
ORA-01841: (full) year must be between -4713 and +9999, and not be 0


SQL> insert into employee_siddhartha values(4,'david','tenis','aus','17-sep-1999');

1 row created.
```


Output:

```
SQL> SELECT TABLE_NAME, PARTITION_NAME FROM USER_TAB_PARTITIONS WHERE TABLESPACE_NAME='USERS';

TABLE_NAME                     PARTITION_NAME
------------------------------ ------------------------------
BIN$Vw4RjWIKQrmmUB9EeuomVQ==$0 SYS_P85
EMPLOYEE_SIDDHARTHA            A1
EMPLOYEE_SIDDHARTHA            A2
EMPLOYEE_SIDDHARTHA            A3

SQL> select * from employee_siddhartha PARTITION(A1);

no rows selected

SQL> select * from employee_siddhartha PARTITION(A2);

    EMP_ID EMP_NAME
---------- --------------------------------------------------
DEPT                      EMP_LOC                   DOJ
------------------------- ------------------------- ---------
         1 josh
cric                      eng                       01-MAR-00

         3 ronaldo
foot                      port                      11-JUL-99

         4 david
tenis                     aus                       17-SEP-99

SQL> select * from employee_siddhartha PARTITION(A3);

    EMP_ID EMP_NAME
---------- --------------------------------------------------
DEPT                      EMP_LOC                   DOJ
------------------------- ------------------------- ---------
         2 virat
cric                      ind                       31-MAY-00
```

Conclusion: I learned different type of data partitioning from this assignment.