

## Practical-2

**Aim:** Implementation of Data Partitioning through Range and List Partitioning

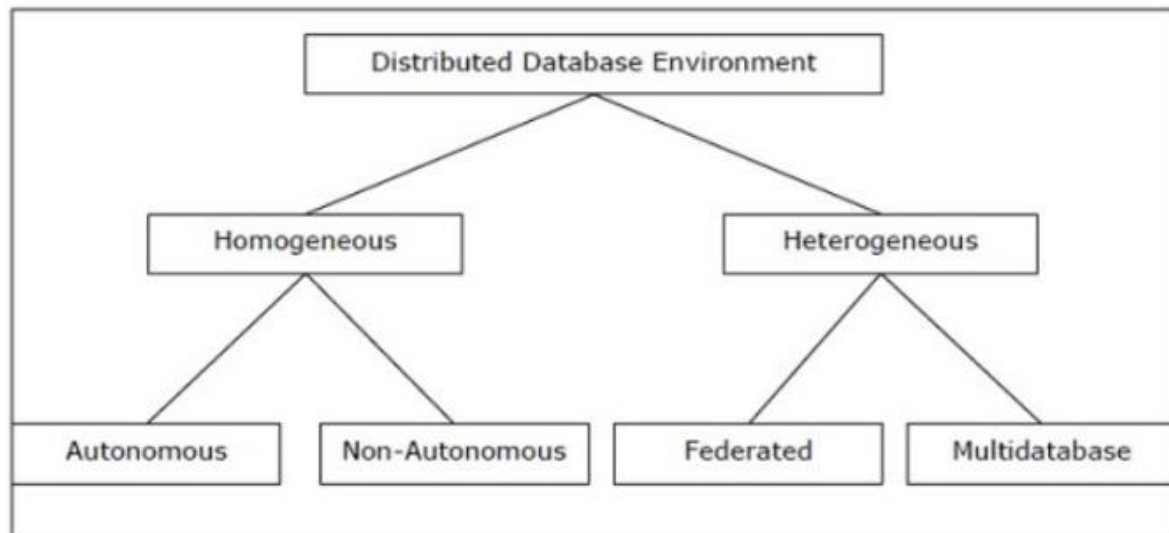
### **Distributed Databases:**

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
- Each site is aware of all other sites and cooperates with other sites to process user requests.
- 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 priori 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; otherwise 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.

**1] Range Partitioning :****Code :**

```

1 create table sales_range_suraj(
2   salesman_id NUMBER(5),
3   salesman_name VARCHAR2(30),
4   sales_amount NUMBER(10),
5   sales_date DATE)
6   PARTITION BY RANGE (sales_date)
7   (
8     PARTITION sales_jan2000 VALUES LESS
9     THAN(TO_DATE('01/01/2000','DD/MM/YYYY')),
10    PARTITION sales_feb2000 VALUES LESS
11    THAN(TO_DATE('01/02/2000','DD/MM/YYYY')),
12    PARTITION sales_mar2000 VALUES LESS
13    THAN(TO_DATE('01/03/2000','DD/MM/YYYY')),
14    PARTITION sales_apr2000 VALUES LESS
15    THAN(TO_DATE('01/04/2000','DD/MM/YYYY')),
16    PARTITION sales_may2000 VALUES LESS
17    THAN(TO_DATE('01/05/2000','DD/MM/YYYY'))
18  );
19
20 SELECT TABLE_NAME, PARTITION_NAME FROM USER_TAB_PARTITIONS WHERE
21 TABLESPACE_NAME='sales_range_suraj';
22 insert into sales_range_suraj values(1,'suraj suryawanshi',10000,TO_DATE('13/05/2000','DD/MM/YYYY'));
23 insert into sales_range_sachin values(2,'eldon pillai',20000,TO_DATE('11/02/2000','DD/MM/YYYY'));
24 insert into sales_range_suraj values(3,'john navghe',30000,TO_DATE('10/03/2000','DD/MM/YYYY'));
25 insert into sales_range_suraj values(4,'drishti bhatia',40000,TO_DATE('25/04/2000','DD/MM/YYYY'));
26 insert into sales_range_suraj values(5,'satyam sisodia',50000,TO_DATE('03/01/2000','DD/MM/YYYY'));
27
28 select * from sales_range_suraj;
29 select * from sales_range_suraj PARTITION(sales_feb2000)

```

**Output:**

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

SALESMAN_ID	SALESMAN_NAME	SALES_AMOUNT	SALES_DATE
5	satyam sisodia	50000	03-JAN-00
3	john navghe	30000	10-MAR-00
4	drishti bhatia	40000	25-APR-00

[Download CSV](#)

3 rows selected.

SALESMAN_ID	SALESMAN_NAME	SALES_AMOUNT	SALES_DATE
5	satyam sisodia	50000	03-JAN-00

[Download CSV](#)

## 2] List Partitioning :

### Code:

```
1 create table sales_list_suraj(  
2 salesman_id NUMBER(5),  
3 salesman_name VARCHAR2(30),  
4 sales_state varchar2(30),  
5 sales_amount NUMBER(10),  
6 sales_date DATE)  
7 PARTITION BY LIST (sales_state)  
8 (  
9 PARTITION sales_west VALUES ('Mumbai','Pune'),  
10 PARTITION sales_east VALUES ('Kolkata'),  
11 PARTITION sales_south VALUES ('Chennai'),  
12 PARTITION sales_north VALUES ('Delhi'),  
13 PARTITION sales_other VALUES (Default)  
14 ) enable row movement;  
15 SELECT TABLE_NAME, PARTITION_NAME FROM USER_TAB_PARTITIONS;  
16  
17 insert into sales_list_suraj values(1,'suraj  
18 suryawanshi','Mumbai',10000,TO_DATE('10/01/2000','DD/MM/YYYY'));  
19 insert into sales_list_suraj values(2,'john  
20 joe','Pune',20000,TO_DATE('10/02/2000','DD/MM/YYYY'));  
21 insert into sales_list_suraj values(3,'om  
22 navghe','Delhi',30000,TO_DATE('10/03/2000','DD/MM/YYYY'));  
23 insert into sales_list_suraj values(4,'drishti  
24 bhatia','Kolkata',40000,TO_DATE('10/04/2000','DD/MM/YYYY'));  
25 insert into sales_list_suraj values(5,'Babu  
26 rao','Chennai',50000,TO_DATE('20/04/2000','DD/MM/YYYY'));  
27 insert into sales_list_suraj values(6,'Rupesh  
28 mishra','Allahabad',60000,TO_DATE('10/05/2000','DD/MM/YYYY'));  
29 select * from sales_list_suraj;  
30 select * from sales_list_suraj PARTITION(sales_west)
```

### Output:

TABLE_NAME	PARTITION_NAME
SALES_LIST_SURAJ	SALES_EAST
SALES_LIST_SURAJ	SALES_NORTH
SALES_LIST_SURAJ	SALES_OTHER
SALES_LIST_SURAJ	SALES_SOUTH
SALES_LIST_SURAJ	SALES_WEST

[Download CSV](#)

5 rows selected.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

SALESMAN_ID	SALESMAN_NAME	SALES_STATE	SALES_AMOUNT	SALES_DATE
1	suraj suryawanshi	Mumbai	10000	10-JAN-00
2	john joe	Pune	20000	10-FEB-00
1	suraj suryawanshi	Mumbai	10000	10-JAN-00
2	john joe	Pune	20000	10-FEB-00
1	suraj suryawanshi	Mumbai	10000	10-JAN-00
2	john joe	Pune	20000	10-FEB-00
4	drishti bhatia	Kolkata	40000	10-APR-00
4	drishti bhatia	Kolkata	40000	10-APR-00
4	drishti bhatia	Kolkata	40000	10-APR-00
5	Babu rao	Chennai	50000	20-APR-00
5	Babu rao	Chennai	50000	20-APR-00
5	Babu rao	Chennai	50000	20-APR-00
3	om navghe	Delhi	30000	10-MAR-00
3	om navghe	Delhi	30000	10-MAR-00
3	om navghe	Delhi	30000	10-MAR-00
6	Rupesh mishra	Allahabad	60000	10-MAY-00
6	Rupesh mishra	Allahabad	60000	10-MAY-00
6	Rupesh mishra	Allahabad	60000	10-MAY-00

[Download CSV](#)

SALESMAN_ID	SALESMAN_NAME	SALES_STATE	SALES_AMOUNT	SALES_DATE
1	suraj suryawanshi	Mumbai	10000	10-JAN-00
2	john joe	Pune	20000	10-FEB-00
1	suraj suryawanshi	Mumbai	10000	10-JAN-00
2	john joe	Pune	20000	10-FEB-00
1	suraj suryawanshi	Mumbai	10000	10-JAN-00
2	john joe	Pune	20000	10-FEB-00

[Download CSV](#)


6 rows selected.

## Hash Partitioning:

Hash partitioning is used to make sure that data is evenly scattered into a certain number of partitions. With Range partitioning, you must specify the range of the column values for each partition when you use Range partitioning, while you just need to specify the number of partitions when you use Hash partitioning.

### 3] Hash Partitioning:

#### Code:

 SQL Plus

SQL\*Plus: Release 12.2.0.1.0 Production on Wed Nov 30 16:27:34 2022

Copyright (c) 1982, 2016, Oracle. All rights reserved.

Enter user-name: user60b@sem1

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

```
SQL> CREATE TABLE employee_suraj_suryawanshi(  
  2 emp_id NUMERIC(10) NOT NULL,  
  3 emp_name VARCHAR2(50) NOT NULL,  
  4 dept VARCHAR2(25) NOT NULL,  
  5 emp_location VARCHAR2(25) NOT NULL,  
  6 dob DATE)  
  7 PARTITION BY HASH (emp_id);
```

Table created.

```
SQL> INSERT INTO employee_suraj_suryawanshi VALUES(101,'rutik',15000);  
1 row created.  
  
SQL> INSERT INTO employee_suraj_suryawanshi VALUES(102,'suraj',9500);  
1 row created.  
  
SQL> INSERT INTO employee_suraj_suryawanshi VALUES(103,'tarun',11000);  
1 row created.  
  
SQL> INSERT INTO employee_suraj_suryawanshi VALUES(104,'sidharth',14500);  
1 row created.  
  
SQL> INSERT INTO employee_suraj_suryawanshi VALUES(105,'nikhil',11500);  
1 row created.  
  
SQL> INSERT INTO employee_suraj_suryawanshi VALUES(106,'ayush',10000);  
1 row created.  
  
SQL> INSERT INTO employee_suraj_suryawanshi VALUES(107,'sree',15000);  
1 row created.  
  
SQL> INSERT INTO employee_suraj_suryawanshi VALUES(108,'atharva',17500);  
1 row created.  
  
SQL> INSERT INTO employee_suraj_suryawanshi VALUES(109,'mandar',7500);  
1 row created.  
  
SQL> INSERT INTO employee_suraj_suryawanshi VALUES(110,'ram',5000);  
1 row created.
```

```
SQL> SELECT * FROM employee_mayur PARTITION(P1);
```

EMP_NO	EMP_NAME	EMP_SALARY
108	atharva	175000

```
SQL> SELECT * FROM employee_mayur PARTITION(P2);
```

EMP_NO	EMP_NAME	EMP_SALARY
101	rutik	15000
104	sidharth	14500
106	ayush	100000
110	ram	50000

```
SQL> SELECT * FROM employee_mayur PARTITION(P3);
```

EMP_NO	EMP_NAME	EMP_SALARY
100	mayur	10000
102	suraj	9500
103	tarun	11000
105	nikhil	11500
107	sree	150000
109	mandar	75000

6 rows selected.

```
SQL>
```