ADBMS Experiment 03

Name: Sakshi Dhenge

UID: 2020400007

Batch: B

Aim: To perform vertical fragmentation on the given tables

Theory:

- Vertical fragmentation is a database design technique used to divide a large table into smaller tables based on their columns. This technique helps to improve query performance by reducing the amount of data that needs to be scanned by the database engine.
- By breaking a large table into smaller tables, the database engine can scan a smaller set of data to answer a query, which can result in faster query performance.
- Vertical fragmentation can also improve data management by separating related columns into smaller, more manageable tables.
- If a particular column is frequently queried, it can be placed in a separate table to improve performance. Alternatively, if a column is rarely used, it can be placed in a separate table to reduce the size of the main table and improve performance.

Creation and data insertion in bank_details table :

```
CREATE TABLE bank_details (
      acc no INT,
      cust_id INT,
      cust name VARCHAR (50),
      mob no bigint,
      branch VARCHAR (50),
      acc bal INT,
      loan amt INT,
      amt_due INT,
      dob DATE,
      trans no INT,
      trans_date DATE,
      trans mode VARCHAR(20),
      trans type VARCHAR(20),
      trans amt INT,
      PRIMARY KEY (acc no, cust id)
  );
```

```
INSERT INTO bank_details VALUES

(10001, 1, 'Priya Sharma', 9876543210, 'New Delhi', 5000, 1000, 500, '1990-01-01', 1, '2022-12-01', 'NEFT', 'Deposit', 5000),

(10002, 2, 'Anand Patel', 9876543211, 'Mumbai', 6000, 2000, 1500, '1980-02-01', 2, '2022-10-02', 'Cash', 'Withdrawal', 1500),

(10003, 3, 'Neha Singh', 9876543212, 'Chennai', 7000, 2500, 500, '1985-03-01', 3, '2021-02-03', 'Cheque', 'Deposit', 5000),

(10004, 4, 'Rajesh Kaur', 9876543213, 'Hyderabad', 8000, 3000, 1000, '1987-04-01', 4, '2020-12-04', 'NEFT', 'Withdrawal', 6000),

(10005, 5, 'Mohan Kumar', 9876543214, 'Bangalore', 9000, 3500, 1500, '1989-05-01', 5, '2021-10-15', 'Online', 'Deposit', 1000),

(10006, 6, 'Sunita Verma', 9876543215, 'Lucknow', 10000, 4000, 2000, '1981-06-01', 6, '2022-03-20', 'Cash', 'Withdrawal', 3000),

(10007, 7, 'Kunal Shah', 9876543216, 'Jaipur', 11000, 4500, 2500, '1983-07-01', 7, '2018-10-18', 'Cheque', 'Deposit', 2000),

(10008, 8, 'Madhuri Mehta', 9876543217, 'Kolkata', 12000, 5000, 3000, '1985-08-01', 8, '2020-06-08', 'Online', 'Withdrawal', 7000),

(10009, 9, 'Ravi Patel', 9876543218, 'Pune', 13000, 5500, 3500, '1987-09-01', 9, '2019-11-09', 'NEFT', 'Withdrawal', 10000),

(10011, 11, 'Rohan Shah', 9876543212, 'Ahmedabad', 15000, 6500, 4500, '1981-10-01', 10, '2020-09-14', 'Cash', 'Withdrawal', 5000),

(10013, 13, 'Amit Kumar', 9876543223, 'Bhopal', 17000, 7500, 5500, '1984-01-01', 13, '2022-12-13', 'Online', 'Withdrawal', 1000),

(10014, 14, 'Simran Kaur', 9876543224, 'Vishakhapatnam', 18000, 8000, 6000, '1986-02-01', 14, '2022-12-14', 'NEFT', 'Deposit', 5000),

(10015, 15, 'Vikas Mehta', 9876543226, 'Patna', 19000, 8500, 6500, '1988-03-01', 15, '2022-12-15', 'Cash', 'Withdrawal', 2000)
```

Data	Output	Explain	Messages	History											
			cust_name character va	rying(50)	mob_no bigint	branch character varying(50)					trans_no integer	trans_date	trans_mode character varying(20)	trans_type character varying(20)	trans_amt integer
1	10001	1	Priya Sharm	na	9876543	New Delhi	5000	1000	500	1990	1	2022-12-0	NEFT	Deposit	5000
2	10002	2	Anand Patel	1	9876543	Mumbai	6000	2000	1500	1980	2	2022-10-0	Cash	Withdrawal	1500
3	10003	3	Neha Singh		9876543	Chennai	7000	2500	500	1985	3	2021-02-0	Cheque	Deposit	5000
4	10004	4	Rajesh Kaur	r	9876543	Hyderabad	8000	3000	1000	1987	4	2020-12-0	NEFT	Withdrawal	6000
5	10005	5	Mohan Kuman	r	9876543	Bangalore	9000	3500	1500	1989	5	2021-10-1	Online	Deposit	1000
6	10006	6	Sunita Verm	na	9876543	Lucknow	10000	4000	2000	1981	6	2022-03-2	Cash	Withdrawal	3000
7	10007	7	Kunal Shah		9876543	Jaipur	11000	4500	2500	1983	7	2018-10-1	Cheque	Deposit	2000
8	10008	8	Madhuri Meh	nta	9876543	Kolkata	12000	5000	3000	1985	8	2020-06-0	Online	Withdrawal	7000
9	10009	9	Ravi Patel		9876543	Pune	13000	5500	3500	1987	9	2019-11-0	NEFT	Withdrawal	10000
10	10010	10	Tanvi Patel	L	9876543	Chandigarh	14000	6000	4000	1989	10	2020-09-1	Cash	Withdrawal	5000
11	10011	11	Rohan Shah		9876543	Ahmedabad	15000	6500	4500	1981	11	2022-11-3	Cheque	Deposit	4000
12	10013	13	Amit Kumar		9876543	Bhopal	17000	7500	5500	1984	13	2022-12-1	Online	Withdrawal	1000
13	10014	14	Simran Kaur		9876543	Vishakhapatnam	18000	8000	6000	1986	14	2022-12-1	NEFT	Deposit	5000
14	10015	15	Vikas Mehta	3	9876543	Patna	19000	8500	6500	1988	15	2022-12-1	Cash	Withdrawal	2000

Vertical Fragmentation of bank details table

Creation and insertion of data in cust table:

```
CREATE TABLE cust (
    acc_no INT,
    cust_id INT,
    cust_name VARCHAR(50),
    mob_no bigint,
    branch VARCHAR(50),
    acc_bal INT,
    dob DATE,
    PRIMARY KEY (acc_no, cust_id)
);

INSERT INTO cust(select acc_no, cust_id, cust_name, mob_no, branch, acc_bal, dob from bank_details);
```

Data Ou	tput	Explain	Messages History				
	acc_no nteger		cust_name character varying(50)	mob_no bigint	branch character varying(50)	acc_bal integer	
1 1	0001	1	Priya Sharma	9876543	New Delhi	5000	1990
2 1	10002	2	Anand Patel	9876543	Mumbai	6000	1980
3 1	10003	3	Neha Singh	9876543	Chennai	7000	1985
4 1	10004	4	Rajesh Kaur	9876543	Hyderabad	8000	1987
5 1	10005	5	Mohan Kumar	9876543	Bangalore	9000	1989
6 1	10006	6	Sunita Verma	9876543	Lucknow	10000	1981
7 1	10007	7	Kunal Shah	9876543	Jaipur	11000	1983
8 1	8000	8	Madhuri Mehta	9876543	Kolkata	12000	1985
9 1	10009	9	Ravi Patel	9876543	Pune	13000	1987
10 1	0010	10	10 Tanvi Patel		Chandigarh	14000	1989
11 1	1 10011 11 Rohan Shah		9876543	Ahmedabad	15000	1981	
12 1	2 10013 13 Amit Kumar		9876543	Bhopal	17000	1984	
13 1	0014 14 Simran Kaur		9876543	Vishakhapatnam	18000	1986	
14 1	0015	15	Vikas Mehta	9876543	Patna	19000	1988

Creation and insertion of data in cust_loan table :

```
CREATE TABLE cust_loan (
    acc_no INT,
    cust_id INT,
    loan_amt INT,
    amt_due INT,
    PRIMARY KEY (acc_no, cust_id)
);
INSERT INTO cust_loan (SELECT acc_no, cust_id, loan_amt, amt_due FROM bank_details);
```

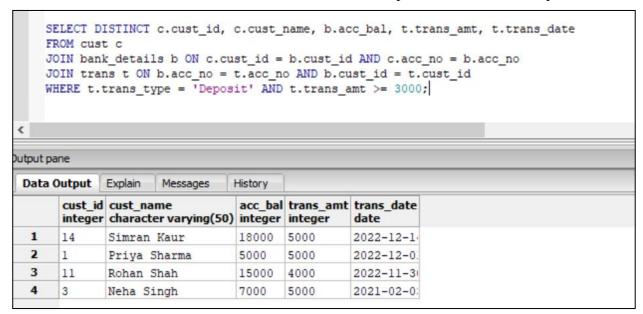
	Y		Y	Υ
Data	Output	Explain	Messages	History
		cust_id integer	loan_amt integer	amt_due integer
1	10001	1	1000	500
2	10002	2	2000	1500
3	10003	3	2500	500
4	10004	4	3000	1000
5	10005	5	3500	1500
6	10006	6	4000	2000
7	10007	7	4500	2500
8	10008	8	5000	3000
9	10009	9	5500	3500
10	10010	10	6000	4000
11	10011	11	6500	4500
12	10013	13	7500	5500
13	10014	14	8000	6000
14	10015	15	8500	6500

Creation and insertion of data in transaction table:

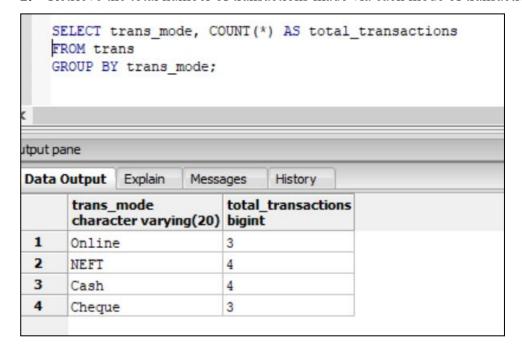
Data Output		Explain	Messages	History			
		cust_id integer		trans_date date	trans_mode character varying(20)	trans_type character varying(20)	trans_amt
1	10001	1	1	2022-12-0	NEFT	Deposit	5000
2	10002	2	2	2022-10-0:	Cash	Withdrawal	1500
3	10003	3	3	2021-02-03	Cheque	Deposit	5000
4	10004	4	4	2020-12-0	NEFT	Withdrawal	6000
5	10005	5	5	2021-10-1	Online	Deposit	1000
6	10006	6	6	2022-03-2	Cash	Withdrawal	3000
7	10007	7	7	2018-10-1	Cheque	Deposit	2000
8	10008	8	8	2020-06-0	Online	Withdrawal	7000
9	10009	9	9	2019-11-0	NEFT	Withdrawal	10000
10	10010	10	10	2020-09-1	Cash	Withdrawal	5000
11	10011	11	11	2022-11-3	Cheque	Deposit	4000
12	10013	13	13	2022-12-13	Online	Withdrawal	1000
13	10014	14	14	2022-12-1	NEFT	Deposit	5000
14	10015	15	15	2022-12-1	Cash	Withdrawal	2000

Observations on Fragmented Tables:

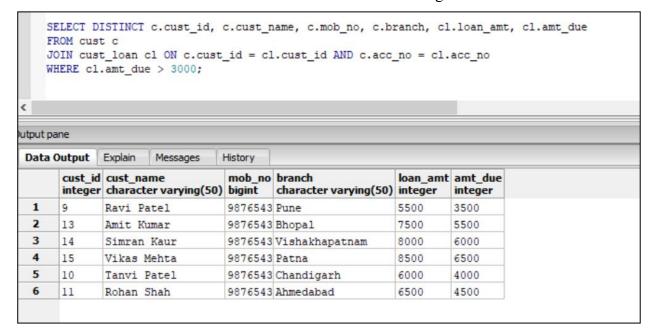
1. Retrieve the details of all customers who have made a deposit of more than or equal to 3000:



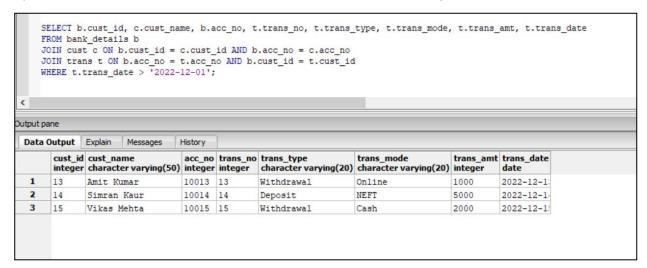
2. Retrieve the total number of transactions made via each mode of transaction:



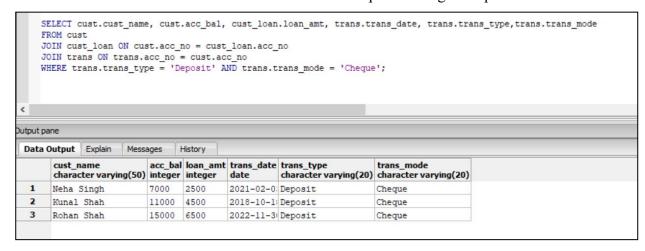
3. Retrieve the details of all customers with due loan amount greater than 3000



4. Retrieve the details of all transactions made after December 2022



5. Retrieve the details of all customers who made deposit through cheque



Conclusion:

- 1. Vertical fragmentation is a useful technique for improving query performance and data management in sql databases.
- 2. Vertical fragmentation helps to reduce query processing time.
- 3. Vertical fragmentation makes easier to update and maintain the data, as well as improve data security by limiting access to specific columns.
- 4. Vertical fragmentation can increase the flexibility of the database design