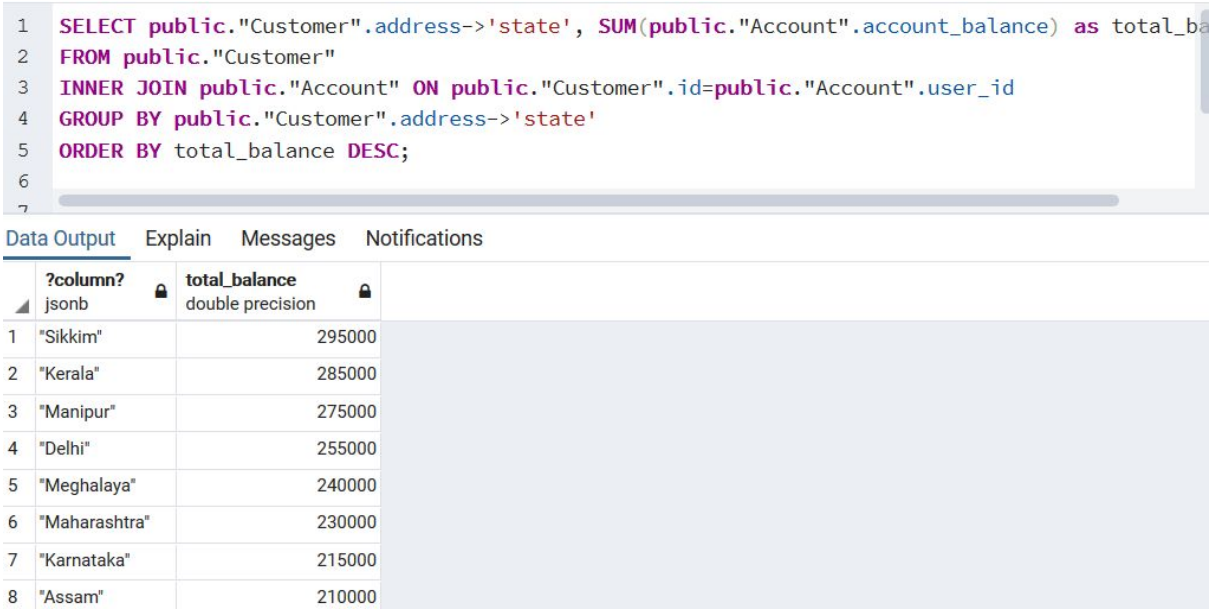
IMPORTANT NOTE : I have used 15 days instead of 10 days and 19 days instead of 14 days (2 weeks). This is because I had updated the database till 10th June. If new rows are entered till the latest day, we can change the respective days (the logic works).

Q2.1.1

```
SELECT public."Customer".address->'state', SUM(public."Account".account_balance) as
total_balance
FROM public."Customer"
INNER JOIN public."Account" ON public."Customer".id=public."Account".user_id
GROUP BY public."Customer".address->'state'
ORDER BY total_balance DESC;
```



```
SELECT public."Customer".address->'state', SUM(public."Account".account_balance) as total_ba
FROM public."Customer"
INNER JOIN public."Account" ON public."Customer".id=public."Account".user_id
GROUP BY public."Customer".address->'state'
ORDER BY total_balance DESC;
```

Data Output | Explain | Messages | Notifications

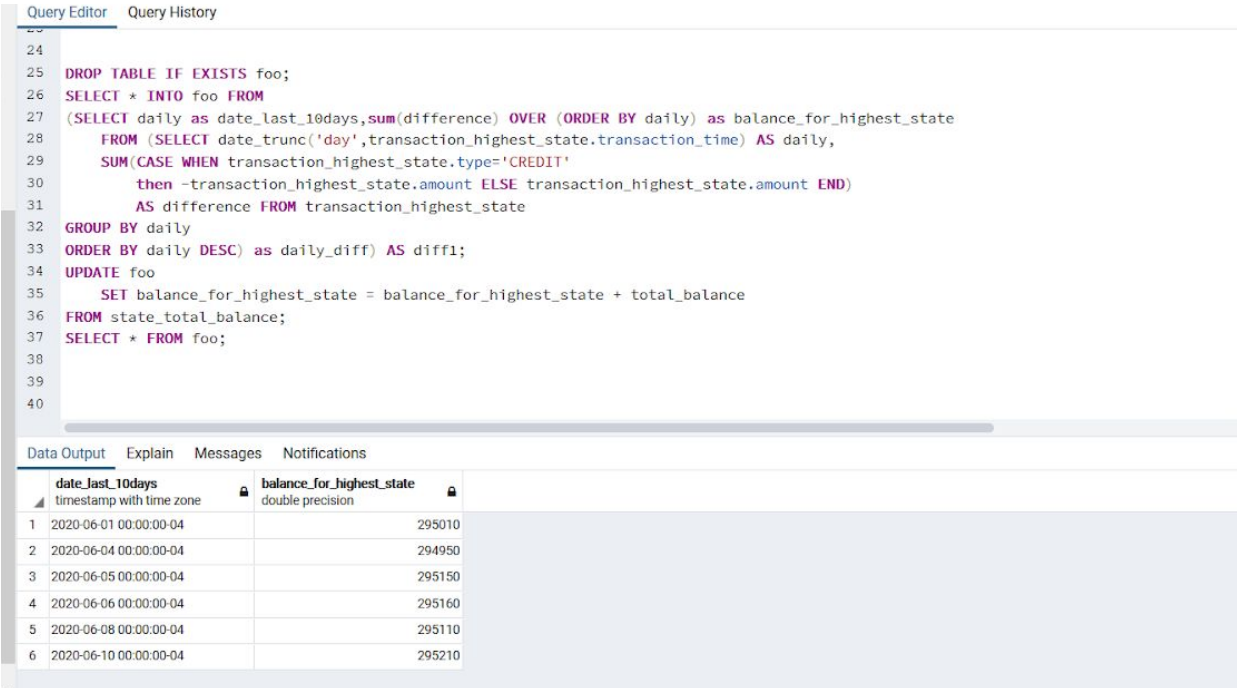
	?column? jsonb	total_balance double precision
1	"Sikkim"	295000
2	"Kerala"	285000
3	"Manipur"	275000
4	"Delhi"	255000
5	"Meghalaya"	240000
6	"Maharashtra"	230000
7	"Karnataka"	215000
8	"Assam"	210000

Q2.2

a.

```
CREATE OR REPLACE VIEW state_total_balance as SELECT total_balance FROM (SELECT
public."Customer".address->'state',

                                        SUM(public."Account".account_balance) as total_balance
FROM public."Customer"
INNER JOIN public."Account" ON public."Customer".id=public."Account".user_id
GROUP BY public."Customer".address->'state'
ORDER BY total_balance DESC LIMIT 1) as st;
SELECT * FROM state_total_balance;


CREATE OR REPLACE VIEW transaction_highest_state as
SELECT * FROM (SELECT public."Transaction".amount, public."Transaction".transaction_time,
public."Transaction".type FROM public."Transaction"
INNER JOIN (SELECT public."Account".id as aid, public."Account".account_balance FROM
public."Account"
INNER JOIN (SELECT public."Customer".id as cid FROM public."Customer"
```

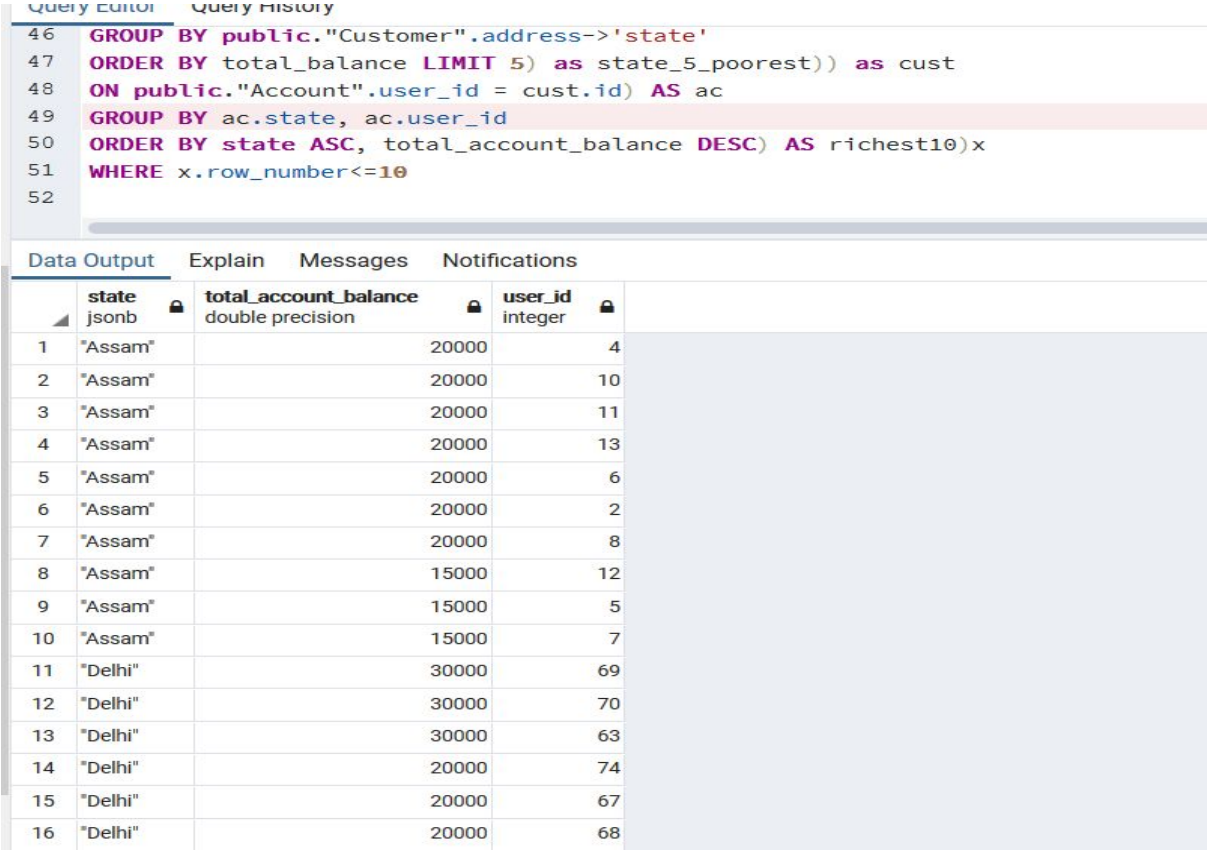
```
WHERE public."Customer".address->'state' = (SELECT state FROM (SELECT
public."Customer".address->'state' as state, SUM(public."Account".account_balance) as total_balance
FROM public."Customer"
INNER JOIN public."Account" ON public."Customer".id=public."Account".user_id
GROUP BY public."Customer".address->'state'
ORDER BY total_balance DESC LIMIT 1) as highest)) as cust
ON public."Account".user_id = cust.cid) as ac
ON public."Transaction".account_id = ac.aid) as tran
WHERE tran."transaction_time" >= NOW() - INTERVAL '15 DAYS'
ORDER BY tran."transaction_time" DESC;


DROP TABLE IF EXISTS foo;
SELECT * INTO foo FROM
(SELECT daily as date_last_10days,sum(difference) OVER (ORDER BY daily) as
balance_for_highest_state
   FROM (SELECT date_trunc('day',transaction_highest_state.transaction_time) AS daily,
   SUM(CASE WHEN transaction_highest_state.type='CREDIT'
         then -transaction_highest_state.amount ELSE transaction_highest_state.amount END)
         AS difference FROM transaction_highest_state
GROUP BY daily
ORDER BY daily DESC) as daily_diff) AS diff1;
UPDATE foo
   SET balance_for_highest_state = balance_for_highest_state + total_balance
FROM state_total_balance;
SELECT * FROM foo;
```



Q2.2
C.

10 richest users belonging to 5 of the poorest states:

```
SELECT x.state, x.total_account_balance, x.user_id FROM(SELECT *,ROW_NUMBER() OVER
(PARTITION BY richest10.state ORDER BY richest10.total_account_balance DESC)
FROM (SELECT ac.state, ac.user_id, SUM(ac.account_balance) as total_account_balance FROM
(SELECT public."Account".id as aid,
public."Account".account_balance,public."Account".user_id,cust.address->'state' as state
FROM public."Account"
INNER JOIN (SELECT * FROM public."Customer"
WHERE public."Customer".address->'state' IN (SELECT state from (SELECT
public."Customer".address->'state' as state, SUM(public."Account".account_balance) as total_balance
FROM public."Customer"
INNER JOIN public."Account" ON public."Customer".id=public."Account".user_id
GROUP BY public."Customer".address->'state'
ORDER BY total_balance LIMIT 5) as state_5_poorest)) as cust
ON public."Account".user_id = cust.id) AS ac
GROUP BY ac.state, ac.user_id
ORDER BY state ASC, total_account_balance DESC) AS richest10)x
WHERE x.row_number<=10
```



```
Query Editor  Query History
46  GROUP BY public."Customer".address->'state'
47  ORDER BY total_balance LIMIT 5) as state_5_poorest)) as cust
48  ON public."Account".user_id = cust.id) AS ac
49  GROUP BY ac.state, ac.user_id
50  ORDER BY state ASC, total_account_balance DESC) AS richest10)x
51  WHERE x.row_number<=10
52
```

Data Output | Explain | Messages | Notifications

	state jsonb	total_account_balance double precision	user_id integer
1	"Assam"	20000	4
2	"Assam"	20000	10
3	"Assam"	20000	11
4	"Assam"	20000	13
5	"Assam"	20000	6
6	"Assam"	20000	2
7	"Assam"	20000	8
8	"Assam"	15000	12
9	"Assam"	15000	5
10	"Assam"	15000	7
11	"Delhi"	30000	69
12	"Delhi"	30000	70
13	"Delhi"	30000	63
14	"Delhi"	20000	74
15	"Delhi"	20000	67
16	"Delhi"	20000	68

Last 10 day balance for the 10 richest users of the 5 poorest states

```
CREATE OR REPLACE VIEW richest10trans AS SELECT
public."Transaction".amount,public."Transaction".type,
public."Transaction".transaction_time, ac1.user_id, ac1.state,ac1.total_account_balance FROM
public."Transaction"
INNER JOIN (SELECT public."Account".id as
aid,richestusers.state,richestusers.total_account_balance,richestusers.user_id as user_id FROM
public."Account"
INNER JOIN (SELECT x.state, x.total_account_balance, x.user_id FROM(SELECT
*,ROW_NUMBER() OVER (PARTITION BY richest10.state

                                                    ORDER BY
richest10.total_account_balance DESC)
FROM (SELECT ac.state, ac.user_id, SUM(ac.account_balance) as total_account_balance
          FROM (SELECT public."Account".id as aid,
public."Account".account_balance,public."Account".user_id,cust.address->'state' as state
FROM public."Account"
INNER JOIN (SELECT * FROM public."Customer"
WHERE public."Customer".address->'state' IN (SELECT state from (SELECT
public."Customer".address->'state' as state,

SUM(public."Account".account_balance) as total_balance
FROM public."Customer"
INNER JOIN public."Account" ON public."Customer".id=public."Account".user_id
GROUP BY public."Customer".address->'state'
ORDER BY total_balance LIMIT 5) as state_5_poorest)) as cust
ON public."Account".user_id = cust.id) AS ac
GROUP BY ac.state, ac.user_id
ORDER BY state ASC, total_account_balance DESC) AS richest10)x
WHERE x.row_number<=10) AS richestusers
ON public."Account".user_id = richestusers.user_id) AS ac1
ON public."Transaction".account_id = ac1.aid
WHERE public."Transaction"."transaction_time" >= NOW() - INTERVAL '15 DAYS';

DROP TABLE IF EXISTS foo;
SELECT * INTO foo FROM
   (SELECT daily as date_last_10_days, user_id, total_account_balance,
   sum(difference) OVER (PARTITION BY user_id ORDER BY daily) as balance
   FROM (SELECT date_trunc('day',richest10trans.transaction_time) AS
daily,user_id,AVG(total_account_balance) as total_account_balance,
                    SUM(CASE WHEN richest10trans.type='CREDIT'
                    then -richest10trans.amount ELSE richest10trans.amount END)
                    AS difference FROM richest10trans
   GROUP BY daily, user_id
   ORDER BY user_id ASC, daily DESC) as dailydiff) as d1;
UPDATE foo
   SET balance = balance + total_account_balance;
SELECT * FROM foo;
```

```
            AS difference FROM richest10trans
    GROUP BY daily, user_id
    ORDER BY user_id ASC, daily DESC) as dailydiff) as d1;

UPDATE foo
    SET balance = balance + total_account_balance;
SELECT * FROM foo;


-- SELECT date_trunc('day',richest10trans.transaction_time) AS daily,user_id,AVG(total_account_balance) as total_account_balance,
--      SUM(CASE WHEN richest10trans.type='CREDIT'
--      then -richest10trans.amount ELSE richest10trans.amount END)
--      AS difference FROM richest10trans
-- GROUP BY daily, user_id
```

Data Output | Explain | Messages | Notifications

	date_last_10_days timestamp with time zone	user_id integer	total_account_balance double precision	balance double precision
1	2020-06-04 00:00:00-04	2	20000	19850
2	2020-06-08 00:00:00-04	2	20000	19860
3	2020-06-02 00:00:00-04	4	20000	20020
4	2020-06-06 00:00:00-04	4	20000	19920
5	2020-06-07 00:00:00-04	4	20000	20120
6	2020-06-10 00:00:00-04	5	15000	14975
7	2020-06-09 00:00:00-04	6	20000	20300
8	2020-06-10 00:00:00-04	6	20000	20350
9	2020-06-08 00:00:00-04	7	15000	14900
10	2020-06-04 00:00:00-04	8	20000	19900
11	2020-06-10 00:00:00-04	8	20000	20100

Q2.2
d.

Weekly sum of daily differences between c(Credit) and d(Debit)

SELECT date_trunc('day',public."Transaction".transaction_time) AS daily,
    SUM(CASE WHEN public."Transaction".type='CREDIT'
        then public."Transaction".amount ELSE -public."Transaction".amount END)
        AS difference FROM public."Transaction"
GROUP BY daily
ORDER BY daily DESC

```
SELECT date_trunc('day',public."Transaction".transaction_time) AS daily,
    SUM(CASE WHEN public."Transaction".type='CREDIT'
        then public."Transaction".amount ELSE -public."Transaction".amount END)
        AS difference FROM public."Transaction"
GROUP BY daily
ORDER BY daily DESC
```

a Output | Explain | Messages | Notifications

daily timestamp with time zone	difference double precision
2020-06-10 00:00:00-04	-246.72
2020-06-09 00:00:00-04	-690
2020-06-08 00:00:00-04	-170
2020-06-07 00:00:00-04	-240
2020-06-06 00:00:00-04	50
2020-06-05 00:00:00-04	-145
2020-06-04 00:00:00-04	210
2020-06-03 00:00:00-04	-210
2020-06-02 00:00:00-04	-10
2020-06-01 00:00:00-04	-260

Q2.3.

a.

Top 5 users from each state based on sum of transaction amount in the last 2 weeks

SELECT state, user_id FROM (SELECT *, ROW_NUMBER() OVER (PARTITION BY top5.state
ORDER BY top5.sum_of_transactions DESC)
FROM (SELECT state, daily, user_id, SUM(amount) as sum_of_transactions FROM (SELECT
amount,date_trunc('day',public."Transaction".transaction_time) AS daily,transaction_time, ac.aid as
aid,
ac.user_id as user_id, ac.state as state FROM public."Transaction"
INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust.state FROM
public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id
FROM public."Customer") as cust
ON public."Account".user_id = cust.cust_id) as ac
ON public."Transaction".account_id = ac.aid
WHERE public."Transaction".transaction_time >= NOW() - INTERVAL '19 DAYS') as tranmax
GROUP BY state, daily, user_id
ORDER BY state ASC, sum_of_transactions DESC) as top5)x
WHERE x.row_number<=5
ORDER BY user_id ASC

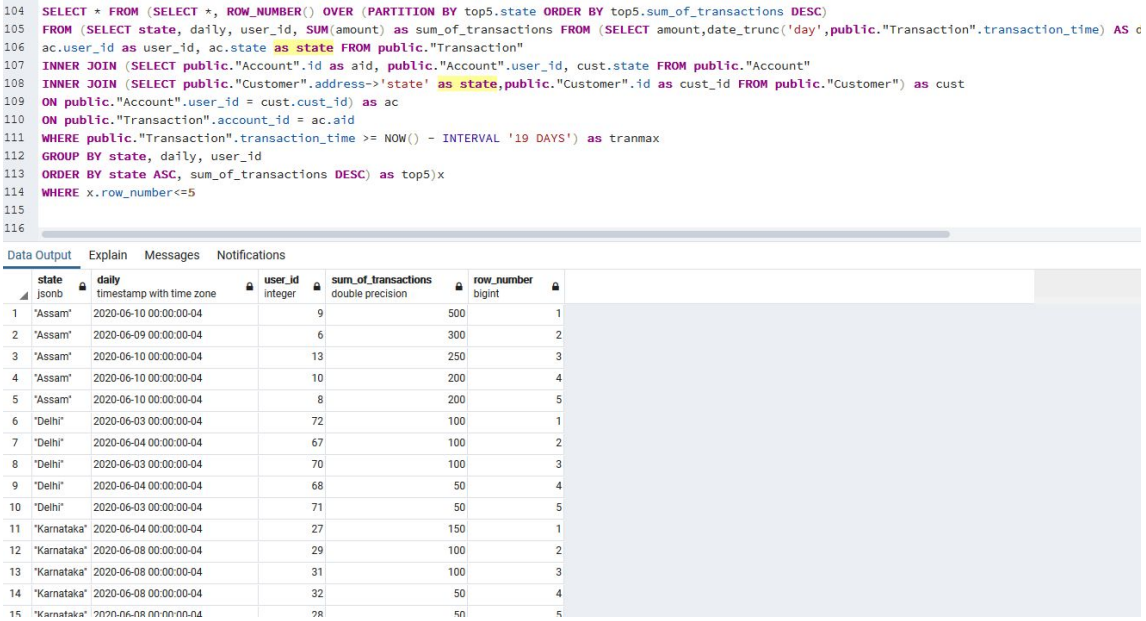
```
103
104 SELECT state, user_id FROM (SELECT *, ROW_NUMBER() OVER (PARTITION BY top5.state ORDER BY top5.sum_of_transactions DESC)
105 FROM (SELECT state, daily, user_id, SUM(amount) as sum_of_transactions FROM (SELECT amount,date_trunc('day',public."Transaction".transaction_time) AS
106 ac.user_id as user_id, ac.state as state FROM public."Transaction"
107 INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust.state FROM public."Account"
108 INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id FROM public."Customer") as cust
109 ON public."Account".user_id = cust.cust_id) as ac
110 ON public."Transaction".account_id = ac.aid
111 WHERE public."Transaction".transaction_time >= NOW() - INTERVAL '19 DAYS') as tranmax
112 GROUP BY state, daily, user_id
113 ORDER BY state ASC, sum_of_transactions DESC) as top5)x
114 WHERE x.row_number<=5
115 ORDER BY user_id ASC
116
```

Data Output | Explain | Messages | Notifications

	state jsonb	user_id integer
1	"Assam"	6
2	"Assam"	8
3	"Assam"	9
4	"Assam"	10
5	"Assam"	13
6	"Kerala"	16
7	"Kerala"	17
8	"Kerala"	21
9	"Kerala"	24
10	"Kerala"	25
11	"Karnataka"	27
12	"Karnataka"	28
13	"Karnataka"	29
14	"Karnataka"	31
15	"Karnataka"	32

Sum of the transaction amount for each of these users

```
SELECT * FROM (SELECT *, ROW_NUMBER() OVER (PARTITION BY top5.state ORDER BY top5.sum_of_transactions DESC)
FROM (SELECT state, daily, user_id, SUM(amount) as sum_of_transactions FROM (SELECT amount,date_trunc('day',public."Transaction".transaction_time) AS daily,transaction_time, ac.aid as aid,
ac.user_id as user_id, ac.state as state FROM public."Transaction"
INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust.state FROM public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id FROM public."Customer") as cust
ON public."Account".user_id = cust.cust_id) as ac
ON public."Transaction".account_id = ac.aid
WHERE public."Transaction".transaction_time >= NOW() - INTERVAL '19 DAYS') as tranmax
GROUP BY state, daily, user_id
ORDER BY state ASC, sum_of_transactions DESC) as top5)x
WHERE x.row_number<=5
```



Data Output | Explain | Messages | Notifications

	state jsonb	daily timestamp with time zone	user_id integer	sum_of_transactions double precision	row_number bigint
1	"Assam"	2020-06-10 00:00:00-04	9	500	1
2	"Assam"	2020-06-09 00:00:00-04	6	300	2
3	"Assam"	2020-06-10 00:00:00-04	13	250	3
4	"Assam"	2020-06-10 00:00:00-04	10	200	4
5	"Assam"	2020-06-10 00:00:00-04	8	200	5
6	"Delhi"	2020-06-03 00:00:00-04	72	100	1
7	"Delhi"	2020-06-04 00:00:00-04	67	100	2
8	"Delhi"	2020-06-03 00:00:00-04	70	100	3
9	"Delhi"	2020-06-04 00:00:00-04	68	50	4
10	"Delhi"	2020-06-03 00:00:00-04	71	50	5
11	"Karnataka"	2020-06-04 00:00:00-04	27	150	1
12	"Karnataka"	2020-06-08 00:00:00-04	29	100	2
13	"Karnataka"	2020-06-08 00:00:00-04	31	100	3
14	"Karnataka"	2020-06-08 00:00:00-04	32	50	4
15	"Karnataka"	2020-06-08 00:00:00-04	28	50	5

Sum of transaction amount of these 5 users combined for that state

```
SELECT state, sum(sum_of_transactions) as total_transactions FROM (SELECT *, ROW_NUMBER()
OVER (PARTITION BY top5.state ORDER BY top5.sum_of_transactions DESC)
FROM (SELECT state, daily, user_id, SUM(amount) as sum_of_transactions FROM (SELECT amount,date_trunc('day',public."Transaction".transaction_time) AS daily,transaction_time, ac.aid as aid,
ac.user_id as user_id, ac.state as state FROM public."Transaction"
INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust.state FROM public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id FROM public."Customer") as cust
ON public."Account".user_id = cust.cust_id) as ac
ON public."Transaction".account_id = ac.aid
WHERE public."Transaction".transaction_time >= NOW() - INTERVAL '19 DAYS') as tranmax
GROUP BY state, daily, user_id
```

ORDER BY state ASC, sum_of_transactions DESC) as top5)x
WHERE x.row_number<=5
GROUP BY state
ORDER BY total_transactions DESC

```
Query Editor  Query History

103
104 SELECT state, sum(sum_of_transactions) as total_transactions FROM (SELECT *, ROW_NUMBER() OVER (PARTITION BY top5.state ORDER BY top5.sum_of_transaction
105 FROM (SELECT state, daily, user_id, SUM(amount) as sum_of_transactions FROM (SELECT amount,date_trunc('day',public."Transaction".transaction_time) AS dai
106 ac.user_id as user_id, ac.state as state FROM public."Transaction"
107 INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust.state FROM public."Account"
108 INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id FROM public."Customer") as cust
109 ON public."Account".user_id = cust.cust_id) as ac
110 ON public."Transaction".account_id = ac.aid
111 WHERE public."Transaction".transaction_time >= NOW() - INTERVAL '19 DAYS') as tranmax
112 GROUP BY state, daily, user_id
113 ORDER BY state ASC, sum_of_transactions DESC) as top5)x
114 WHERE x.row_number<=5
115 GROUP BY state
116 ORDER BY total_transactions DESC
```

Data Output | Explain | Messages | Notifications

	state jsonb	total_transactions double precision
1	"Assam"	1450
2	"Sikkim"	600
3	"Karnataka"	450
4	"Meghalay...	450
5	"Kerala"	400
6	"Delhi"	400
7	"Maharas...	350
8	"Manipur"	300

Q2.3.
b.
State wise cumulative daily sum of transaction amount for users from each state who are present in a.

SELECT state, daily, sum(difference) OVER (Partition by state Order by daily) as cummulative_amount
FROM (SELECT state, daily, sum(amount) as difference FROM (SELECT amount,date_trunc('day',public."Transaction".transaction_time) AS daily,transaction_time, ac1.aid as aid,
ac1.user_id as user_id, ac1.state as state FROM public."Transaction"
INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust1.state FROM public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id FROM public."Customer"
WHERE public."Customer".id in (SELECT user_id FROM (SELECT *, ROW_NUMBER() OVER (PARTITION BY top5.state ORDER BY top5.sum_of_transactions DESC)
FROM (SELECT state, daily, user_id, SUM(amount) as sum_of_transactions FROM (SELECT amount,date_trunc('day',public."Transaction".transaction_time) AS daily,transaction_time, ac.aid as aid,
ac.user_id as user_id, ac.state as state FROM public."Transaction"
INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust.state FROM public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id FROM public."Customer") as cust
ON public."Account".user_id = cust.cust_id) as ac
ON public."Transaction".account_id = ac.aid
WHERE public."Transaction".transaction_time >= NOW() - INTERVAL '19 DAYS') as tranmax
GROUP BY state, daily, user_id
ORDER BY state ASC, sum_of_transactions DESC) as top5)x

WHERE x.row_number<=5
ORDER BY user_id ASC)) as cust1
ON public."Account".user_id = cust1.cust_id) as ac1
ON public."Transaction".account_id = ac1.aid) AS d
GROUP BY state, daily
ORDER BY daily ASC, state ASC) as t

```
SELECT state, daily, sum(difference) OVER (Partition by state Order by daily) as cummulative_amount
FROM (SELECT state, daily, sum(amount) as difference FROM (SELECT amount,date_trunc('day',public."Transaction".transaction_time) AS daily,transaction_time
ac1.user_id as user_id, ac1.state as state FROM public."Transaction"
INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust1.state FROM public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id  FROM public."Customer"
WHERE public."Customer".id in (SELECT user_id FROM (SELECT *, ROW_NUMBER() OVER (PARTITION BY top5.state ORDER BY top5.sum_of_transactions DESC)
FROM (SELECT state, daily, user_id, SUM(amount) as sum_of_transactions FROM (SELECT amount,date_trunc('day',public."Transaction".transaction_time) AS dail
ac.user_id as user_id, ac.state as state FROM public."Transaction"
INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust.state FROM public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id FROM public."Customer") as cust
ON public."Account".user_id = cust.cust_id) as ac
ON public."Transaction".account_id = ac.aid
WHERE public."Transaction".transaction_time >= NOW() - INTERVAL '19 DAYS') as tranmax
```

Data Output | Explain | Messages | Notifications

	state jsonb	daily timestamp with time zone	cummulative_amount double precision
1	"Assam"	2020-05-25 00:00:00-04	10
2	"Assam"	2020-05-26 00:00:00-04	60
3	"Assam"	2020-05-28 00:00:00-04	160
4	"Assam"	2020-06-04 00:00:00-04	310
5	"Assam"	2020-06-09 00:00:00-04	610
6	"Assam"	2020-06-10 00:00:00-04	1810
7	"Delhi"	2020-06-03 00:00:00-04	250
8	"Delhi"	2020-06-04 00:00:00-04	400
9	"Karnataka"	2020-06-04 00:00:00-04	150
10	"Karnataka"	2020-06-08 00:00:00-04	500
11	"Kerala"	2020-06-09 00:00:00-04	400
12	"Maharashtra"	2020-06-01 00:00:00-04	250
13	"Maharashtra"	2020-06-03 00:00:00-04	350
14	"Manipur"	2020-06-06 00:00:00-04	200
15	"Manipur"	2020-06-07 00:00:00-04	300
16	"Meghalaya"	2020-06-04 00:00:00-04	300

Q2.3.

c.

Fetch and display state wise account balance sum of all such users who are present in a.

SELECT sum(public."Account".account_balance) AS total_account_balance,cust1.state FROM public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id FROM public."Customer"
WHERE public."Customer".id in (SELECT user_id FROM (SELECT *, ROW_NUMBER() OVER (PARTITION BY top5.state ORDER BY top5.sum_of_transactions DESC)
FROM (SELECT state, daily, user_id, SUM(amount) as sum_of_transactions FROM (SELECT amount,date_trunc('day',public."Transaction".transaction_time) AS daily,transaction_time, ac.aid as aid,
ac.user_id as user_id, ac.state as state FROM public."Transaction"
INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust.state FROM public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id FROM public."Customer") as cust
ON public."Account".user_id = cust.cust_id) as ac
ON public."Transaction".account_id = ac.aid
WHERE public."Transaction".transaction_time >= NOW() - INTERVAL '19 DAYS') as tranmax
GROUP BY state, daily, user_id
ORDER BY state ASC, sum_of_transactions DESC) as top5)x
WHERE x.row_number<=5

ORDER BY user_id ASC)) as cust1
ON public."Account".user_id = cust1.cust_id
GROUP BY state
ORDER BY total_account_balance DESC

```
SELECT sum(public."Account".account_balance) AS total_account_balance,cust1.state FROM public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id  FROM public."Customer"
WHERE public."Customer".id in (SELECT user_id FROM (SELECT *, ROW_NUMBER() OVER (PARTITION BY top5.state ORDER BY top5.sum_of_transactions DESC)
FROM (SELECT state, daily, user_id, SUM(amount) as sum_of_transactions FROM (SELECT amount,date_trunc('day',public."Transaction".transaction_time) AS
ac.user_id as user_id, ac.state as state FROM public."Transaction"
INNER JOIN (SELECT public."Account".id as aid, public."Account".user_id, cust.state FROM public."Account"
INNER JOIN (SELECT public."Customer".address->'state' as state,public."Customer".id as cust_id FROM public."Customer") as cust
ON public."Account".user_id = cust.cust_id) as ac
ON public."Transaction".account_id = ac.aid
WHERE public."Transaction".transaction_time >= NOW() - INTERVAL '19 DAYS') as tranmax
GROUP BY state, daily, user_id
ORDER BY state ASC, sum_of_transactions DESC) as top5)x
WHERE x.row_number<=5
ORDER BY user_id ASC)) as cust1
ON public."Account".user_id = cust1.cust_id
GROUP BY state
ORDER BY total_account_balance DESC
```

Data Output | Explain | Messages | Notifications

	total_account_balance double precision	state jsonb
1	105000	"Manipur"
2	103000	"Delhi"
3	95000	"Meghalay...
4	95000	"Maharas...
5	94000	"Karnataka"
6	92000	"Kerala"
7	90000	"Assam"
8	70000	"Sikkim"