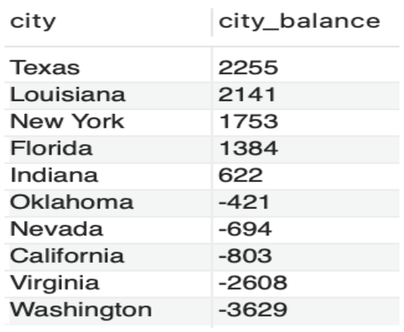
- Find the final transaction balance of each city (the difference between all the money received by the city inhabitants and the money sent) and display the result in descending order of the transaction balance

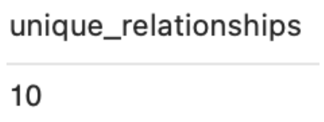
```sql
with city_payments as (
select city, sum(amount) as total_amount_paid
from fact_transactions
join dim_accounts
on fact_transactions.payer_id = dim_accounts.account_id
group by city),

city_receptions as (
select city, sum(amount) as total_amount_received
from fact_transactions
join dim_accounts
on fact_transactions.receiver_id = dim_accounts.account_id
group by city)

select city , total_amount_received - total_amount_paid as city_balance
from city_payments
join city_receptions
using (city)
order by city_balance desc ;
```



city	city_balance
Texas	2255
Louisiana	2141
New York	1753
Florida	1384
Indiana	622
Oklahoma	-421
Nevada	-694
California	-803
Virginia	-2608
Washington	-3629

- A two-way unique relationship is established when two people send money back and forth. Write a query to find the number of two-way unique relationships in this data

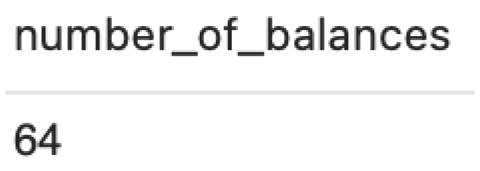
```sql
SELECT FLOOR( COUNT(payer_id)/2 ) AS unique_relationships
FROM (
  SELECT payer_id, receiver_id
  FROM fact_transactions
  INTERSECT
  SELECT receiver_id, payer_id
  FROM fact_transactions) AS relationships;
```



unique_relationships
10

- Find the number of accounts that have a final balance (after all transactions have been completed) greater than $1,000

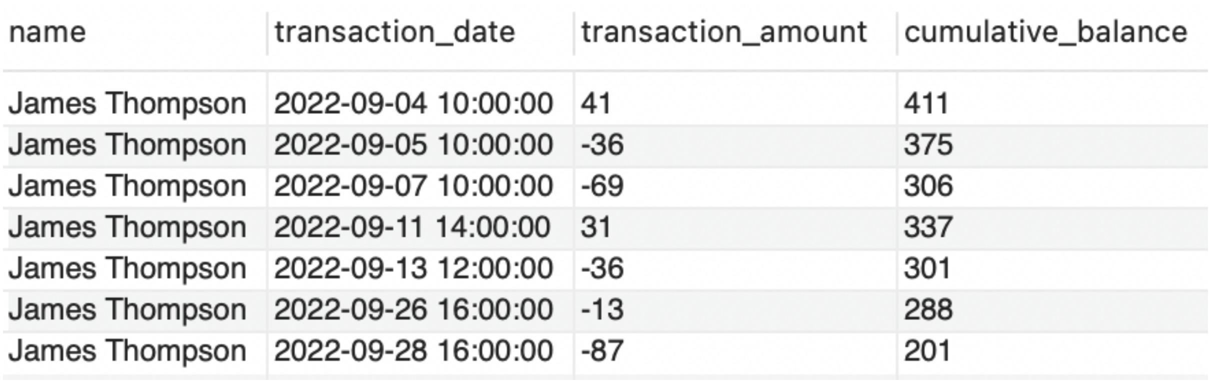
```sql
with payers as (
select payer_id, sum(amount) as amount_paid
from fact_transactions
group by payer_id),

receivers as (
select receiver_id, sum(amount) as amount_received
from fact_transactions
group by receiver_id)

select count(*) as number_of_balances
from dim_accounts
left join payers
on payers.payer_id = dim_accounts.account_id
left join receivers
on receivers.receiver_id = dim_accounts.account_id
where balance + coalesce(amount_paid,0) - coalesce(amount_received,0) >
1000;
```



```
number_of_balances
64
```

- Find the cumulative balance over transactions of the account owned by 'James Thompson'. Output the transaction date and cumulative balance

```sql
with transaction_history as
(select 'James Thompson' as name, transaction_date,
case when dm1.name = 'James Thompson'  then -amount
     when dm2.name = 'James Thompson'  then amount
     end transaction_amount,
case when dm1.name = 'James Thompson'  then dm1.balance
     when dm2.name = 'James Thompson'  then dm2.balance
     end initial_balance
from fact_transactions ft
left join dim_accounts dm1
on ft.payer_id = dm1.account_id
left join dim_accounts dm2
on ft.receiver_id = dm2.account_id
where dm1.name = 'James Thompson' or dm2.name = 'James Thompson')

select name, transaction_date,
transaction_amount,
sum(transaction_amount) over(order by transaction_date) +
initial_balance as cumulative_balance
from transaction_history;
```



name	transaction_date	transaction_amount	cumulative_balance
James Thompson	2022-09-04 10:00:00	41	411
James Thompson	2022-09-05 10:00:00	-36	375
James Thompson	2022-09-07 10:00:00	-69	306
James Thompson	2022-09-11 14:00:00	31	337
James Thompson	2022-09-13 12:00:00	-36	301
James Thompson	2022-09-26 16:00:00	-13	288
James Thompson	2022-09-28 16:00:00	-87	201