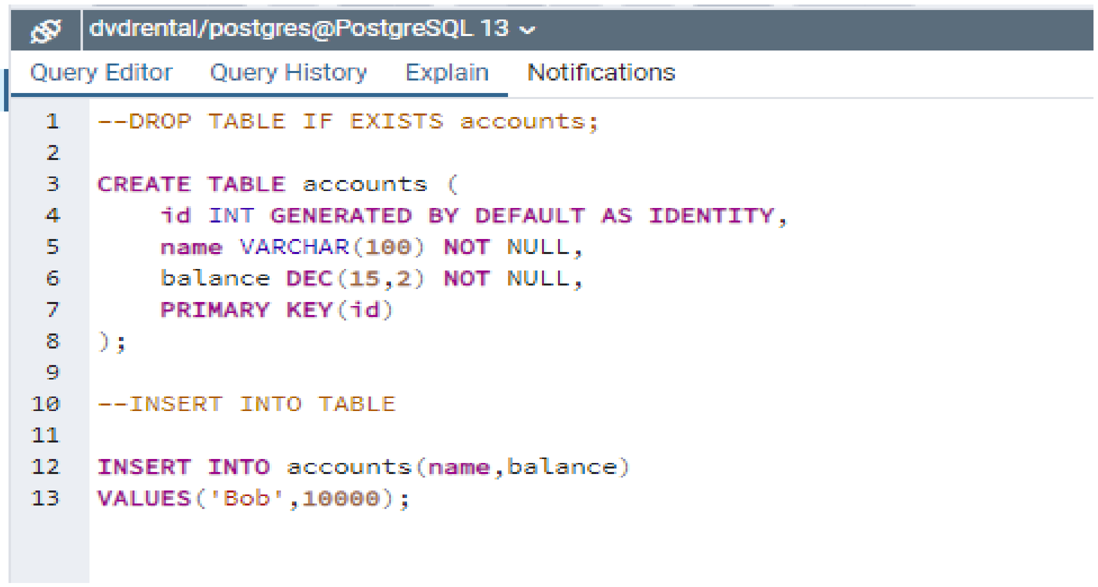
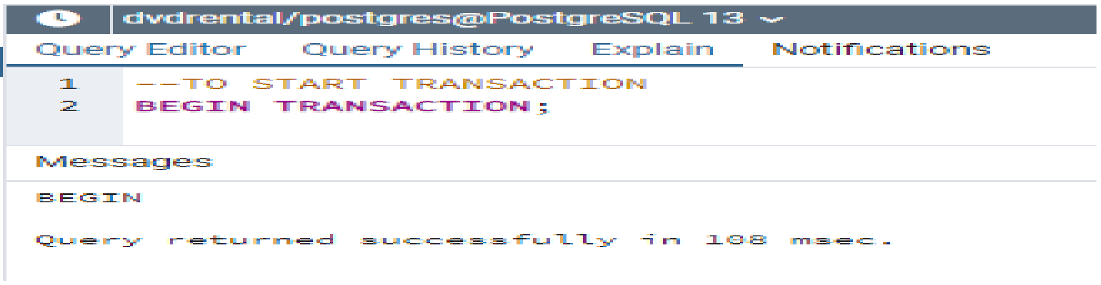
POSTGRESQL TRANSACTION

Setting up a sample table

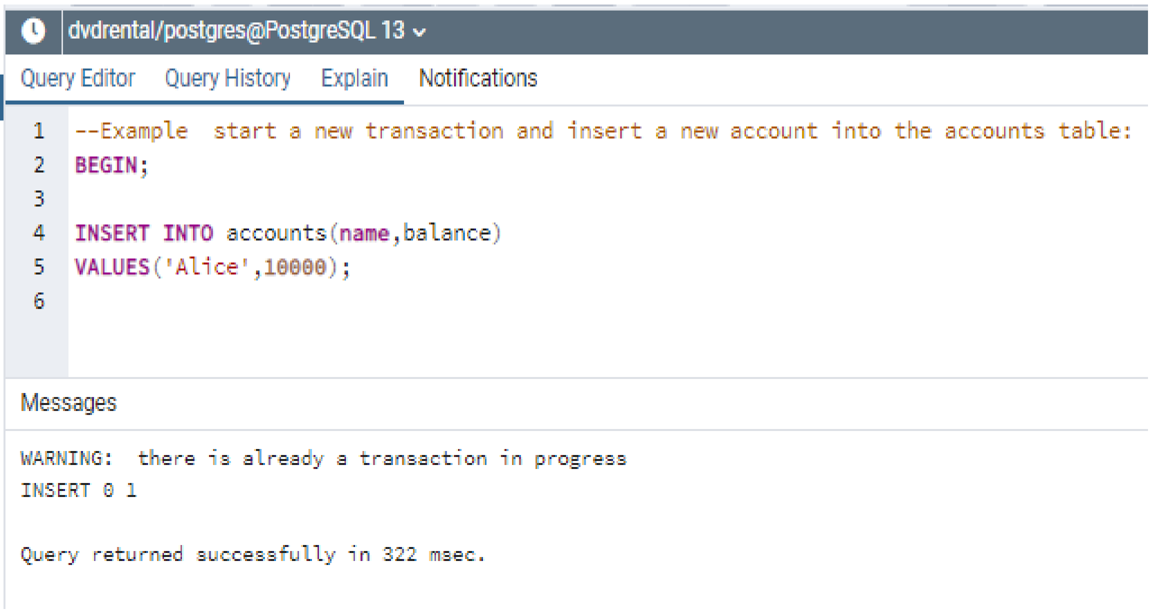


```
--DROP TABLE IF EXISTS accounts;

CREATE TABLE accounts (
    id INT GENERATED BY DEFAULT AS IDENTITY,
    name VARCHAR(100) NOT NULL,
    balance DEC(15,2) NOT NULL,
    PRIMARY KEY(id)
);

--INSERT INTO TABLE

INSERT INTO accounts(name,balance)
VALUES('Bob',10000);
```



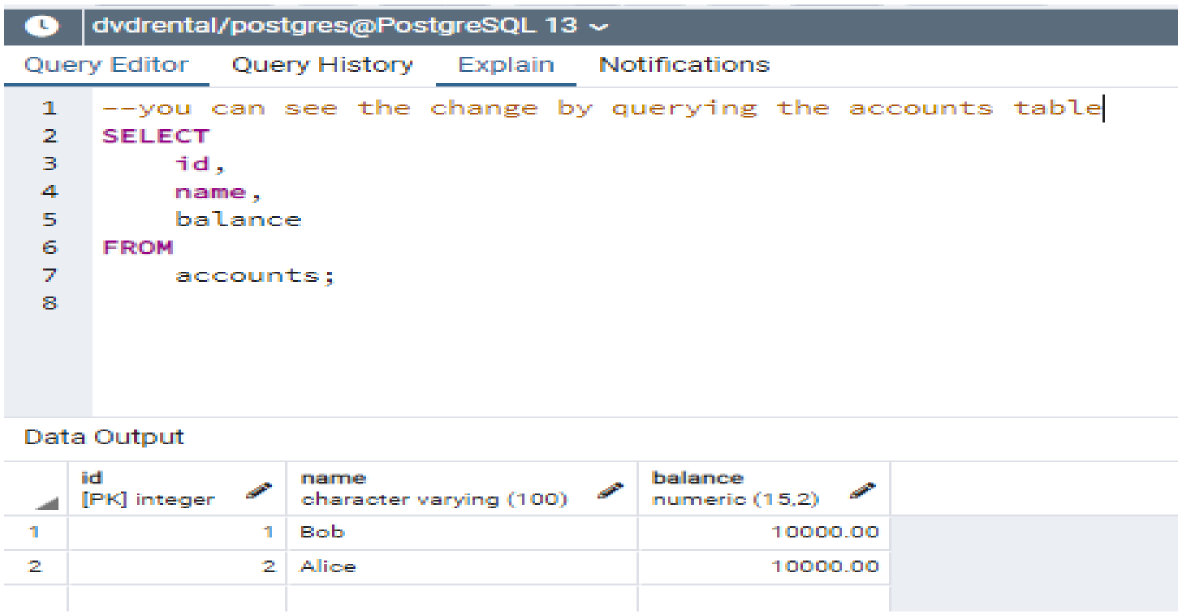
```
--TO START TRANSACTION
BEGIN TRANSACTION;
```

```
BEGIN

Query returned successfully in 108 msec.
```



```
--Example  start a new transaction and insert a new account into the accounts table:
BEGIN;

INSERT INTO accounts(name,balance)
VALUES('Alice',10000);

```

```
WARNING:  there is already a transaction in progress
INSERT 0 1

Query returned successfully in 322 msec.
```


dvdrental/postgres@PostgreSQL 13

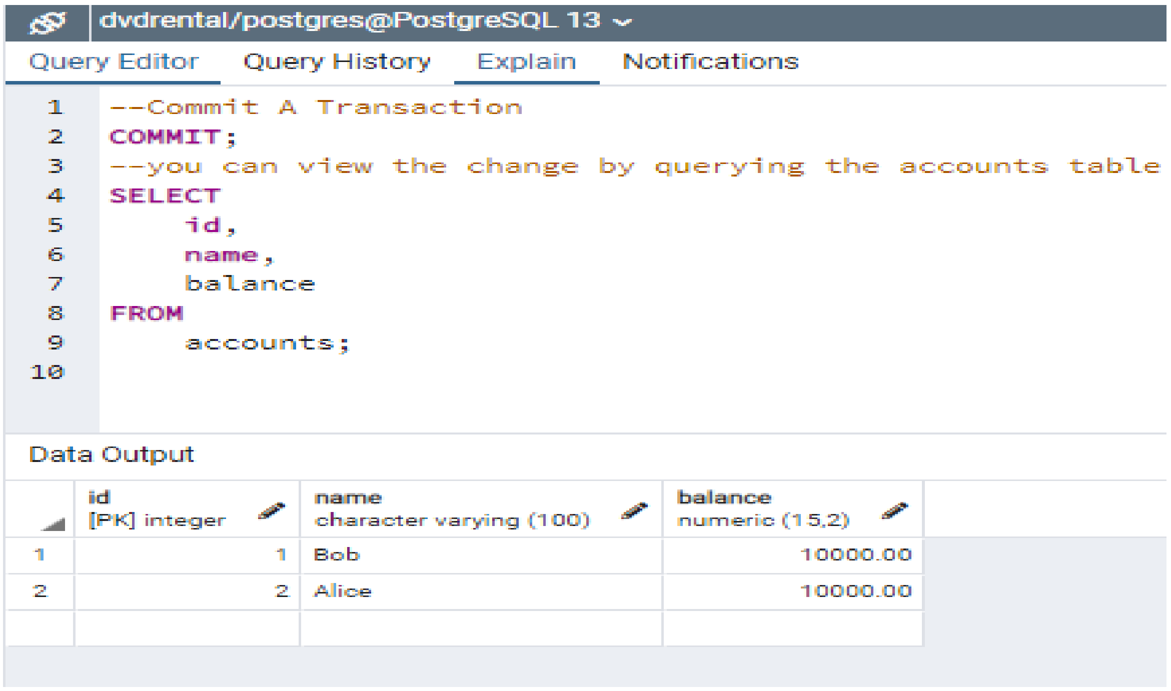
Query Editor | Query History | Explain | Notifications

```
--you can see the change by querying the accounts table
SELECT
     id,
     name,
     balance
FROM
     accounts;
```

Data Output

	id [PK] integer	name character varying (100)	balance numeric (15,2)
1	1	Bob	10000.00
2	2	Alice	10000.00

Commit A Transaction

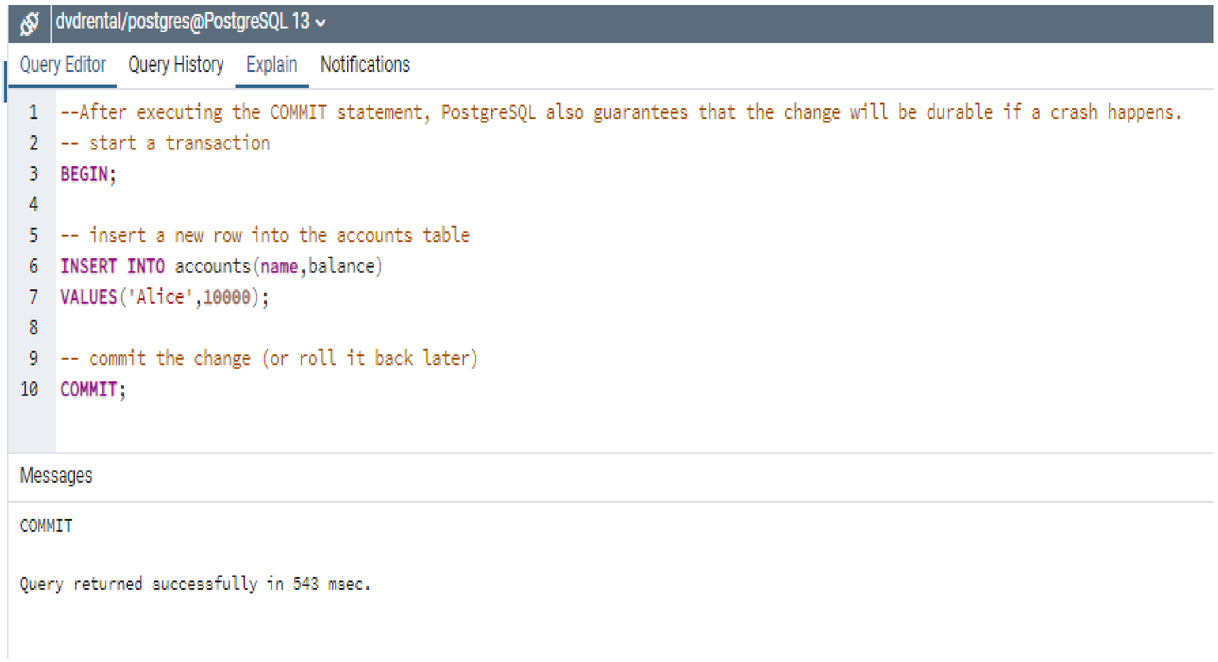

dvdrental/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications

```
--Commit A Transaction
COMMIT;
--you can view the change by querying the accounts table
SELECT
     id,
     name,
     balance
FROM
     accounts;
```

Data Output

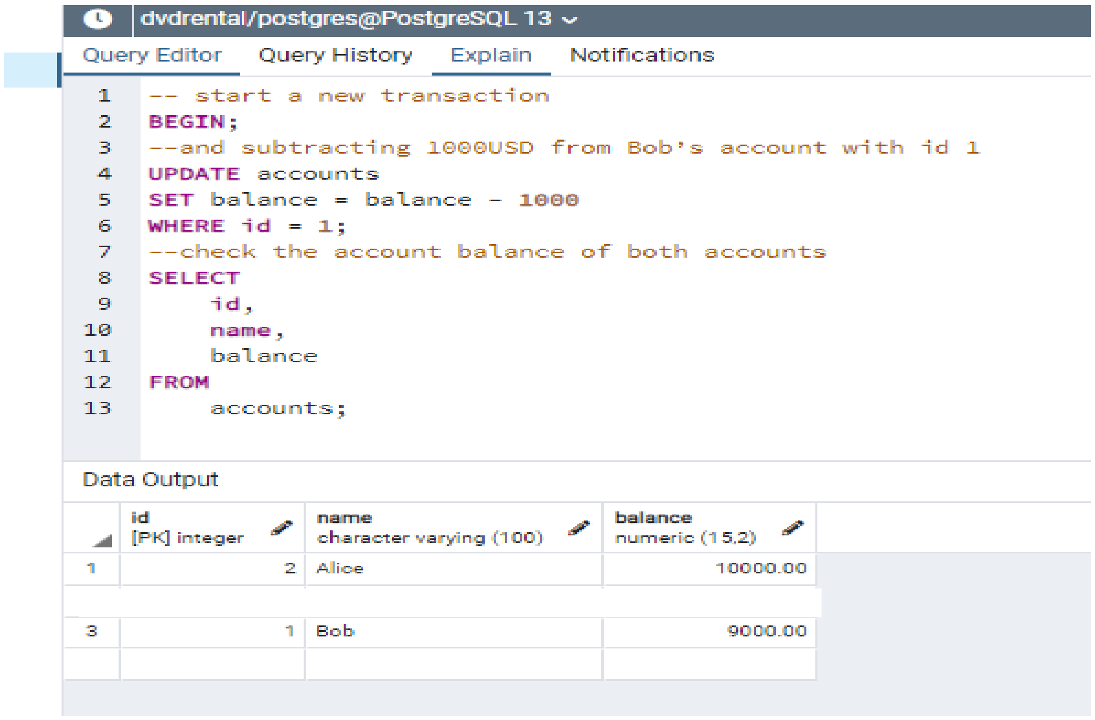
	id [PK] integer	name character varying (100)	balance numeric (15,2)
1	1	Bob	10000.00
2	2	Alice	10000.00



```
--After executing the COMMIT statement, PostgreSQL also guarantees that the change will be durable if a crash happens.
-- start a transaction
BEGIN;

-- insert a new row into the accounts table
INSERT INTO accounts(name,balance)
VALUES('Alice',10000);

-- commit the change (or roll it back later)
COMMIT;
```

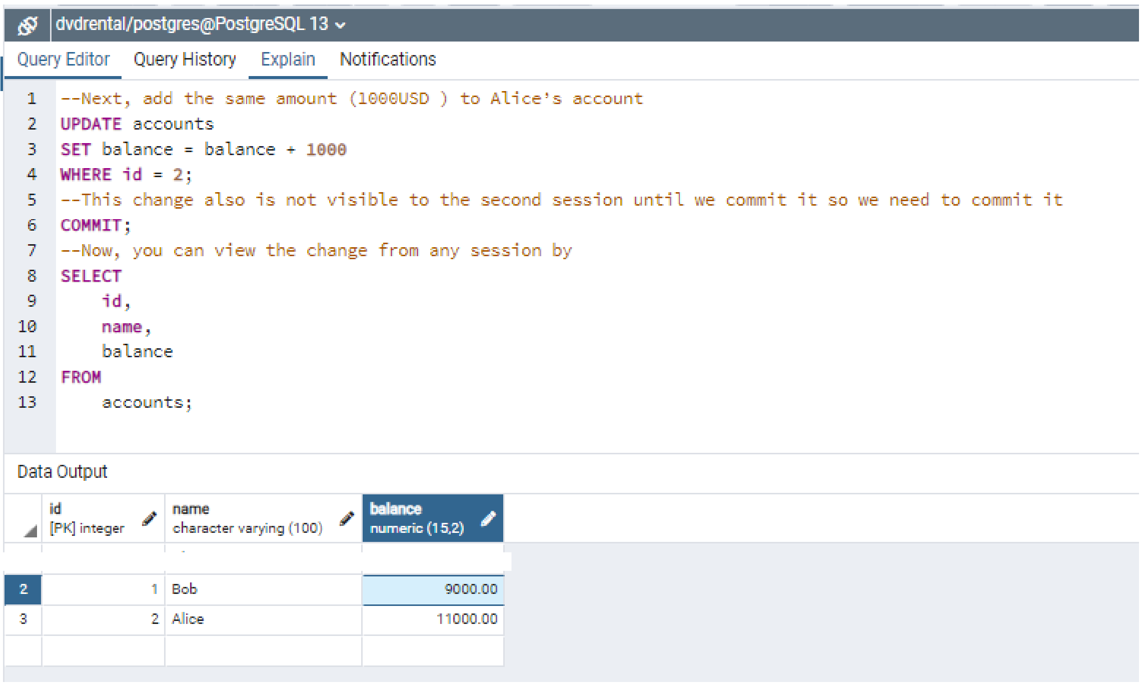
```
COMMIT

Query returned successfully in 543 msec.
```

PostgreSQL COMMIT: Bank account transfer



```
-- start a new transaction
BEGIN;
--and subtracting 1000USD from Bob's account with id 1
UPDATE accounts
SET balance = balance - 1000
WHERE id = 1;
--check the account balance of both accounts
SELECT
    id,
    name,
    balance
FROM
    accounts;
```

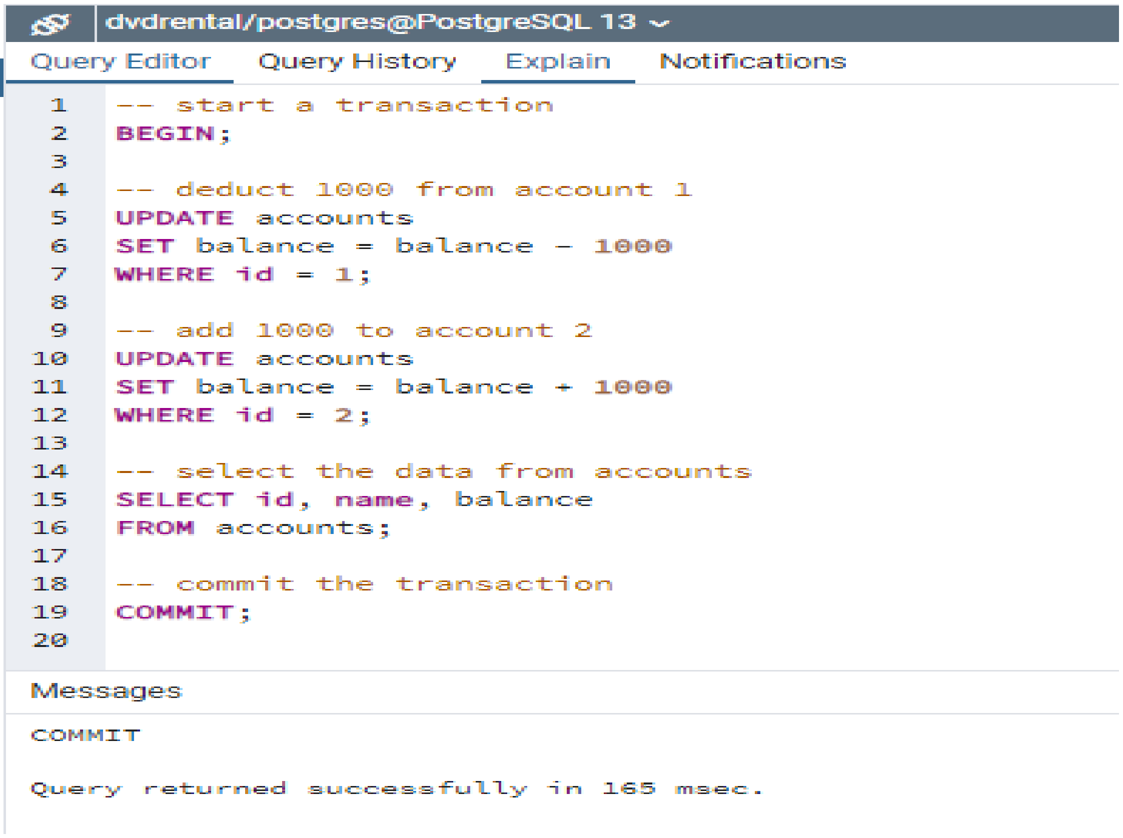
	id [PK] integer	name character varying (100)	balance numeric (15,2)
1	2	Alice	10000.00
3	1	Bob	9000.00



```sql
--Next, add the same amount (1000USD ) to Alice's account
UPDATE accounts
SET balance = balance + 1000
WHERE id = 2;
--This change also is not visible to the second session until we commit it so we need to commit it
COMMIT;
--Now, you can view the change from any session by
SELECT
    id,
    name,
    balance
FROM
    accounts;
```

	id [PK] integer	name character varying (100)	balance numeric (15,2)
2	1	Bob	9000.00
3	2	Alice	11000.00

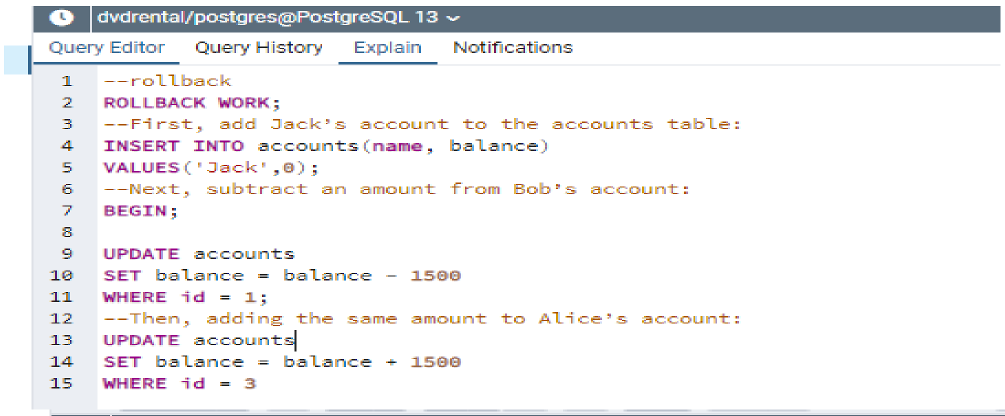
Put it all together



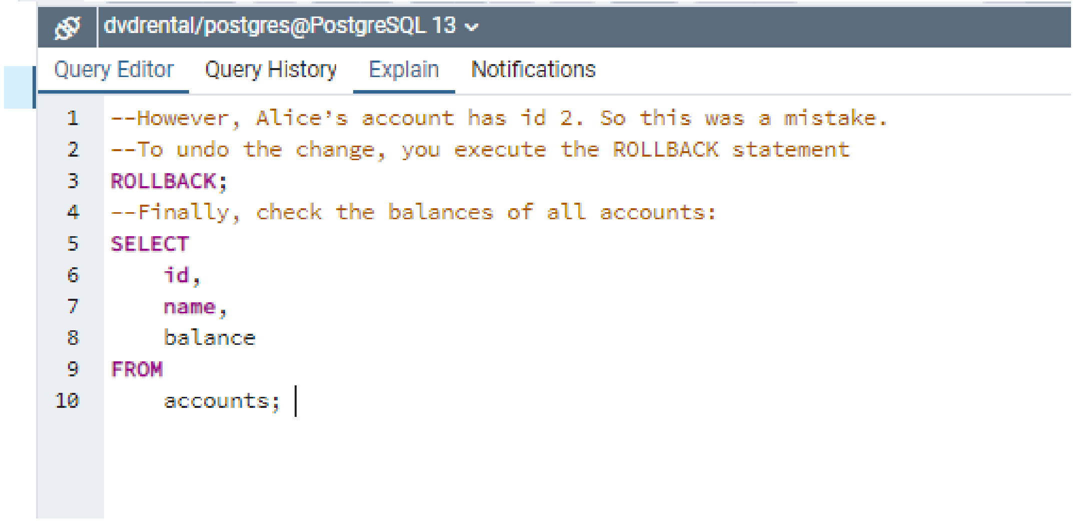
```sql
-- start a transaction
BEGIN;

-- deduct 1000 from account 1
UPDATE accounts
SET balance = balance - 1000
WHERE id = 1;

-- add 1000 to account 2
UPDATE accounts
SET balance = balance + 1000
WHERE id = 2;

-- select the data from accounts
SELECT id, name, balance
FROM accounts;

-- commit the transaction
COMMIT;
```

```
COMMIT

Query returned successfully in 165 msec.
```

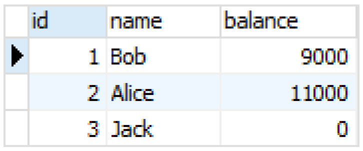
Rolling back a transaction



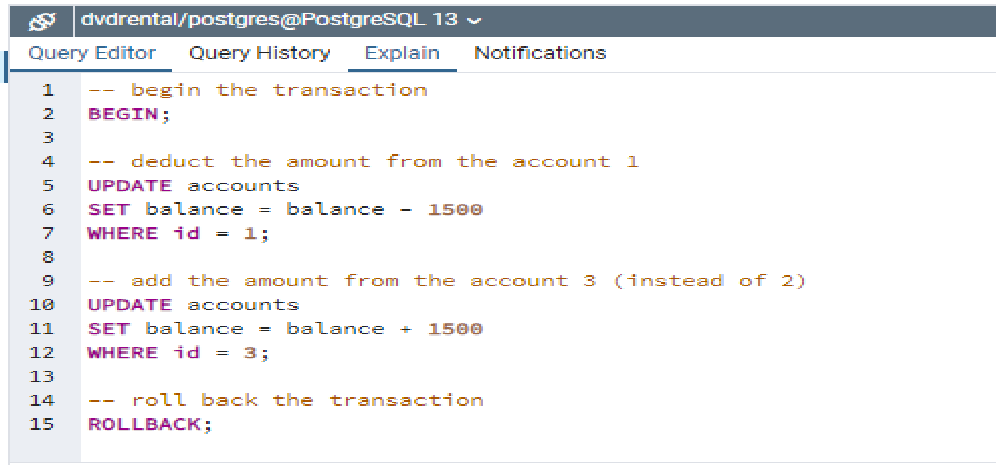
```
--rollback
ROLLBACK WORK;
--First, add Jack's account to the accounts table:
INSERT INTO accounts(name, balance)
VALUES('Jack',0);
--Next, subtract an amount from Bob's account:
BEGIN;

UPDATE accounts
SET balance = balance - 1500
WHERE id = 1;
--Then, adding the same amount to Alice's account:
UPDATE accounts
SET balance = balance + 1500
WHERE id = 3
```



```
--However, Alice's account has id 2. So this was a mistake.
--To undo the change, you execute the ROLLBACK statement
ROLLBACK;
--Finally, check the balances of all accounts:
SELECT
    id,
    name,
    balance
FROM
    accounts;
```



id	name	balance
1	Bob	9000
2	Alice	11000
3	Jack	0



```
-- begin the transaction
BEGIN;

-- deduct the amount from the account 1
UPDATE accounts
SET balance = balance - 1500
WHERE id = 1;

-- add the amount from the account 3 (instead of 2)
UPDATE accounts
SET balance = balance + 1500
WHERE id = 3;

-- roll back the transaction
ROLLBACK;
```