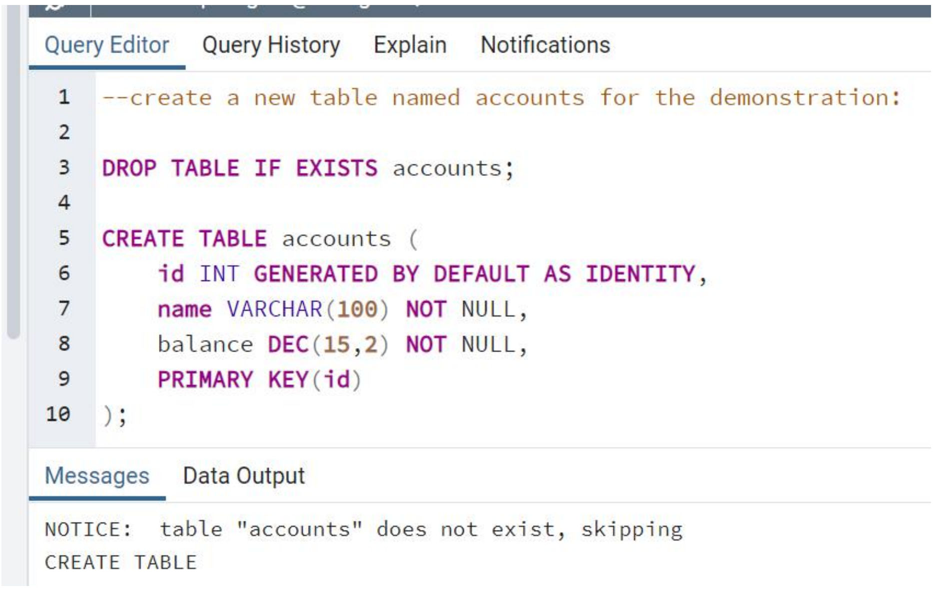
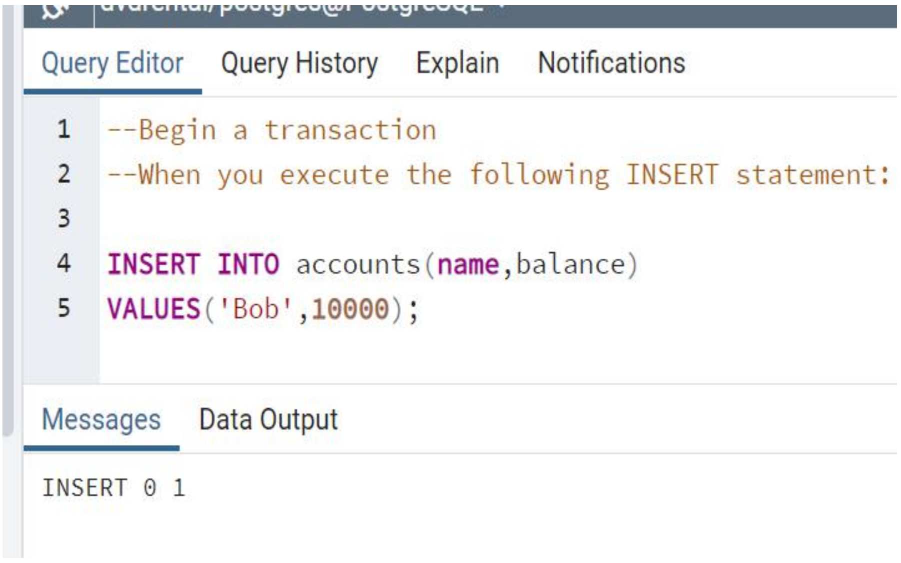
PostgreSQL Transaction



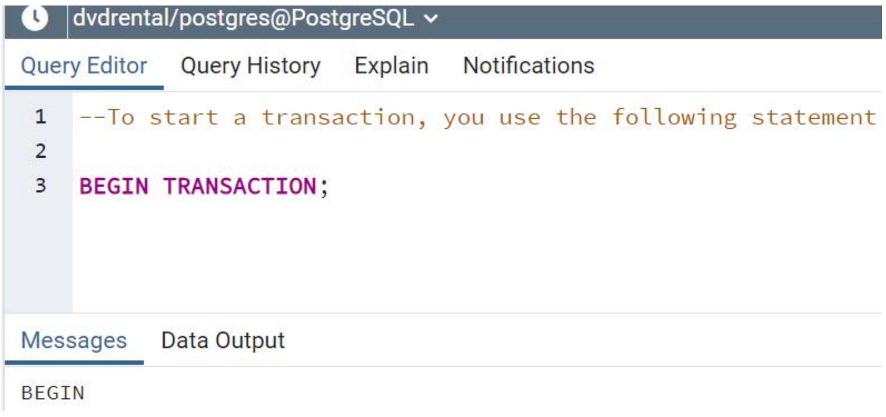
```sql
--create a new table named accounts for the demonstration:

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)
);
```

```
NOTICE:  table "accounts" does not exist, skipping
CREATE TABLE
```



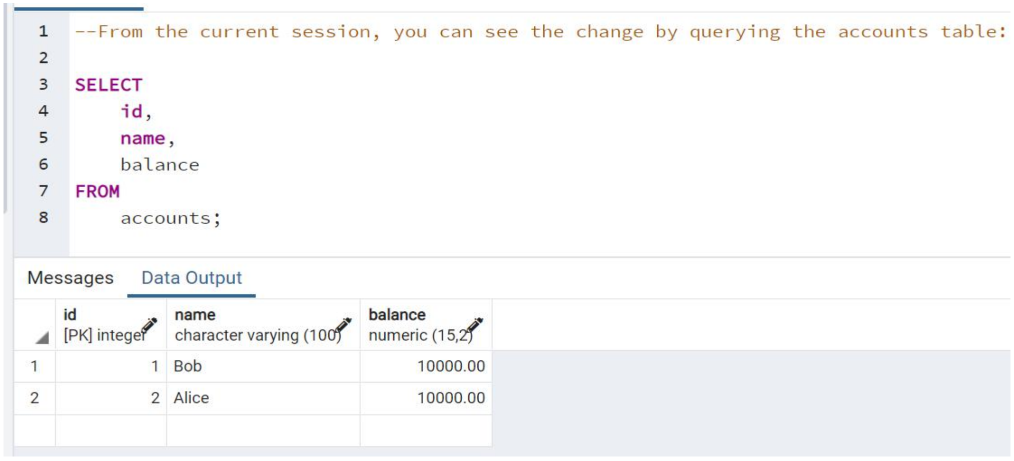
```sql
--Begin a transaction
--When you execute the following INSERT statement:

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

```
INSERT 0 1
```

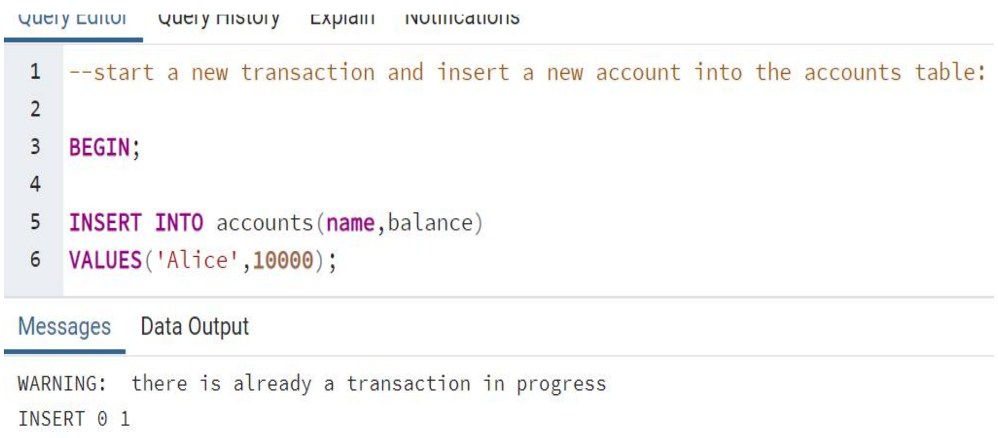


```sql
--To start a transaction, you use the following statement

BEGIN TRANSACTION;
```

```
BEGIN
```

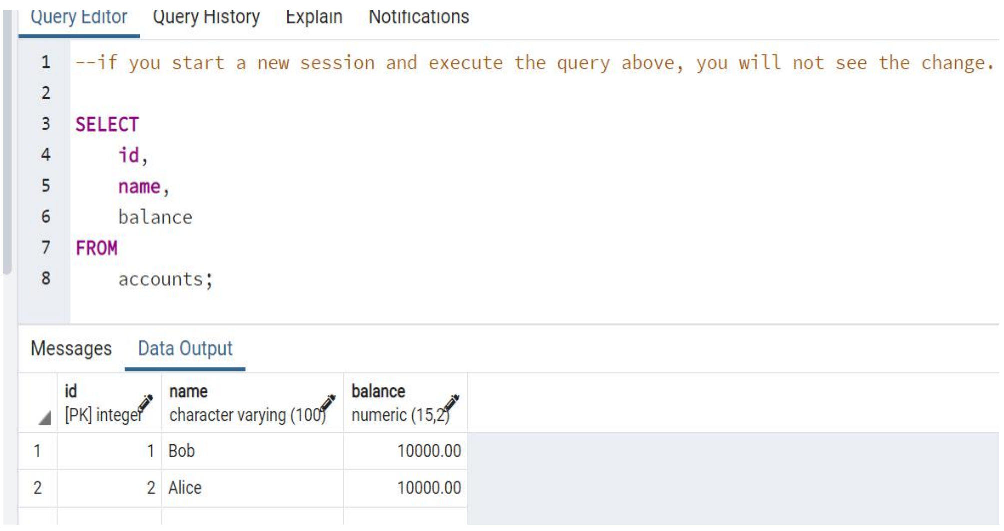

```
--From the current session, you can see the change by querying the accounts table:

SELECT
    id,
    name,
    balance
FROM
    accounts;
```

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



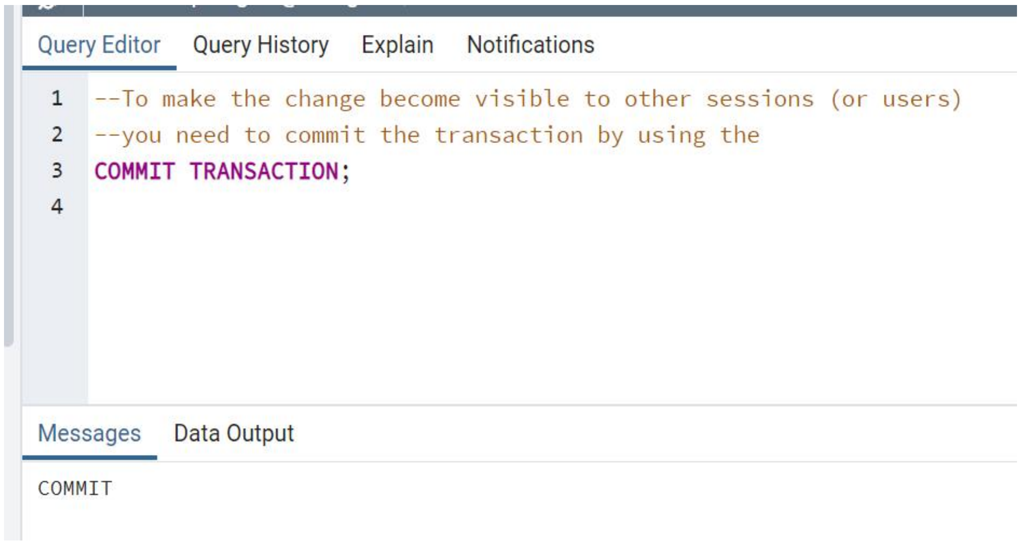
```
--start a new transaction and insert a new account into the accounts table:

BEGIN;

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

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



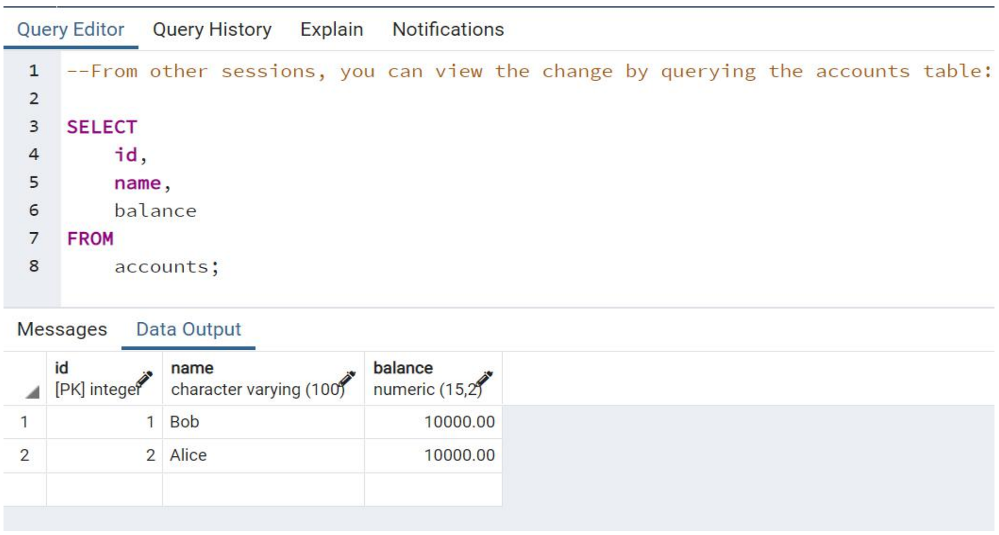
```
--if you start a new session and execute the query above, you will not see the change.

SELECT
    id,
    name,
    balance
FROM
    accounts;
```

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



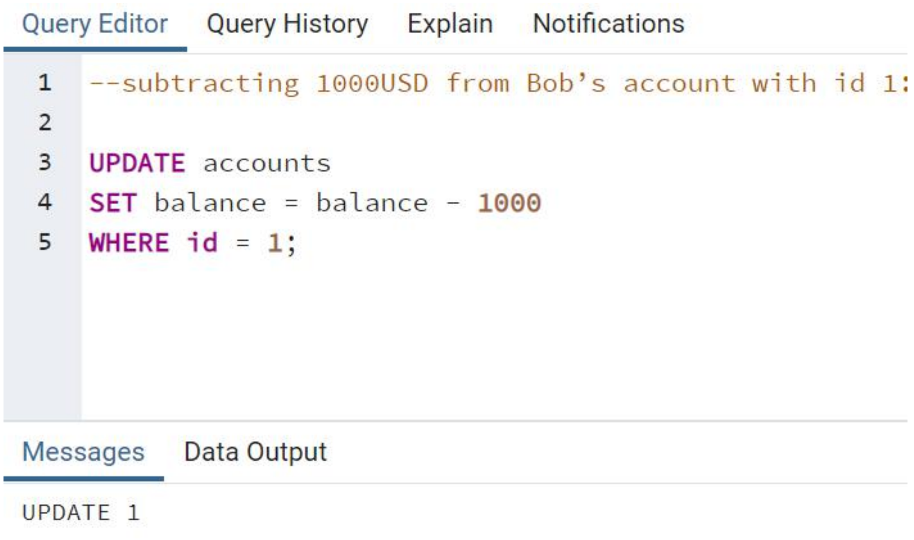
```
--To make the change become visible to other sessions (or users)
--you need to commit the transaction by using the
COMMIT TRANSACTION;

```

```
COMMIT
```



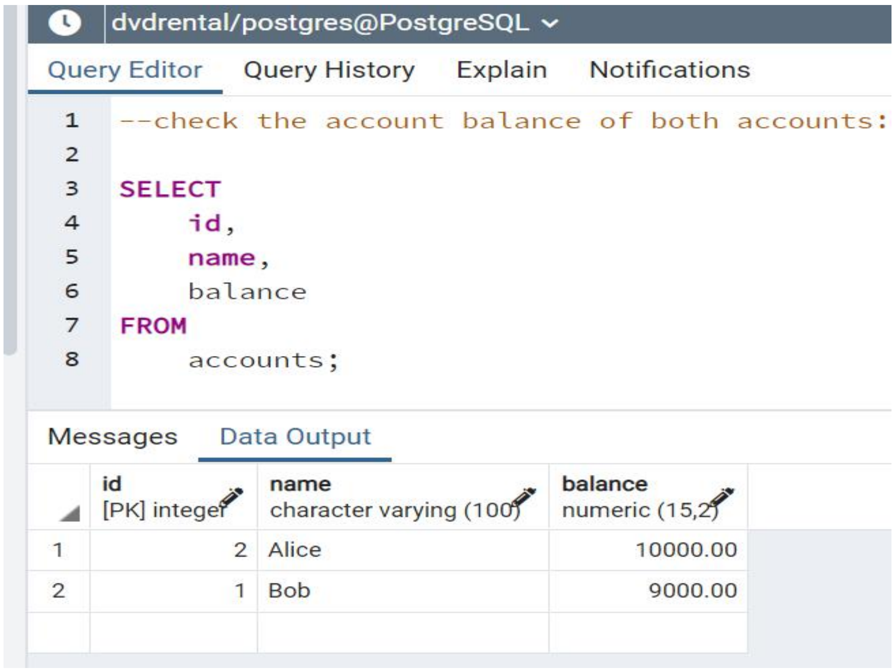
```
--From other sessions, you can view the change by querying the accounts table:

SELECT
    id,
    name,
    balance
FROM
    accounts;
```

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



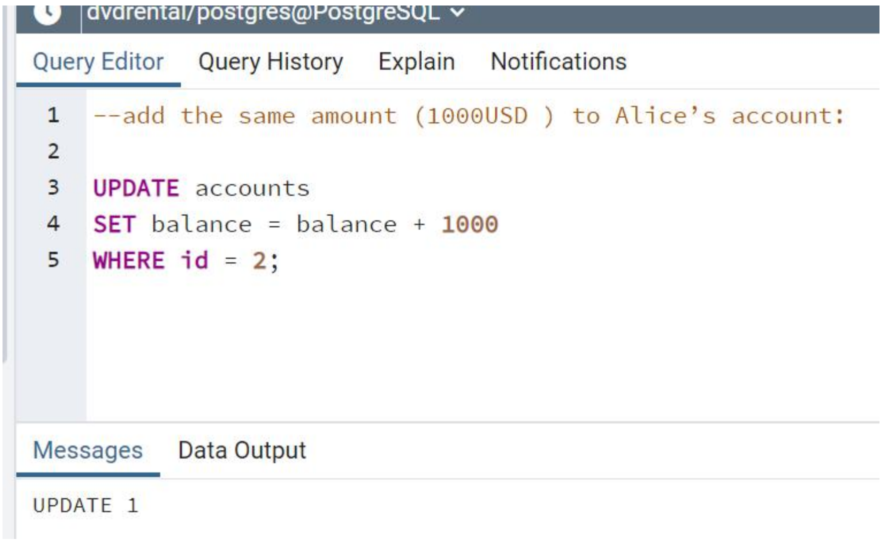
```
--subtracting 1000USD from Bob's account with id 1:

UPDATE accounts
SET balance = balance - 1000
WHERE id = 1;
```

```
UPDATE 1
```



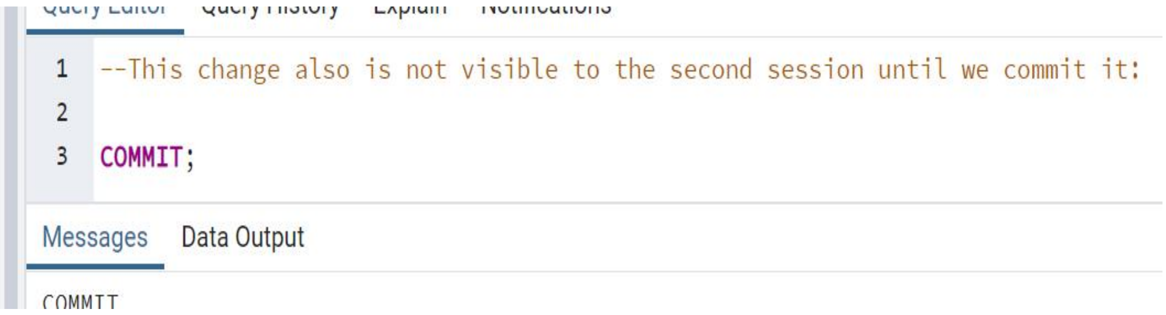
```sql
--check the account balance of both accounts:

SELECT
    id,
    name,
    balance
FROM
    accounts;
```

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



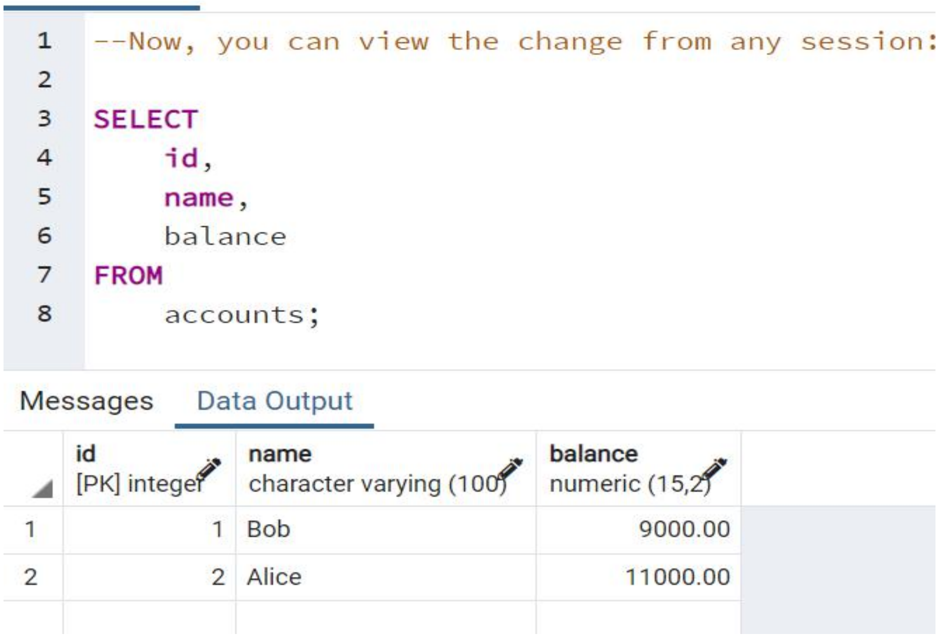
```sql
--add the same amount (1000USD ) to Alice’s account:

UPDATE accounts
SET balance = balance + 1000
WHERE id = 2;
```

```
UPDATE 1
```



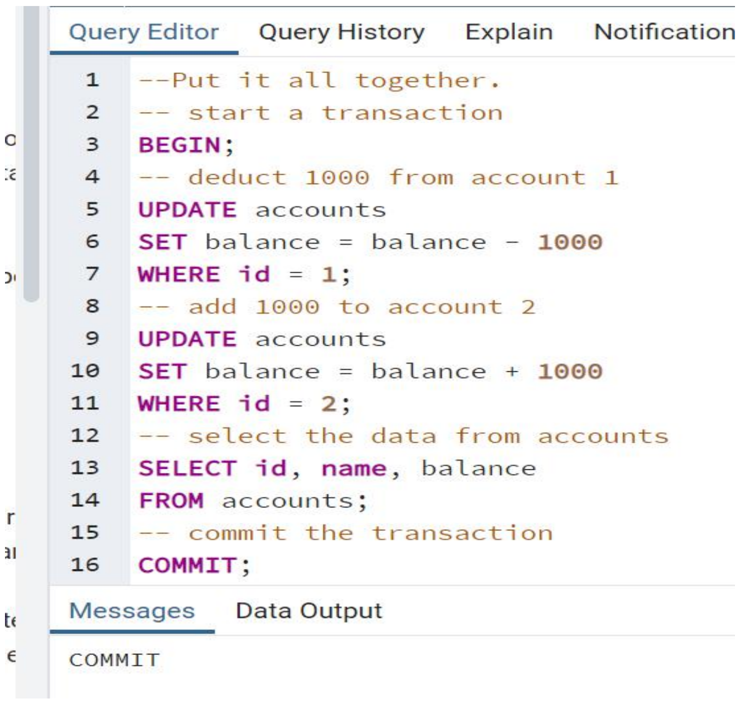
```sql
--This change also is not visible to the second session until we commit it:

COMMIT;
```

```
COMMIT
```



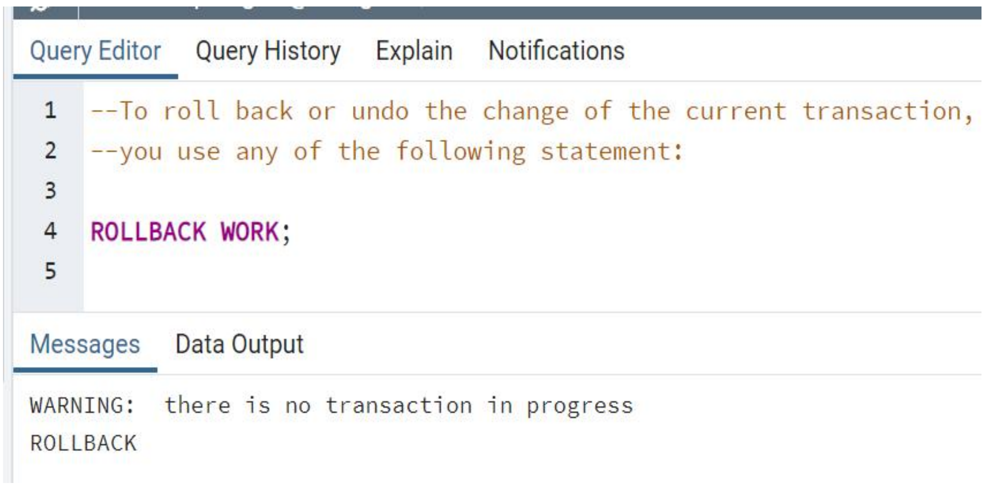
```
--Now, you can view the change from any session:

SELECT
    id,
    name,
    balance
FROM
    accounts;
```

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



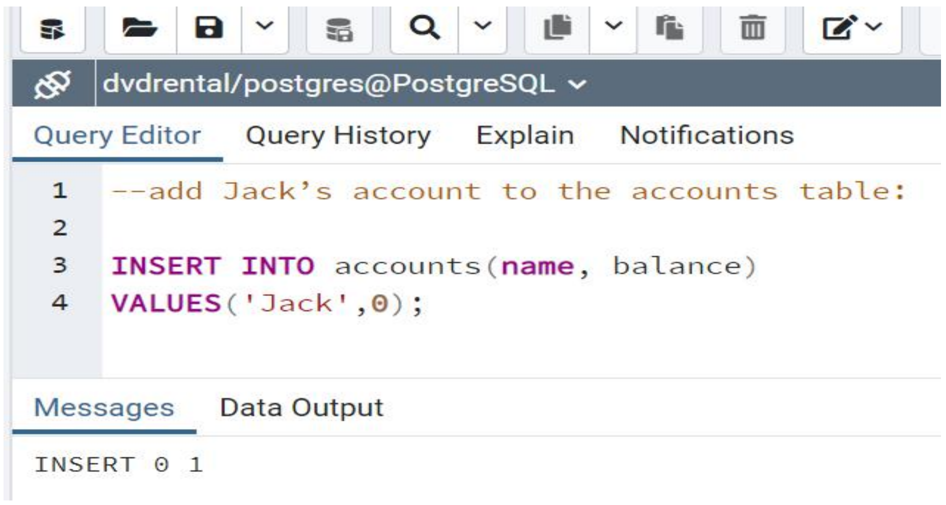
```
--Put it all together.
-- start a transaction
BEGIN;
-- deduct 1000 from account 1
UPDATE accounts
SET balance = balance - 1000
WHERE id = 1;
-- add 1000 to account 2
UPDATE accounts
SET balance = balance + 1000
WHERE id = 2;
-- select the data from accounts
SELECT id, name, balance
FROM accounts;
-- commit the transaction
COMMIT;
```

```
COMMIT
```



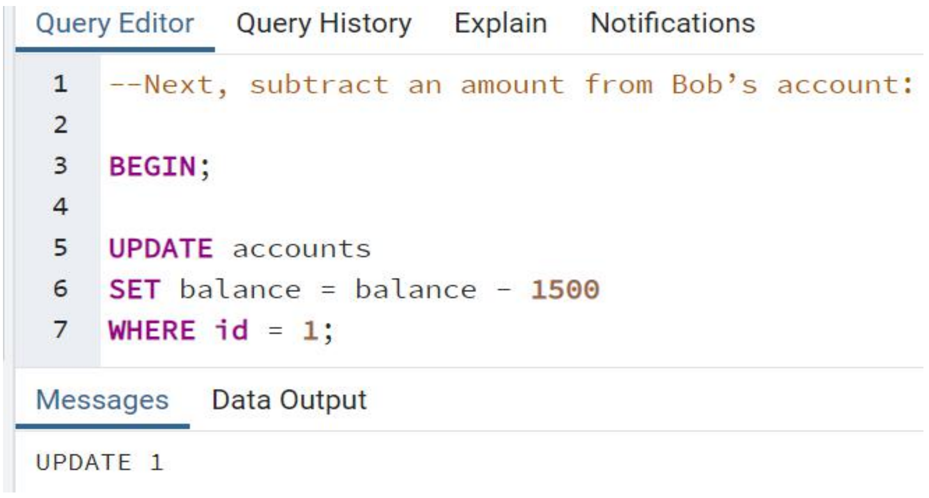
```
--To roll back or undo the change of the current transaction,
--you use any of the following statement:

ROLLBACK WORK;

```

```
WARNING:  there is no transaction in progress
ROLLBACK
```



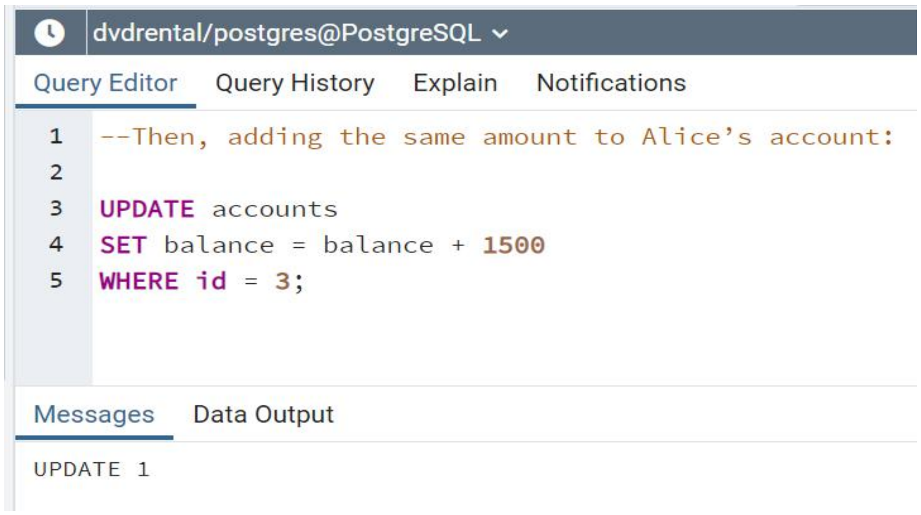
```
--add Jack's account to the accounts table:

INSERT INTO accounts(name, balance)
VALUES('Jack',0);
```

```
INSERT 0 1
```



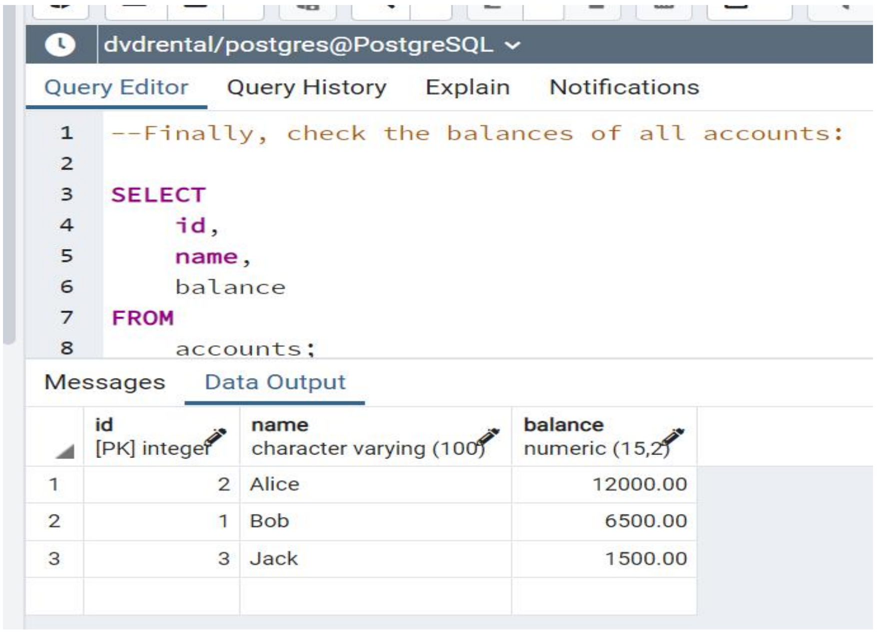
```
--Next, subtract an amount from Bob's account:

BEGIN;

UPDATE accounts
SET balance = balance - 1500
WHERE id = 1;
```

```
UPDATE 1
```



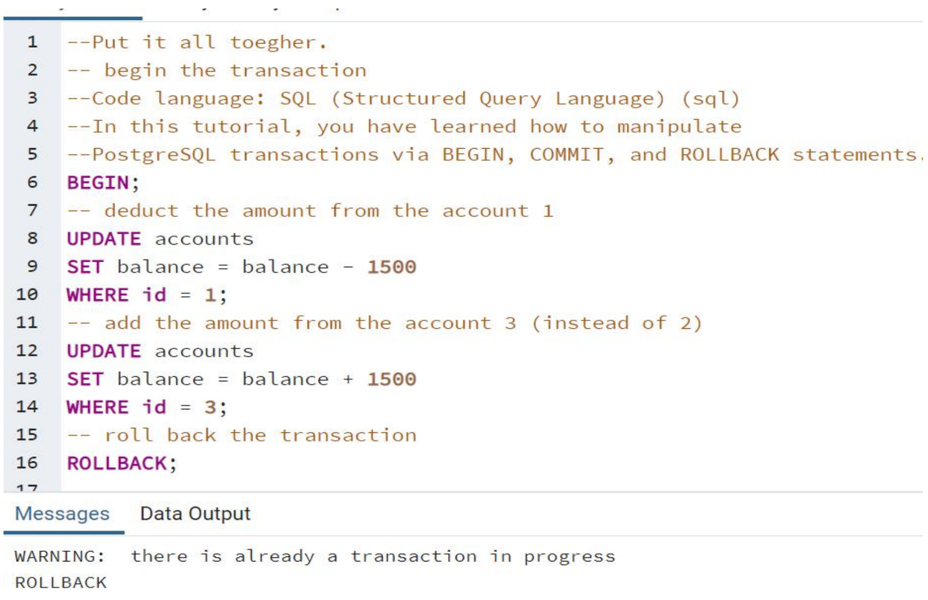
```sql
--Then, adding the same amount to Alice's account:

UPDATE accounts
SET balance = balance + 1500
WHERE id = 3;
```

```
UPDATE 1
```



```sql
--Finally, check the balances of all accounts:

SELECT
     id,
     name,
     balance
FROM
     accounts;
```

	id [PK] integer	name character varying (100)	balance numeric (15,2)
1	2	Alice	12000.00
2	1	Bob	6500.00
3	3	Jack	1500.00



```sql
--Put it all toegher.
-- begin the transaction
--Code language: SQL (Structured Query Language) (sql)
--In this tutorial, you have learned how to manipulate
--PostgreSQL transactions via BEGIN, COMMIT, and ROLLBACK statements.
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;
```

```
WARNING:  there is already a transaction in progress
ROLLBACK
```