The most common example is a bank transfer: imagine Abby is trying to send $20 to Barbara, and we want to record this fact in a database. So we deduct $20 from Abby's account, but before we can increase Barbara's balance by $20, there's a power failure. In this case, the whole transaction should be cancelled. Otherwise, Abby would be out $20!

In PostgreSQL, we can begin a transaction with `BEGIN TRANSACTION`. Inside of our transaction, any SQL we write won't make permanent changes to the database. If we make a change we don't like, we can cancel the transaction with the `ROLLBACK` command.

But more importantly for our present purposes, if there's a change we do like, we need to `COMMIT` the transaction.

Here's a small example you can explore in the demo code:

demo/colors.sql

```
-- from the terminal run:
-- psql < colors.sql

DROP DATABASE IF EXISTS colors;

CREATE DATABASE colors;

\c colors

CREATE TABLE colors
(
  id SERIAL PRIMARY KEY,
  name TEXT
);

INSERT INTO colors (name) VALUES ('red'), ('blue'), ('green');

BEGIN TRANSACTION;
  DELETE FROM colors;
  SELECT * FROM colors;
  -- no colors are left!
ROLLBACK;

SELECT * FROM colors;
-- all the colors are still here!
-- we only removed them in a rolled back transaction.

BEGIN TRANSACTION;
  DELETE FROM colors;
  SELECT * FROM colors;
  -- no colors are left!
COMMIT;

SELECT * FROM colors;
-- Since we committed the transaction,
```