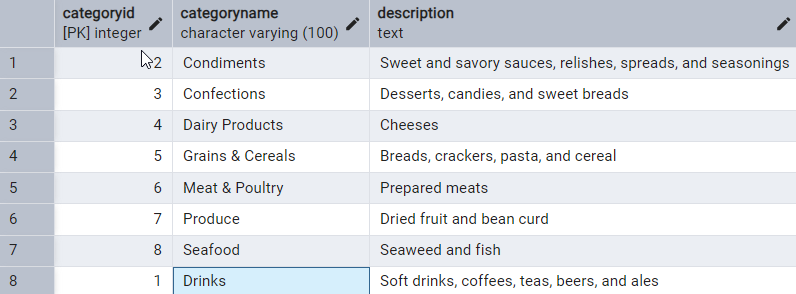
**Elamathi\_DAY3 \_Assignment3**

**1: Update the categoryName From “Beverages” to "Drinks" in the categories table.**

UPDATE categories SET categoryname = 'Drinks' WHERE categoryname = 'Beverages';

SELECT \* FROM categories

**OUTPUT:**



**------------------------------------------------------------------------------------------------------------------------------------------**

**2: Insert into shipper new record (give any values) Delete that new record from shippers table.**

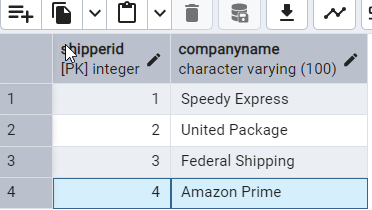
**2.1. Insert new record into shipper**

INSERT INTO shippers (shipperid, companyname)

VALUES (4, 'Amazon Prime');

SELECT \* FROM shippers

**OUTPUT:**

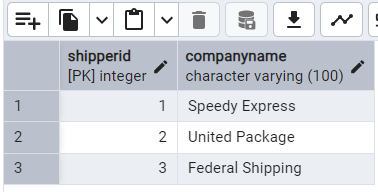


**2.2. Delete that new record from shippers table**

DELETE FROM shippers WHERE shipperid = 4;

**OUTPUT:**

**4th record got deleted**



**------------------------------------------------------------------------------------------------------------------------------------------**

**3: Cascade**

**3.1. Alter the foreign key on products (categoryID) to add ON UPDATE CASCADE, ON DELETE CASCADE**

**-- First delete the existing foreign key**

ALTER TABLE products DROP CONSTRAINT products\_categoryid\_fkey;

**-- Next add the foreign key with ON UPDATE and ON DELETE CASCADE**

ALTER TABLE products

ADD CONSTRAINT fk\_products\_categoryID

FOREIGN KEY (categoryID)

REFERENCES categories (categoryID)

ON UPDATE CASCADE

ON DELETE CASCADE;

**3.2 Update categoryID=1 to categoryID=1001. Make sure related products update their categoryID**

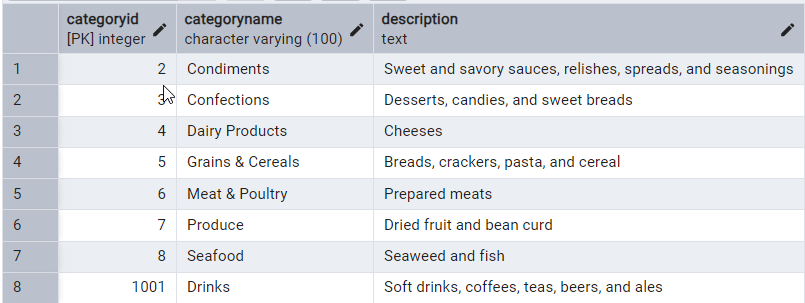
UPDATE categories SET categoryid = 1001 WHERE categoryid = 1;

SELECT \* FROM categories

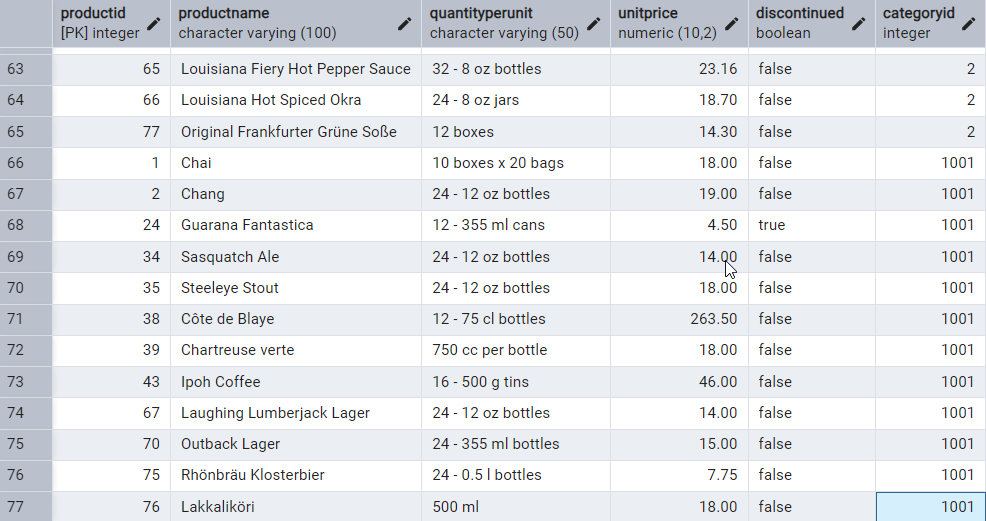
SELECT \* FROM products

**OUTPUT:**

**Categories Table**



**Product Table**



**3.3: Delete the categoryID= “3” from categories. Verify products table too.**

**--since categoryid --> product table and productid --> order details table set CASCADE in order table**

**-- First Drop existing foreign key**

ALTER TABLE order\_details

DROP CONSTRAINT order\_details\_productid\_fkey;

**-- Next add foreign key with CASCADE**

ALTER TABLE order\_details

ADD CONSTRAINT order\_details\_productid\_fkey

FOREIGN KEY (productid)

REFERENCES products (productid)

ON UPDATE CASCADE

ON DELETE CASCADE;

**-- Then delete the record from categories table and verify products table too**

DELETE FROM categories WHERE categoryid = 3;

SELECT \* FROM categories WHERE categoryid = 3;

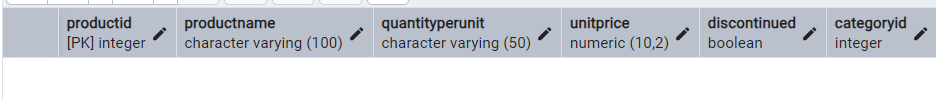
SELECT \* FROM products WHERE categoryid = 3;

**OUTPUT:**

Record is deleted in categories table

https://i.imgur.com/KvapRtO.png

Upon cascading record with categoryid =3 is deleted in products table as well



**------------------------------------------------------------------------------------------------------------------------------------------**

**4: Delete the customer = “VINET” from customers. Corresponding customers in orders table should be set to null**

**-- First drop the existing foreign key constraint**

ALTER TABLE orders

DROP CONSTRAINT orders\_customerid\_fkey;

**-- Next set it with ON DELETE SET NULL**

ALTER TABLE orders

ADD CONSTRAINT orders\_customerid\_fkey

FOREIGN KEY (customerid)

REFERENCES customers (customerid)

ON DELETE SET NULL;

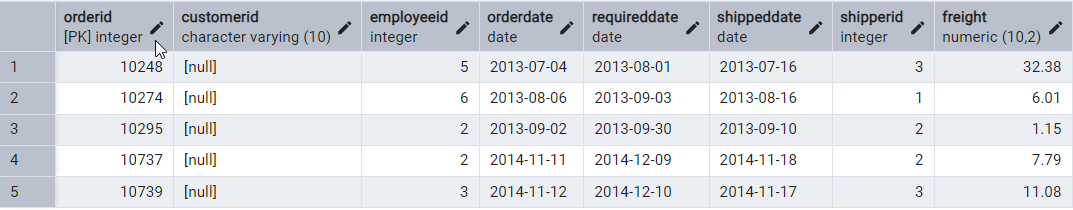
**-- Then delete the record with customerid = 'VINET' in customers table**

DELETE FROM customers WHERE customerid = 'VINET';

**-- Verify in orders table**

SELECT \* FROM orders WHERE customerid IS NULL;

**OUTPUT:**



**------------------------------------------------------------------------------------------------------------------------------------------**

**5. Insert the following data to Products using UPSERT:**

**--First record**

INSERT INTO products (productid, productname, quantityperunit, unitprice, discontinued, categoryid)

VALUES (100, 'Wheat bread', '10 boxes', 13, FALSE, 3)

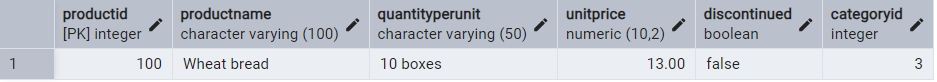
ON CONFLICT (productid)

DO UPDATE

SET quantityperunit = EXCLUDED.quantityperunit;

SELECT \* FROM products WHERE productid = 100;

**OUTPUT:**



**--Second record**

INSERT INTO products (productid, productname, quantityperunit, unitprice, discontinued, categoryid)

VALUES (101, 'White bread', '5 boxes', 13, FALSE, 3)

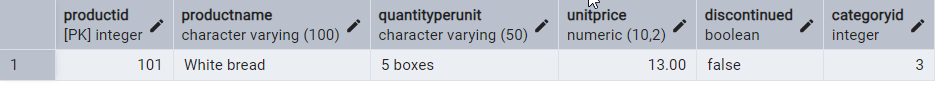
ON CONFLICT (productid)

DO UPDATE

SET quantityperunit = EXCLUDED.quantityperunit;

SELECT \* FROM products WHERE productid = 101;

**OUTPUT:**



**--Third record**

INSERT INTO products (productid, productname, quantityperunit, unitprice, discontinued, categoryid)

VALUES (100, 'Wheat bread', '10 boxes', 13, FALSE, 3)

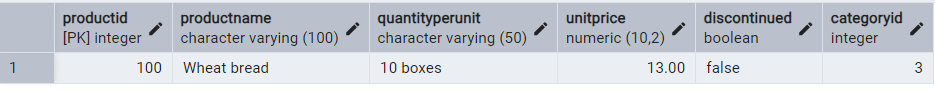
ON CONFLICT (productid)

DO UPDATE

SET quantityperunit = EXCLUDED.quantityperunit;

SELECT \* FROM products WHERE productid = 100;

**OUTPUT:**



**------------------------------------------------------------------------------------------------------------------------------------------**

**6. Write a MERGE query:**

**-- Step 1: Create Temp table**

CREATE TEMP TABLE updated\_products (

productid INT,

productname TEXT,

quantityperunit TEXT,

unitprice NUMERIC,

discontinued BOOLEAN,

categoryid INT

);

INSERT INTO updated\_products (productid, productname, quantityperunit, unitprice, discontinued, categoryid)

VALUES

(100, 'Wheat bread', '10', 20, 'TRUE', 3),

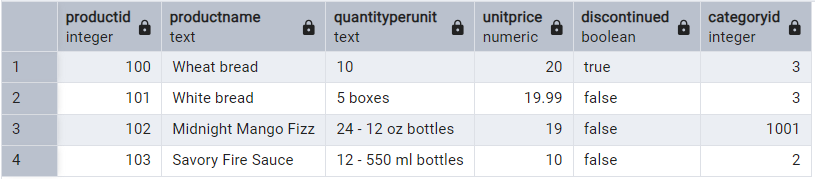
(101, 'White bread', '5 boxes', 19.99, 'FALSE', 3),

(102, 'Midnight Mango Fizz', '24 - 12 oz bottles', 19, 'FALSE', 1001),

(103, 'Savory Fire Sauce', '12 - 550 ml bottles', 10, 'FALSE', 2);

SELECT \* FROM updated\_products;

**OUTPUT:**



**-- Step 2:**

MERGE INTO products p

USING updated\_products u

ON p.productid = u.productid

WHEN MATCHED AND u.discontinued = 'FALSE' THEN

UPDATE SET

unitprice = u.unitprice,

discontinued = u.discontinued

WHEN MATCHED AND u.discontinued = 'FALSE' THEN

DELETE

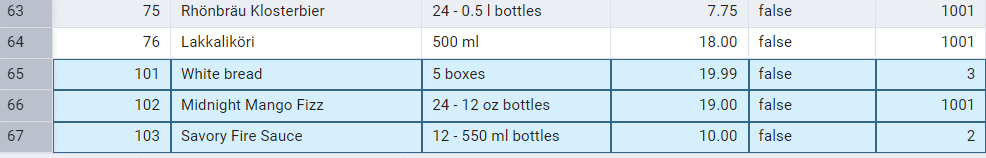
WHEN NOT MATCHED AND u.discontinued = 'TRUE' THEN

INSERT (productid, productname, quantityperunit, unitprice, discontinued, categoryid)

VALUES (u.productid, u.productname, u.quantityperunit, u.unitprice, u.discontinued, u.categoryid);

SELECT \* FROM products;

**OUTPUT:**



**------------------------------------------------------------------------------------------------------------------------------------------**

**7. List all orders with employee full names. (Inner join)**

SELECT

CONCAT(e.first\_name, ' ', e.last\_name) AS employeefullname,

o.employee\_id,

o.order\_id,

o.customer\_id,

o.order\_date,

o.required\_date,

o.shipped\_date,

o.ship\_via,

o.freight

FROM orders o

INNER JOIN employees e ON o.employee\_id = e.employee\_id;

**OUTPUT:**

