שלב 4:

פונקציה 1: מחזירה את מספר הבקבוקים שיוצרו עבור סוג יין מסוים

CREATE OR REPLACE FUNCTION get\_bottle\_count\_by\_type(winetype\_input VARCHAR)

RETURNS INTEGER AS $$

DECLARE

bottle\_count INTEGER;

BEGIN

SELECT SUM(numbottls)

INTO bottle\_count

FROM finalproduct\_

WHERE winetype\_ = winetype\_input;

IF bottle\_count IS NULL THEN

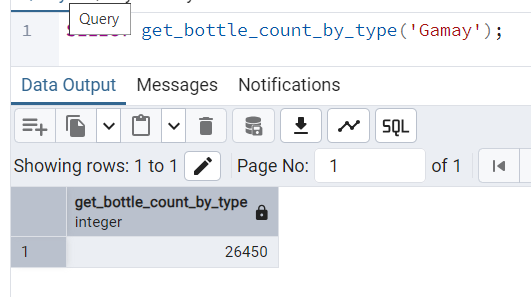
RETURN 0;

END IF;

RETURN bottle\_count;

END;

$$ LANGUAGE plpgsql;



-- ✅ פונקציה 2: מחזירה REF CURSOR עם כל ההזמנות מספק מסוים

CREATE OR REPLACE FUNCTION get\_orders\_by\_supplier(supplier\_name\_input VARCHAR)

RETURNS refcursor AS $$

DECLARE

orders\_cursor refcursor;

BEGIN

OPEN orders\_cursor FOR

SELECT o.orderid, o.orderdate

FROM orders\_local o

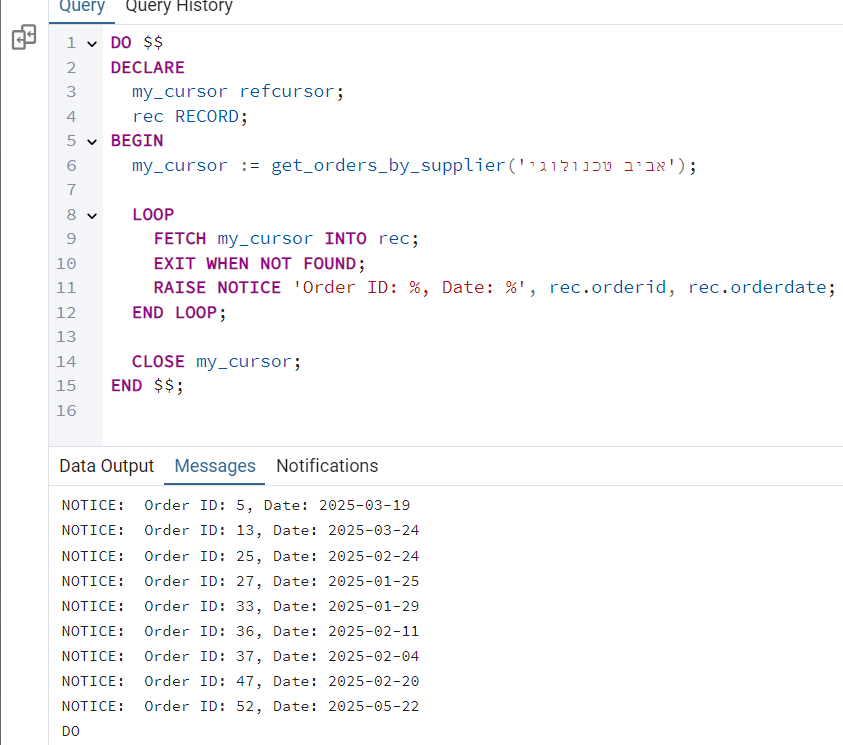
JOIN supplier\_local s ON o.supplierid = s.supplierid

WHERE s.suppliername = supplier\_name\_input;

RETURN orders\_cursor;

END;

$$ LANGUAGE plpgsql;



-- ✅ פרוצדורה 1: מעדכנת מחיר של כל מוצר מספק מסוים באחוז מסוים

CREATE OR REPLACE PROCEDURE increase\_prices\_by\_supplier(supplier\_name\_input VARCHAR, percent\_increase NUMERIC)

LANGUAGE plpgsql AS $$

DECLARE

prod RECORD;

BEGIN

FOR prod IN

SELECT p.productid, p.price

FROM product\_local p

JOIN orders\_local o ON o.supplierid = (SELECT supplierid FROM supplier\_local WHERE suppliername = supplier\_name\_input)

JOIN orderitems\_local oi ON o.orderid = oi.orderid AND oi.productid = p.productid

LOOP

UPDATE product\_local

SET price = price \* (1 + percent\_increase / 100)

WHERE productid = prod.productid;

END LOOP;

EXCEPTION

WHEN OTHERS THEN

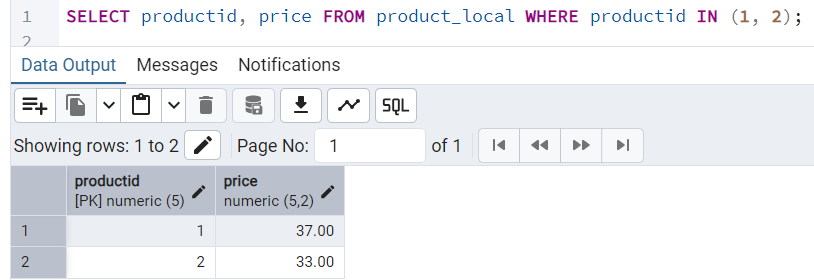
RAISE NOTICE 'Error during price update: %', SQLERRM;

END;

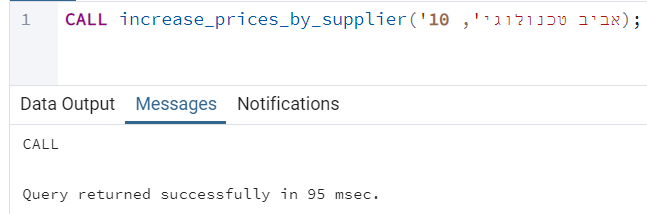
$$;

הוכחה שהפרוצדורה 1 עובדת:

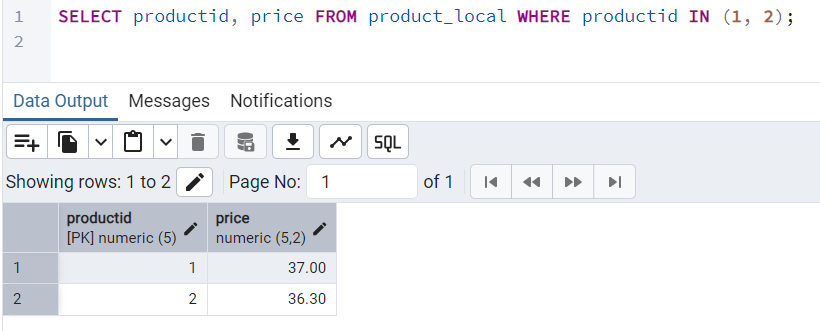
לפני הפעלת הפרוצדורה:



הרצת הפרוצדורה:



אחרי הפעלת הפרוצדורה:



-- ✅ פרוצדורה 2: מוסיפה חומר גלם חדש אם הוא לא קיים

CREATE OR REPLACE PROCEDURE add\_material\_if\_not\_exists(mat\_id INT, mat\_name VARCHAR, quantity FLOAT, supplier\_id INT)

LANGUAGE plpgsql AS $$

DECLARE

existing\_material RECORD;

BEGIN

SELECT \* INTO existing\_material FROM materials\_ WHERE materialid\_ = mat\_id;

IF NOT FOUND THEN

INSERT INTO materials\_ (materialid\_, name\_, quantityavailable\_, supplierid\_)

VALUES (mat\_id, mat\_name, quantity, supplier\_id);

ELSE

RAISE NOTICE 'Material already exists: %', existing\_material.name\_;

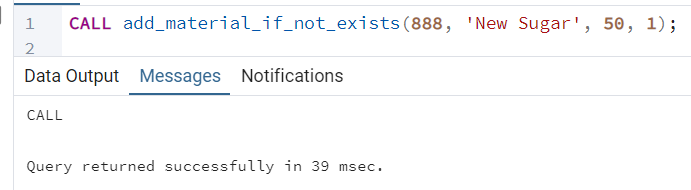
END IF;

END;

$$;

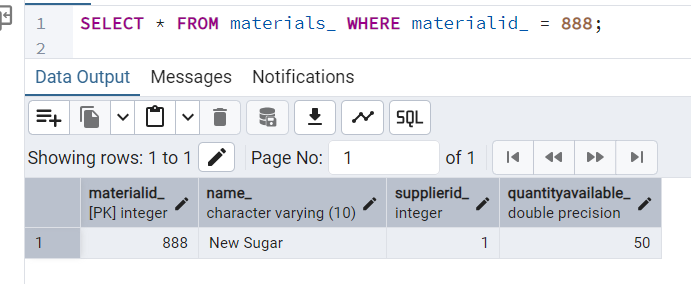
הוכחה שהפרוצדורה 2 עובדת:

הרצת הפרוצדורה:



א

אחרי הפעלת הפרוצדורה:



-- ✅ טריגר 1: עדכון תאריך עדכון אחרון במוצר כשיש שינוי מחיר

ALTER TABLE product\_local ADD COLUMN last\_updated TIMESTAMP;

CREATE OR REPLACE FUNCTION update\_last\_updated()

RETURNS TRIGGER AS $$

BEGIN

NEW.last\_updated := NOW();

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg\_update\_last\_updated

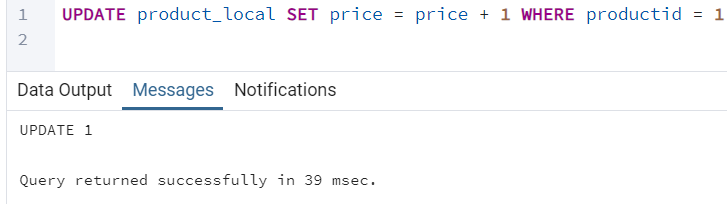
BEFORE UPDATE OF price ON product

FOR EACH ROW

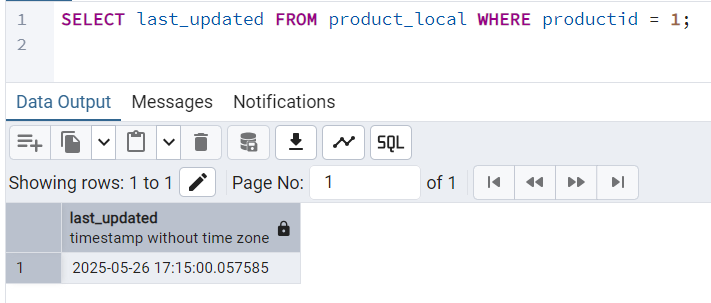
EXECUTE FUNCTION update\_last\_updated();

נוכיח שהטריגר עובד:

עשינו עדכון ערך בפרודקט:



הראנו איך זה נראה לאחר העדכון:



-- ✅ טריגר 2: וידוא שלא נכניס חומר גלם עם כמות שלילית

CREATE OR REPLACE FUNCTION validate\_material\_quantity()

RETURNS TRIGGER AS $$

BEGIN

IF NEW.quantityavailable\_ < 0 THEN

RAISE EXCEPTION 'Cannot insert material with negative quantity';

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

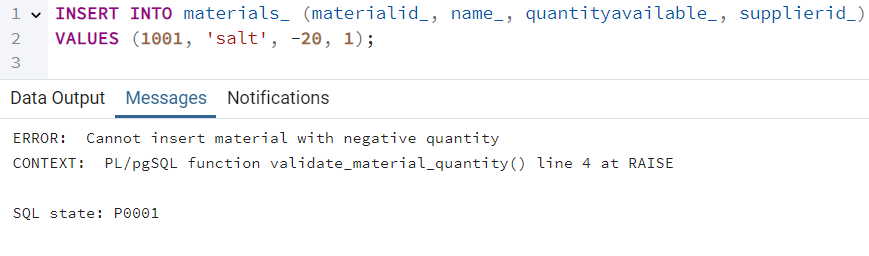
CREATE TRIGGER trg\_validate\_material\_insert

BEFORE INSERT ON materials\_

FOR EACH ROW

EXECUTE FUNCTION validate\_material\_quantity();

הוכחה שהטריגר עובד- ניסיון הכנסת חומר עם כמות שלילית:



-- ✅ תוכנית ראשית 1: מזמנת פונקציה של בקבוקים ופרוצדורה להוספת חומר

DO $$

DECLARE

bottle\_count INTEGER;

BEGIN

bottle\_count := get\_bottle\_count\_by\_type('Merlot');

RAISE NOTICE 'Total bottles for Merlot: %', bottle\_count;

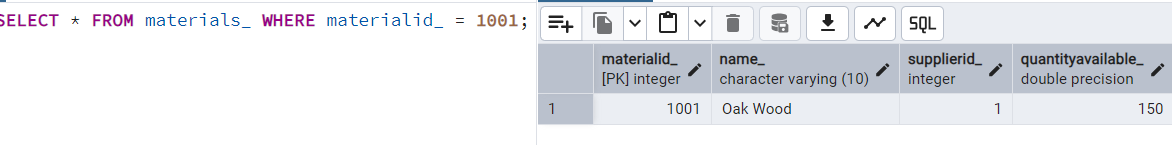
CALL add\_material\_if\_not\_exists(1001, 'Oak Wood', 150, 1);

END $$;

הרצה של התוכנית:



הוכחה שהפרוצדורה אכן עידכנה:



-- ✅ תוכנית ראשית 2: מזמנת פונקציית REF CURSOR ופרוצדורה לעדכון מחירים

DO $$

DECLARE

my\_cursor refcursor;

rec RECORD;

BEGIN

-- קריאה לפרוצדורה

CALL increase\_prices\_by\_supplier('Tara', 7);

-- קריאה לפונקציית REF CURSOR

my\_cursor := get\_orders\_by\_supplier('Tara');

-- הדפסת תוצאות ההזמנות

LOOP

FETCH my\_cursor INTO rec;

EXIT WHEN NOT FOUND;

RAISE NOTICE 'Order ID: %, Date: %', rec.orderid, rec.orderdate;

END LOOP;

CLOSE my\_cursor;

END $$;

הרצה של התוכנית:

(ההוכחה שהתוכנית באמת עדכנה את מחירי הספק- כבר הוכחה למעלה, פשוט הרצתי את התוכנית ושכחתי לצלם לפני בשביל להוכיח את האחרי אז נסתפק בהרצה שמראה למעלה לפני ואחרי העדכון)

