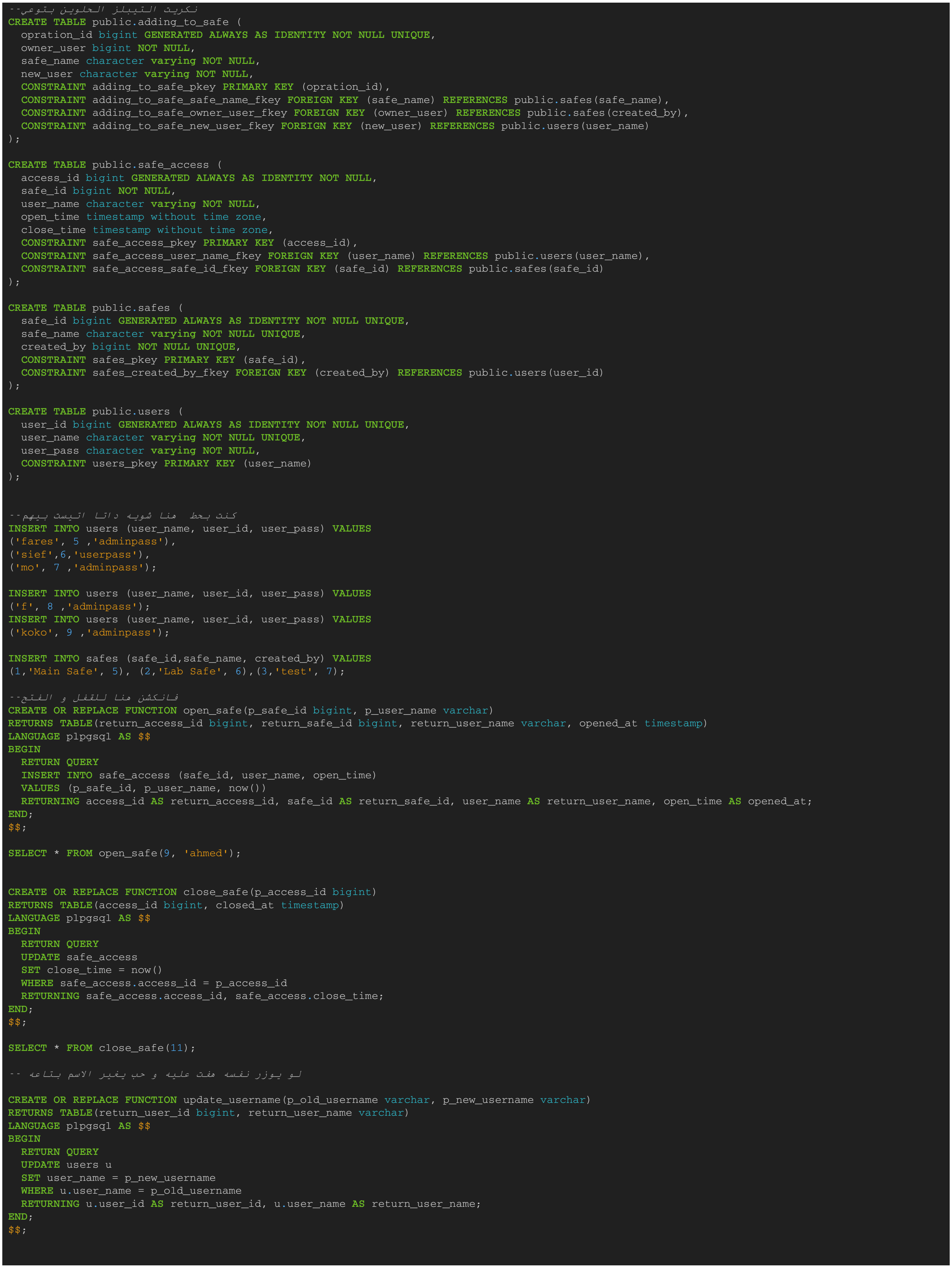
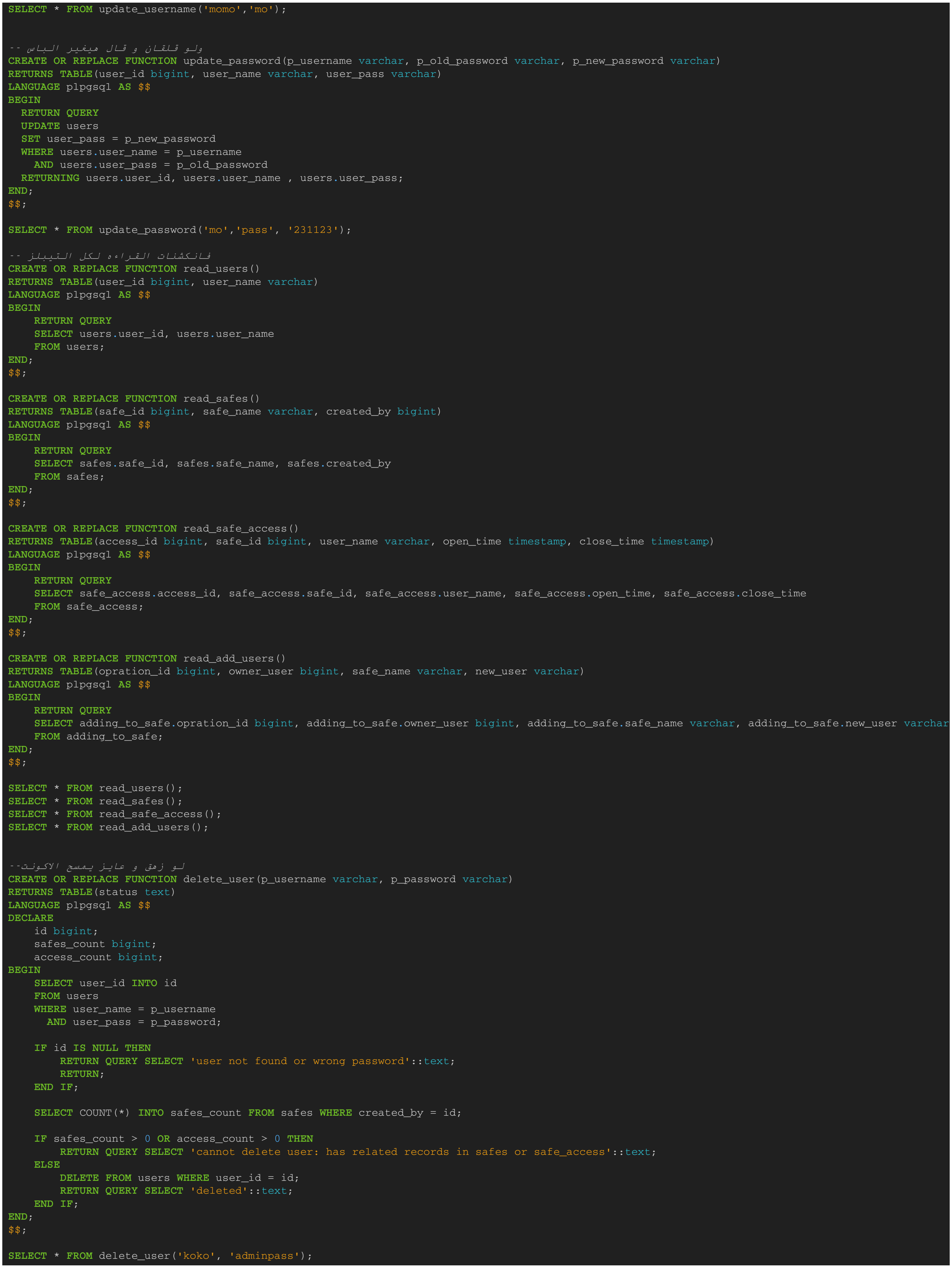


```sql
--نكريت التيبلز الحلوين بتوعي
CREATE TABLE public.adding_to_safe (
  opration_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL UNIQUE,
  owner_user bigint NOT NULL,
  safe_name character varying NOT NULL,
  new_user character varying NOT NULL,
  CONSTRAINT adding_to_safe_pkey PRIMARY KEY (opration_id),
  CONSTRAINT adding_to_safe_safe_name_fkey FOREIGN KEY (safe_name) REFERENCES public.safes(safe_name),
  CONSTRAINT adding_to_safe_owner_user_fkey FOREIGN KEY (owner_user) REFERENCES public.safes(created_by),
  CONSTRAINT adding_to_safe_new_user_fkey FOREIGN KEY (new_user) REFERENCES public.users(user_name)
);

CREATE TABLE public.safe_access (
  access_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL,
  safe_id bigint NOT NULL,
  user_name character varying NOT NULL,
  open_time timestamp without time zone,
  close_time timestamp without time zone,
  CONSTRAINT safe_access_pkey PRIMARY KEY (access_id),
  CONSTRAINT safe_access_user_name_fkey FOREIGN KEY (user_name) REFERENCES public.users(user_name),
  CONSTRAINT safe_access_safe_id_fkey FOREIGN KEY (safe_id) REFERENCES public.safes(safe_id)
);

CREATE TABLE public.safes (
  safe_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL UNIQUE,
  safe_name character varying NOT NULL UNIQUE,
  created_by bigint NOT NULL UNIQUE,
  CONSTRAINT safes_pkey PRIMARY KEY (safe_id),
  CONSTRAINT safes_created_by_fkey FOREIGN KEY (created_by) REFERENCES public.users(user_id)
);

CREATE TABLE public.users (
  user_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL UNIQUE,
  user_name character varying NOT NULL UNIQUE,
  user_pass character varying NOT NULL,
  CONSTRAINT users_pkey PRIMARY KEY (user_name)
);


--كنت بحط  هنا شويه داتا اتيست بيهم
INSERT INTO users (user_name, user_id, user_pass) VALUES
('fares', 5 ,'adminpass'),
('sief',6,'userpass'),
('mo', 7 ,'adminpass');

INSERT INTO users (user_name, user_id, user_pass) VALUES
('f', 8 ,'adminpass');
INSERT INTO users (user_name, user_id, user_pass) VALUES
('koko', 9 ,'adminpass');

INSERT INTO safes (safe_id,safe_name, created_by) VALUES
(1,'Main Safe', 5), (2,'Lab Safe', 6),(3,'test', 7);

--فانكشن هنا للقفل و الفتح
CREATE OR REPLACE FUNCTION open_safe(p_safe_id bigint, p_user_name varchar)
RETURNS TABLE(return_access_id bigint, return_safe_id bigint, return_user_name varchar, opened_at timestamp)
LANGUAGE plpgsql AS $$
BEGIN
  RETURN QUERY
  INSERT INTO safe_access (safe_id, user_name, open_time)
  VALUES (p_safe_id, p_user_name, now())
  RETURNING access_id AS return_access_id, safe_id AS return_safe_id, user_name AS return_user_name, open_time AS opened_at;
END;
$$;

SELECT * FROM open_safe(9, 'ahmed');


CREATE OR REPLACE FUNCTION close_safe(p_access_id bigint)
RETURNS TABLE(access_id bigint, closed_at timestamp)
LANGUAGE plpgsql AS $$
BEGIN
  RETURN QUERY
  UPDATE safe_access
  SET close_time = now()
  WHERE safe_access.access_id = p_access_id
  RETURNING safe_access.access_id, safe_access.close_time;
END;
$$;

SELECT * FROM close_safe(11);

-- لو يوزر نفسه هفت عليه و حب يغير الاسم بتاعه

CREATE OR REPLACE FUNCTION update_username(p_old_username varchar, p_new_username varchar)
RETURNS TABLE(return_user_id bigint, return_user_name varchar)
LANGUAGE plpgsql AS $$
BEGIN
  RETURN QUERY
  UPDATE users u
  SET user_name = p_new_username
  WHERE u.user_name = p_old_username
  RETURNING u.user_id AS return_user_id, u.user_name AS return_user_name;
END;
$$;
```



```sql
SELECT * FROM update_username('momo','mo');


-- ولو قلقان و قال هيغير الباس
CREATE OR REPLACE FUNCTION update_password(p_username varchar, p_old_password varchar, p_new_password varchar)
RETURNS TABLE(user_id bigint, user_name varchar, user_pass varchar)
LANGUAGE plpgsql AS $$
BEGIN
  RETURN QUERY
  UPDATE users
  SET user_pass = p_new_password
  WHERE users.user_name = p_username
    AND users.user_pass = p_old_password
  RETURNING users.user_id, users.user_name , users.user_pass;
END;
$$;

SELECT * FROM update_password('mo','pass', '231123');

-- فانكشنات القراءه لكل التيبلز
CREATE OR REPLACE FUNCTION read_users()
RETURNS TABLE(user_id bigint, user_name varchar)
LANGUAGE plpgsql AS $$
BEGIN
    RETURN QUERY
    SELECT users.user_id, users.user_name
    FROM users;
END;
$$;

CREATE OR REPLACE FUNCTION read_safes()
RETURNS TABLE(safe_id bigint, safe_name varchar, created_by bigint)
LANGUAGE plpgsql AS $$
BEGIN
    RETURN QUERY
    SELECT safes.safe_id, safes.safe_name, safes.created_by
    FROM safes;
END;
$$;

CREATE OR REPLACE FUNCTION read_safe_access()
RETURNS TABLE(access_id bigint, safe_id bigint, user_name varchar, open_time timestamp, close_time timestamp)
LANGUAGE plpgsql AS $$
BEGIN
    RETURN QUERY
    SELECT safe_access.access_id, safe_access.safe_id, safe_access.user_name, safe_access.open_time, safe_access.close_time
    FROM safe_access;
END;
$$;

CREATE OR REPLACE FUNCTION read_add_users()
RETURNS TABLE(opration_id bigint, owner_user bigint, safe_name varchar, new_user varchar)
LANGUAGE plpgsql AS $$
BEGIN
    RETURN QUERY
    SELECT adding_to_safe.opration_id bigint, adding_to_safe.owner_user bigint, adding_to_safe.safe_name varchar, adding_to_safe.new_user varchar
    FROM adding_to_safe;
END;
$$;

SELECT * FROM read_users();
SELECT * FROM read_safes();
SELECT * FROM read_safe_access();
SELECT * FROM read_add_users();


--لو زهق و عايز يمسح الاكونت
CREATE OR REPLACE FUNCTION delete_user(p_username varchar, p_password varchar)
RETURNS TABLE(status text)
LANGUAGE plpgsql AS $$
DECLARE
    id bigint;
    safes_count bigint;
    access_count bigint;
BEGIN
    SELECT user_id INTO id
    FROM users
    WHERE user_name = p_username
      AND user_pass = p_password;

    IF id IS NULL THEN
        RETURN QUERY SELECT 'user not found or wrong password'::text;
        RETURN;
    END IF;

    SELECT COUNT(*) INTO safes_count FROM safes WHERE created_by = id;

    IF safes_count > 0 OR access_count > 0 THEN
        RETURN QUERY SELECT 'cannot delete user: has related records in safes or safe_access'::text;
    ELSE
        DELETE FROM users WHERE user_id = id;
        RETURN QUERY SELECT 'deleted'::text;
    END IF;
END;
$$;

SELECT * FROM delete_user('koko', 'adminpass');
```

```sql
--الي كرييت الخزنه عايز يضيف حد معاه
CREATE OR REPLACE FUNCTION add_user(
    v_safe_name varchar,
    v_admin_id bigint,
    v_user_name varchar
)
RETURNS TABLE(return_safe_name varchar, return_owner_user bigint, return_new_user varchar)
LANGUAGE plpgsql AS $$
DECLARE
    owner_id bigint;
    already_in_safe bigint;
BEGIN

    SELECT s.created_by INTO owner_id
    FROM safes s
    WHERE s.safe_name = v_safe_name;

    IF owner_id IS DISTINCT FROM v_admin_id THEN
        RAISE EXCEPTION 'Only the owner can add users to this safe';
    END IF;

    SELECT sa.access_id INTO already_in_safe
    FROM safe_access sa
    WHERE sa.user_name = v_user_name
    LIMIT 1;

    IF already_in_safe IS NOT NULL THEN
        RAISE EXCEPTION 'User is already assigned to another safe';
    END IF;

    RETURN QUERY
    INSERT INTO adding_to_safe(owner_user, safe_name, new_user)
    VALUES (v_admin_id, v_safe_name, v_user_name)
    RETURNING safe_name AS return_safe_name, owner_user AS return_owner_user, new_user AS return_new_user;
END;
$$;

--هنا عايز يكرييت اكونت
CREATE OR REPLACE FUNCTION signup(
    p_username varchar,
    p_password varchar
)
RETURNS TABLE(return_user_id bigint, return_user_name varchar)
LANGUAGE plpgsql AS $$
DECLARE

    existing_user bigint;
BEGIN
    SELECT u.user_id INTO existing_user
    FROM users u
    WHERE u.user_name = p_username;

    IF existing_user IS NOT NULL THEN
        RAISE EXCEPTION 'Username "%" already exists', p_username;
    END IF;

    RETURN QUERY
    INSERT INTO users (user_name, user_pass)
    VALUES (p_username, p_password)
    RETURNING users.user_id AS return_user_id,
              users.user_name AS return_user_name;
END;
$$;

SELECT * FROM signup('ahmed', '12345');

--هنا فانكشن عشان اليوزر يسجل حسابه (موجود اصلا)
CREATE OR REPLACE FUNCTION login(
    p_username varchar,
    p_password varchar
)
RETURNS TABLE(return_user_id bigint, return_user_name varchar)
LANGUAGE plpgsql AS $$
BEGIN
    RETURN QUERY
    SELECT u.user_id, u.user_name
    FROM users u
    WHERE u.user_name = p_username
      AND u.user_pass = p_password;

    IF NOT FOUND THEN
        RAISE EXCEPTION 'Invalid username or password';
    END IF;
END;
$$;

SELECT * FROM login('ahmed', '12345');


--هنا اليوزر الي عمل اكونت ده عايز يعمل بقا خزنه يسترزق شويه
CREATE OR REPLACE FUNCTION create_safe(
    p_safe_name varchar,
    p_user_name varchar,
    p_user_pass varchar
)
```

```
RETURNS TABLE(safe_id bigint, safe_name varchar, created_by bigint)
LANGUAGE plpgsql AS $$
DECLARE
    v_user_id bigint;
BEGIN
    SELECT u.user_id INTO v_user_id
    FROM users u
    WHERE u.user_name = p_user_name
      AND u.user_pass = p_user_pass;

    IF v_user_id IS NULL THEN
        RAISE EXCEPTION 'Invalid username or password';
    END IF;
    IF EXISTS (SELECT 1 FROM safes s WHERE s.created_by = v_user_id) THEN
        RAISE EXCEPTION 'User already owns a safe';
    END IF;

    RETURN QUERY
    INSERT INTO safes (safe_name, created_by)
    VALUES (p_safe_name, v_user_id)
    RETURNING safes.safe_id, safes.safe_name, safes.created_by;
END;
$$;


SELECT * FROM create_safe('ahmed Safe', 'ahmed', '12345');
```