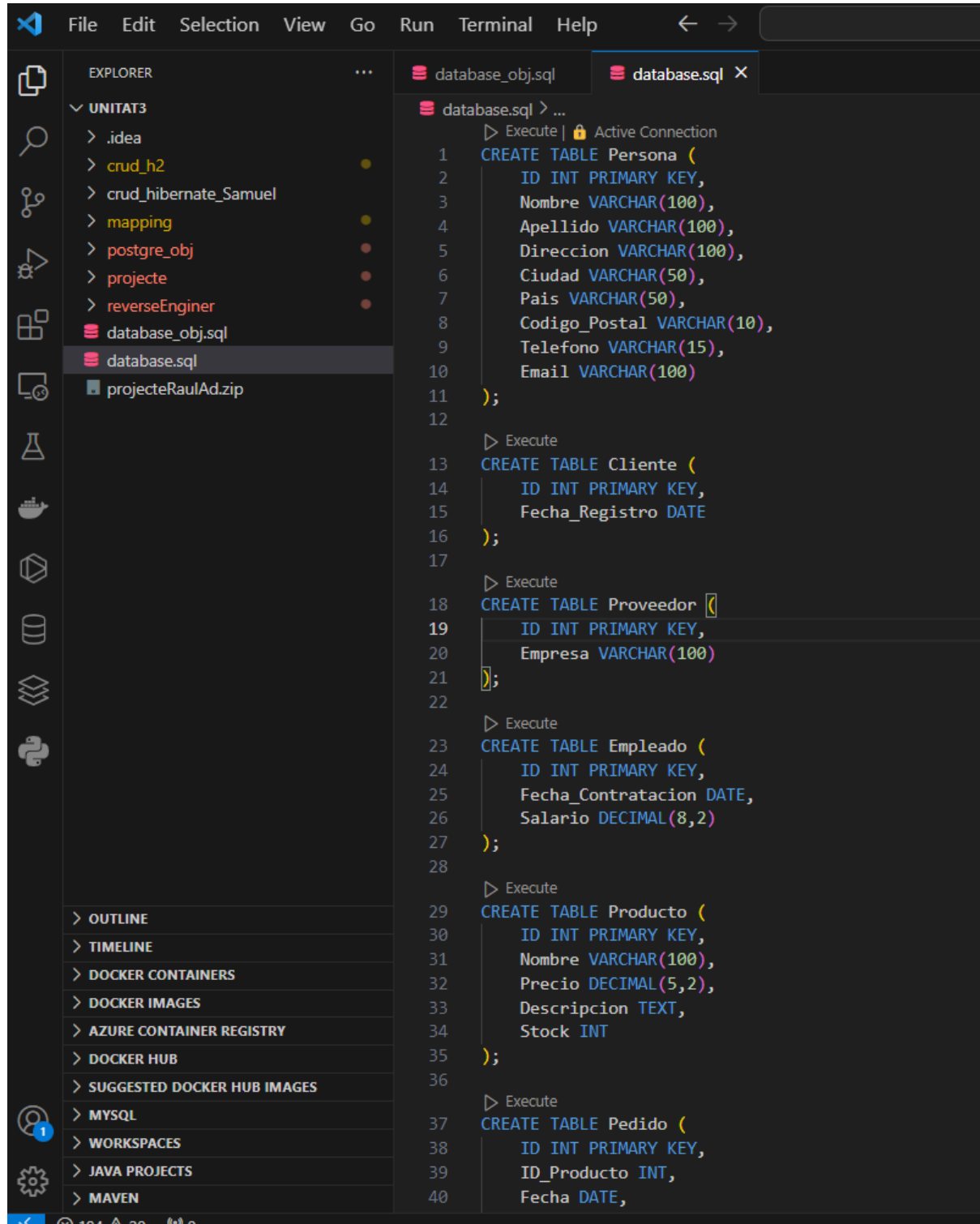


Practica relational:object-relational

Imagen del script .sql:	2
Imagen del script .sql para postgre aplicando el uso de objeto-relacional	3
Cambios realizados:	3
RESUMEN:	4

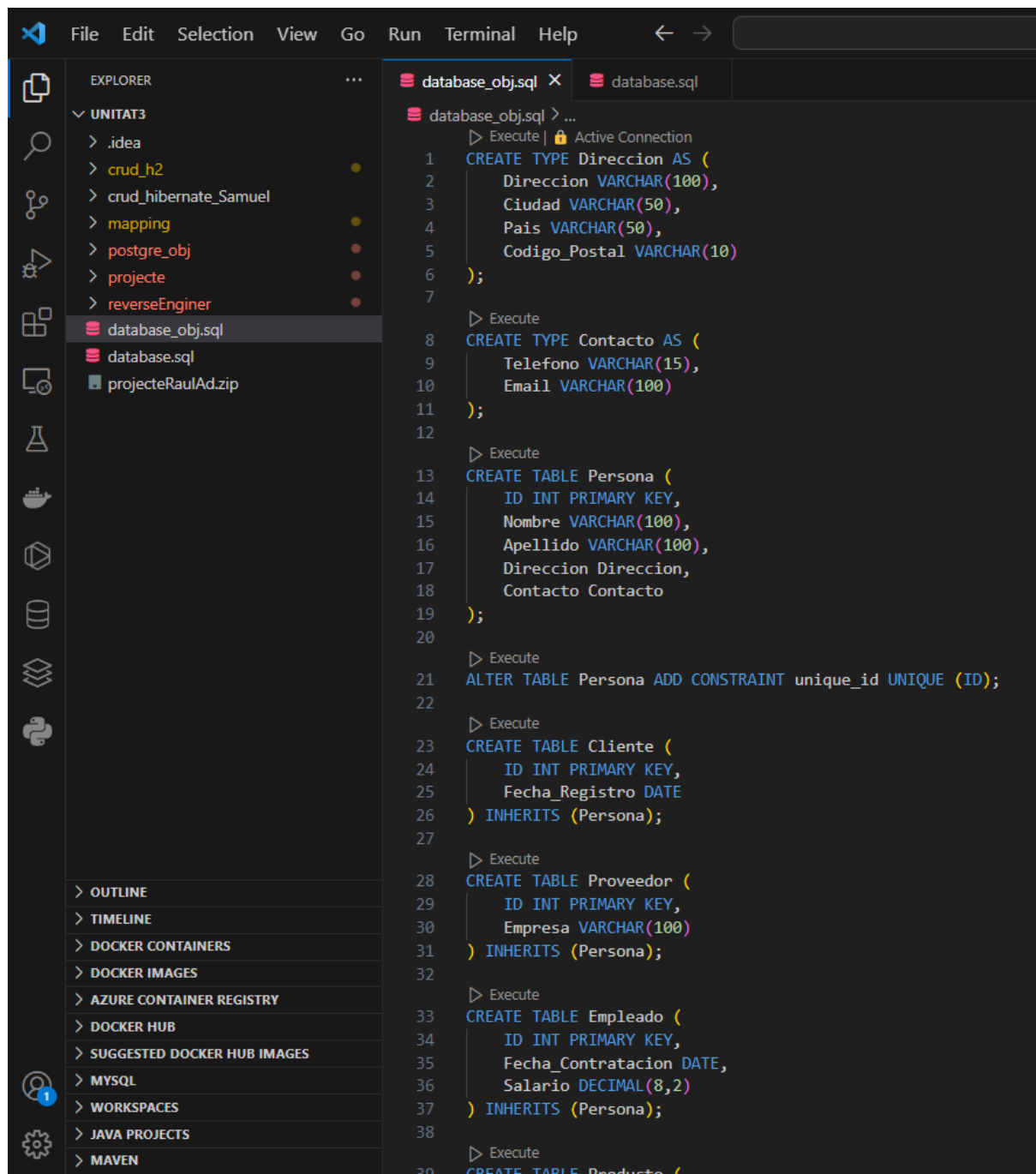
Imagen del script .sql:



The image shows a screenshot of an IDE (Integrated Development Environment) with a dark theme. The left sidebar contains the 'EXPLORER' view, showing a project structure under 'UNITAT3'. The files listed are: '.idea', 'crud_h2', 'crud_hibernate_Samuel', 'mapping', 'postgre_obj', 'projecte', 'reverseEngineer', 'database_obj.sql', 'database.sql', and 'projecteRaulAd.zip'. The 'database.sql' file is selected and highlighted. The main editor area displays the content of 'database.sql', which contains SQL code for creating several tables: 'Persona', 'Cliente', 'Proveedor', 'Empleado', 'Producto', and 'Pedido'. The code is syntax-highlighted, with keywords in blue, identifiers in green, and literals in red. The status bar at the bottom shows '194' lines, '20' columns, and '0' errors.

```
1 CREATE TABLE Persona (  
2     ID INT PRIMARY KEY,  
3     Nombre VARCHAR(100),  
4     Apellido VARCHAR(100),  
5     Direccion VARCHAR(100),  
6     Ciudad VARCHAR(50),  
7     Pais VARCHAR(50),  
8     Codigo_Postal VARCHAR(10),  
9     Telefono VARCHAR(15),  
10    Email VARCHAR(100)  
11 );  
12  
13 CREATE TABLE Cliente (  
14     ID INT PRIMARY KEY,  
15     Fecha_Registro DATE  
16 );  
17  
18 CREATE TABLE Proveedor (  
19     ID INT PRIMARY KEY,  
20     Empresa VARCHAR(100)  
21 );  
22  
23 CREATE TABLE Empleado (  
24     ID INT PRIMARY KEY,  
25     Fecha_Contratacion DATE,  
26     Salario DECIMAL(8,2)  
27 );  
28  
29 CREATE TABLE Producto (  
30     ID INT PRIMARY KEY,  
31     Nombre VARCHAR(100),  
32     Precio DECIMAL(5,2),  
33     Descripcion TEXT,  
34     Stock INT  
35 );  
36  
37 CREATE TABLE Pedido (  
38     ID INT PRIMARY KEY,  
39     ID_Producto INT,  
40     Fecha DATE,
```

Imagen del script .sql para postgre aplicando el uso de objeto-relacional



The screenshot shows an IDE with a dark theme. On the left is the Explorer panel showing a project structure with folders like .idea, crud_h2, crud_hibernate_Samuel, mapping, postgres_obj, projecte, reverseEngineer, database_obj.sql, database.sql, and projecteRaulAd.zip. The main editor displays a SQL script in database_obj.sql. The script includes several SQL statements for creating types and tables in PostgreSQL.

```
1 CREATE TYPE Direccion AS (  
2     Direccion VARCHAR(100),  
3     Ciudad VARCHAR(50),  
4     Pais VARCHAR(50),  
5     Codigo_Postal VARCHAR(10)  
6 );  
7  
8 CREATE TYPE Contacto AS (  
9     Telefono VARCHAR(15),  
10    Email VARCHAR(100)  
11 );  
12  
13 CREATE TABLE Persona (  
14     ID INT PRIMARY KEY,  
15     Nombre VARCHAR(100),  
16     Apellido VARCHAR(100),  
17     Direccion Direccion,  
18     Contacto Contacto  
19 );  
20  
21 ALTER TABLE Persona ADD CONSTRAINT unique_id UNIQUE (ID);  
22  
23 CREATE TABLE Cliente (  
24     ID INT PRIMARY KEY,  
25     Fecha_Registro DATE  
26 ) INHERITS (Persona);  
27  
28 CREATE TABLE Proveedor (  
29     ID INT PRIMARY KEY,  
30     Empresa VARCHAR(100)  
31 ) INHERITS (Persona);  
32  
33 CREATE TABLE Empleado (  
34     ID INT PRIMARY KEY,  
35     Fecha_Contratacion DATE,  
36     Salario DECIMAL(8,2)  
37 ) INHERITS (Persona);  
38  
39 CREATE TABLE Producto (  
40     ID INT PRIMARY KEY,  
41     Nombre VARCHAR(100),  
42     Precio DECIMAL(8,2)
```

Cambios realizados:

se han creado nuevos tipos, de los cuales van a contener atributos que luego vamos a implementar en otras tablas para que se hereden ede ellas. Estos nuevos tipos tienen datos de contacto y la direccion.

CREATE TYPE Direccion AS (

```
Direccion VARCHAR(100),  
Ciudad VARCHAR(50),  
Pais VARCHAR(50),  
Codigo_Postal VARCHAR(10)  
);
```

```
CREATE TYPE Contacto AS (  
    Telefono VARCHAR(15),  
    Email VARCHAR(100)  
);
```

Se ha creado estos nuevos tipos, luego una nueva tabla persona, el cual tiene como atributos estos tipos : CREATE TABLE Persona (

```
    ID INT PRIMARY KEY,  
    Nombre VARCHAR(100),  
    Apellido VARCHAR(100),  
    Direccion Direccion,  
    Contacto Contacto  
);
```

de esta forma la tabla persona utiliza los tipos creados de direccion y Contacto para trabajar con sus atributos

```
ALTER TABLE Persona ADD CONSTRAINT unique_id UNIQUE (ID);
```

esta clausula es utilizada para evitar que los identificadores se repitan (spoiler, vengo del futuro).

y finalmente heredamos de persona, de esta forma evitamos atributos duplicados, que podemos heredar en la tabla cliente, empleado y proveedor. (se heredan todos los campos)

RESUMEN:

De esta forma se consigue que una base de datos pueda ser más eficiente, y seria aún mucho más notoria en una base de datos con muchas tablas, que repiten atributos.