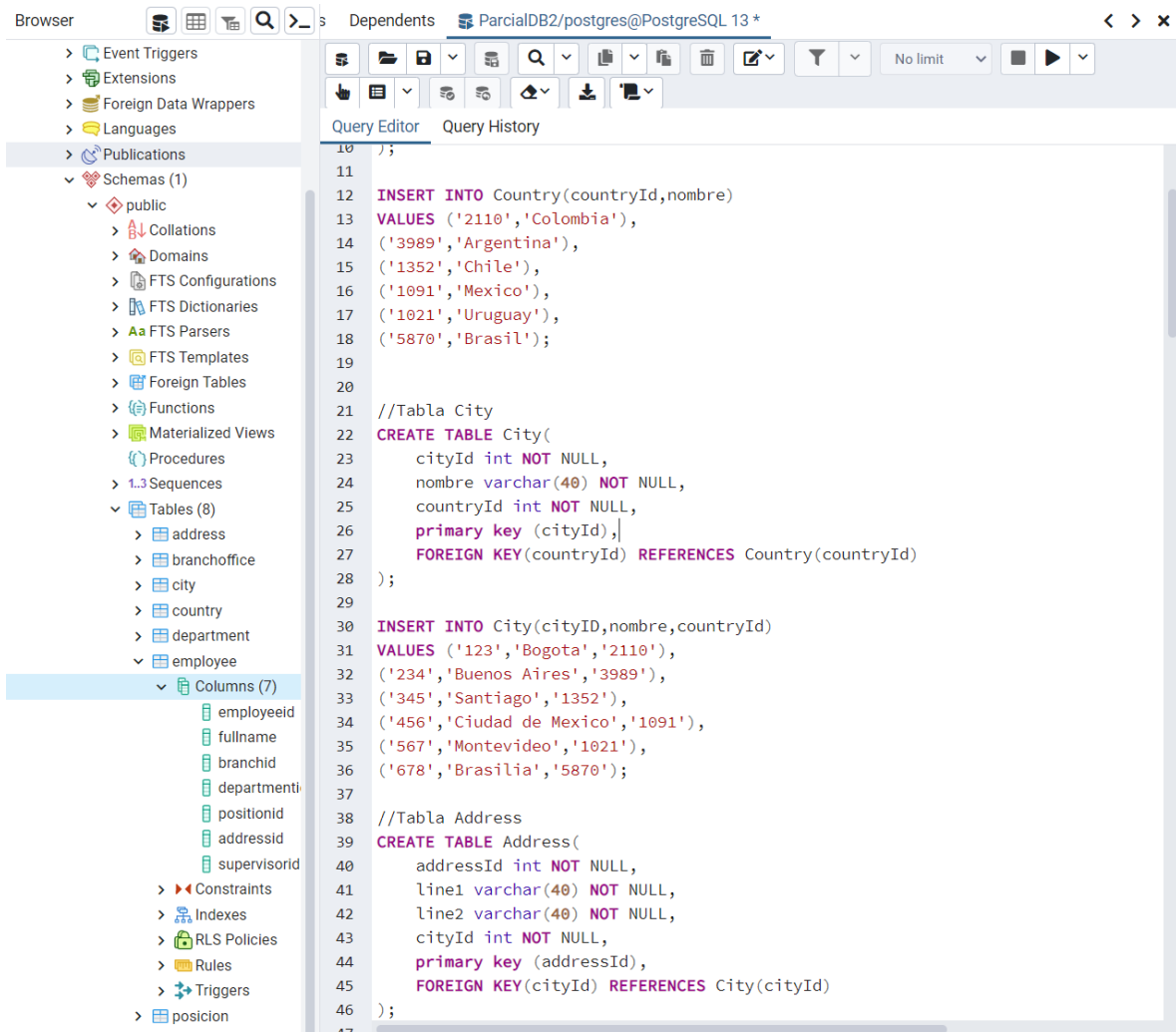


Parcial Base de datos 2

David Esteban Zamora Montero

1. Se crean las tablas e inserción de los datos correspondientes



The screenshot shows the PostgreSQL Query Editor interface. On the left, the 'Browser' pane displays the database structure, including Schemas, Tables, and Columns. The 'Columns' pane for the 'employee' table is expanded, showing columns like employeeid, fullname, branchid, departmentid, positionid, addressid, and supervisorid. The main 'Query Editor' pane contains the following SQL code:

```
10 ;
11
12 INSERT INTO Country(countryId,nombre)
13 VALUES ('2110','Colombia'),
14 ('3989','Argentina'),
15 ('1352','Chile'),
16 ('1091','Mexico'),
17 ('1021','Uruguay'),
18 ('5870','Brasil');
19
20
21 //Tabla City
22 CREATE TABLE City(
23     cityId int NOT NULL,
24     nombre varchar(40) NOT NULL,
25     countryId int NOT NULL,
26     primary key (cityId),
27     FOREIGN KEY(countryId) REFERENCES Country(countryId)
28 );
29
30 INSERT INTO City(cityID,nombre,countryId)
31 VALUES ('123','Bogota','2110'),
32 ('234','Buenos Aires','3989'),
33 ('345','Santiago','1352'),
34 ('456','Ciudad de Mexico','1091'),
35 ('567','Montevideo','1021'),
36 ('678','Brasilia','5870');
37
38 //Tabla Address
39 CREATE TABLE Address(
40     addressId int NOT NULL,
41     line1 varchar(40) NOT NULL,
42     line2 varchar(40) NOT NULL,
43     cityId int NOT NULL,
44     primary key (addressId),
45     FOREIGN KEY(cityId) REFERENCES City(cityId)
46 );
47
```

2. Se hace la creación de la función para ejecutar el trigger

```
160 CREATE OR REPLACE FUNCTION changes()
161 RETURNS TRIGGER
162 LANGUAGE PLPGSQL
163 AS
164 $$
165 DECLARE
166 city varchar(40);
167 cityId int;
168 countryId int;
169 nombre varchar(40);
170
171 BEGIN
172
173 IF NEW.fullName <> OLD.fullName THEN
174
175     INSERT INTO EmployeeAudit(employeeId, fullName,branchId, departmentId, positionId, addressId,registeredA
176     VALUES(old.employeeId, old.fullName, old.branchId, old.departmentId, old.positionId,old.addressId,now())
177
178 END IF;
179
180 RETURN NEW;
181 END;
182 $$
183
```

Data Output Explain Messages Notifications

CREATE FUNCTION

Query returned successfully in 75 msec.

3. Se crea el trigger correspondiente para ejecutar luego del update, delete o insert

```
184 CREATE TRIGGER changes_employee
185 AFTER INSERT OR DELETE OR UPDATE
186 ON employee
187 FOR EACH ROW
188 EXECUTE PROCEDURE changes();
189
```

Data Output Explain Messages Notifications

ERROR: ya existe un trigger «changes_employee» para la relación «employee»
SQL state: 42710

4. Se valida la tabla employee para saber como esta antes del cambio

```
191 SELECT * FROM employee;
192
193
```

	employeeid [PK] integer	fullname character varying (40)	branchid integer	departmentid integer	positionid integer	addressid integer	supervisorid integer	
1	13	Maria	2	23456	22	2	13	
2	14	Claudia	3	34567	23	3	14	
3	15	Daniela	4	45678	24	4	15	
4	16	Manuel	5	56789	25	5	16	
5	17	David	6	90123	26	6	17	
6	12	Camila	1	12345	21	1	12	

5. Se ejecuta el cambio

```
194 UPDATE employee
195 SET fullname = 'Julia'
196 WHERE employeeId = 12;
197
198
```

Data Output	Explain	Messages	Notifications
UPDATE 1			
Query returned successfully in 59 msec.			

6. Se valida la tabla employeeAudit con el cambio

```
198 SELECT * FROM EmployeeAudit;
199
```

	employeeid integer	fullname character varying (40)	branchid integer	departmentid integer	positionid integer	addressid integer	city character varying (40)	country character varying (40)	evento character varying (40)	registeredat date
1	12	Anita	1	12345	21	1	[null]	[null]	[null]	2021-09-16
2	12	Camila	1	12345	21	1	[null]	[null]	[null]	2021-09-16

7. Se crea la vista para poder ver la posición para el equipo de recursos humanos

```
200 CREATE MATERIALIZED VIEW RrhView AS
201     select
202         (Department.nombre) as NameDepartment,
203         (Posicion.nombre) as Posicion,
204         BranchOffice.nombre
205     from Department,Posicion,BranchOffice,Employee
206     where Department.departmentId = Employee.departmentId
207     and Posicion.positionId = Employee.positionId
208     and BranchOffice.nombre='SegundaNueva'
209     order by (Department.nombre);
```

Data Output Explain Messages Notifications

ERROR: la relación «rrhhview» ya existe
SQL state: 42P07

8. Se comprueba la vista

```
211 select * from RrhView;
```

Data Output Explain Messages Notifications

	namedepartment character varying (40)	posicion character varying (40)	nombre character varying (40)
1	alejada	alejada	SegundaNueva
2	centralizada	centralizada	SegundaNueva
3	medianaCerca	medianaCerca	SegundaNueva
4	medianalejana	medianalejana	SegundaNueva
5	muycerca	muycerca	SegundaNueva
6	muylejos	muylejos	SegundaNueva