

De la creación a la manipulación de una base de datos relacional

NOMBRE Y APELLIDOS: Paula Corbatón Álvarez

EJERCICIO 1 (35%)

- a) Tal y como nos indica el ejercicio creamos las columnas, la función y el disparador

pgAdmin

Browser File Object Tools Help

Query Query History

```

1 -- a)
2 ALTER TABLE erp.tb_pump
3 ADD COLUMN updated_dt_tm TIMESTAMP NOT NULL DEFAULT now(); -- creamos la columna que no permita valores nulos
4 -- función
5 CREATE OR REPLACE FUNCTION update_tb_pump_table_a()
6 RETURNS TRIGGER AS
7 $$
8 BEGIN
9   UPDATE erp.tb_pump
10  SET updated_dt_tm = CURRENT_TIMESTAMP
11  WHERE pm_id = NEW.pm_id;
12
13  RETURN NEW;
14 END;
15 $$;
16 LANGUAGE plpgsql;
17 |
18 -- trigger
19 CREATE TRIGGER update_timestamp
20 AFTER INSERT ON erp.tb_pump
21 FOR EACH ROW EXECUTE PROCEDURE update_tb_pump_table_a();
22

```

Data Output Messages Notifications

CREATE FUNCTION

Query returned successfully in 56 msec.

Total rows: 0 of 0 Query complete 00:00:00.056

Ln 17, Col 1

Podemos hacer un insert para comprobar que funciona

pgAdmin

Browser File Object Tools Help

Query Query History

```

4 -- function
5 CREATE OR REPLACE FUNCTION update_tb_pump_table_a()
6 RETURNS TRIGGER AS
7 $$
8 BEGIN
9   UPDATE erp.tb_pump
10  SET updated_dt_tm = CURRENT_TIMESTAMP
11  WHERE pm_id = NEW.pm_id;
12
13  RETURN NEW;
14 END;
15 $$;
16 LANGUAGE plpgsql;
17 |
18 -- trigger
19 CREATE TRIGGER update_timestamp
20 AFTER INSERT ON erp.tb_pump
21 FOR EACH ROW EXECUTE PROCEDURE update_tb_pump_table_a();
22
23 --insert
24 INSERT INTO erp.tb_pump(pm_id,pm_parent_id,pm_descr,gas_id) VALUES (57,22,'Surtidor N°1','GS01');
25
26

```

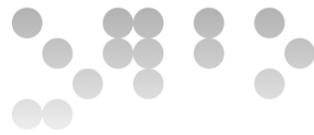
Data Output Messages Notifications

INSERT 0 1

Query returned successfully in 48 msec.

Total rows: 0 of 0 Query complete 00:00:00.048

Ln 24, Col 1



Si vemos la tabla tb_pump podemos comprobar que todo ha salido correctamente

pgAdmin

File Object Tools Help

Browser dbdw_pec3

- Casts
- Catalogs
- Event Triggers
- Extensions
- Foreign Data Wrappers
- Languages
- Publications
- Schemas (2)
 - erp
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (7)
 - tb_cars
 - tb_fuel_price
 - tb_gas_station
 - tb_invoice
 - tb_lines_invoice
 - tb_pump
 - tb_refueling
 - Trigger Functions
 - Types

Dashboard Properties SQL Statistics Dependencies Dependents Processes PEC3_Ej1.sql* PEC3_Ej2.sql dbdw_pec3/postgres@PostgreSQL 15*

Query Query History

```
1 SELECT pm_id, pm_parent_id, gas_id, pm_descr, updated_dt_tm
2      FROM erp.tb_pump;
```

Data Output Messages Notifications

pm_id	pm_parent_id	gas_id	pm_descr	updated_dt_tm
45	45	17	GS04	Diesel
46	46	17	GS04	Gasolina
47	47	18	GS04	Diesel
48	48	18	GS04	Gasolina
49	49	19	GS04	Diesel
50	50	19	GS04	Gasolina
51	51	20	GS05	Diesel
52	52	20	GS05	Gasolina
53	53	21	GS05	Diesel
54	54	21	GS05	Gasolina
55	55	22	GS05	Diesel
56	56	22	GS05	Gasolina

Total rows: 57 of 57 Query complete 00:00:00.167 Rows selected: 1

b) Tal y como nos indica el ejercicio creamos las columnas, la función y el disparador

pgAdmin

File Object Tools Help

Browser dbdw_pec3

- Casts
- Catalogs
- Event Triggers
- Extensions
- Foreign Data Wrappers
- Languages
- Publications
- Schemas (2)
 - erp
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (7)
 - tb_cars
 - tb_fuel_price
 - tb_gas_station
 - tb_invoice
 - tb_lines_invoice
 - tb_pump
 - tb_refueling
 - Trigger Functions
 - Types

Dashboard Properties SQL Statistics Dependencies Dependents Processes PEC3_Ej1.sql* PEC3_Ej2.sql dbdw_pec3/postgres@PostgreSQL 15*

Query Query History

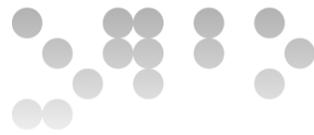
```
27 -- b)
28 ALTER TABLE erp.tb_lines_invoice
29 ADD COLUMN line_updated_dt_tm VARCHAR(20);
30
31 -- function
32 -- Entiendo que como el enunciado no dice nada no se tienen que alterar los valores ya existentes
33 CREATE OR REPLACE FUNCTION fn_line_inserted()
34 RETURNS TRIGGER AS
35 $$
36 BEGIN
37     NEW.line_updated_dt_tm = to_char(now(), 'YYYY-MM-DD hh:mm');
38     RETURN NEW;
39 END;
40 $$
41 LANGUAGE plpgsql;
42
43 -- trigger
44 CREATE TRIGGER tg_line_inserted
45 BEFORE INSERT OR UPDATE ON erp.tb_lines_invoice
46 FOR EACH ROW EXECUTE PROCEDURE fn_line_inserted();
47
48
```

Data Output Messages Notifications

```
CREATE TRIGGER
```

Query returned successfully in 48 msec.

Total rows: 0 of 0 Query complete 00:00:00.048



Podemos hacer un insert para comprobar que funciona (también podríamos haber hecho un update)

pgAdmin File Object Tools Help

Browser dbdw_pec3

- Casts
- Catalogs
- Event Triggers
- Extensions
- Foreign Data Wrappers
- Languages
- Publications
- Schemas (2)
 - erp
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Types
- Tables (7)
 - tb_cars
 - tb_fuel_price
 - tb_gas_station
 - tb_invoice
 - tb_lines_invoice
 - tb_pump
 - tb_refueling
 - Trigger Functions
 - Types

Query History

```

32 -- Entiendo que como el enunciado no dice nada no se tienen que alterar los valores ya existentes
33 CREATE OR REPLACE FUNCTION fn_line_inserted()
34 RETURNS TRIGGER AS
35 $$
36 BEGIN
37     NEW.line_updated_dt_tm = to_char(now(), 'YYYY-MM-DD hh:mm');
38     RETURN NEW;
39 END;
40 $$;
41 LANGUAGE plpgsql;
42
43 -- trigger
44 CREATE TRIGGER tg_line_inserted
45 BEFORE INSERT OR UPDATE ON erp.tb_lines_invoice
46 FOR EACH ROW EXECUTE PROCEDURE fn_line_inserted();
47
48 --- insert
49 INSERT INTO erp.tb_lines_invoice(
50     inv_id, linv_id, cars_registration, gas_id, linv_liters, linv_amount
51     VALUES (1,43,'0815GVR','GS02 ',60,120.54);
52
53
54

```

Data Output Messages Notifications

INSERT 0 1

Query returned successfully in 72 msec.

Total rows: 0 of 0 Query complete 00:00:00.072 Ln 50, Col 1

Si vemos la tabla tb_lines_invoice podemos comprobar que todo ha salido correctamente

pgAdmin File Object Tools Help

Browser dbdw_pec3

- Casts
- Catalogs
- Event Triggers
- Extensions
- Foreign Data Wrappers
- Languages
- Publications
- Schemas (2)
 - erp
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Types
- Tables (7)
 - tb_cars
 - tb_fuel_price
 - tb_gas_station
 - tb_invoice
 - tb_lines_invoice
 - tb_pump
 - tb_refueling
 - Trigger Functions
 - Types

Query History

```

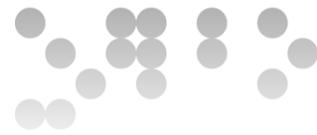
1 SELECT inv_id, linv_id, cars_registration, gas_id, linv_liters, linv_amount, line_updated_dt_tm
2 FROM erp.tb_lines_invoice;

```

Data Output Messages Notifications

Inv_Id	Linv_Id	Cars_Registration	Gas_Id	Linv_Liters	Linv_Amount	Line_Updated_dt_tm
89	3	16 3685HDP	GS01	125	250.60	[null]
90	3	17 3685HDP	GS02	63	123.33	[null]
91	3	18 4273GFK	GS01	104	199.95	[null]
92	3	19 4273GFK	GS02	156	309.36	[null]
93	3	20 4273GFK	GS03	84	179.68	[null]
94	3	21 4273GFK	GS04	64	132.74	[null]
95	3	22 4273GFK	GS05	70	142.46	[null]
96	3	23 5649JSN	GS01	116	235.30	[null]
97	3	24 5649JSN	GS02	70	145.54	[null]
98	3	25 5649JSN	GS03	72	158.32	[null]
99	3	26 5649JSN	GS04	76	168.18	[null]
100	3	27 5649JSN	GS05	78	169.58	[null]
101	3	28 6392KPT	GS01	116	231.04	[null]
102	3	29 6392KPT	GS02	76	152.48	[null]
103	3	30 6392KPT	GS03	98	196.18	[null]
104	3	31 6392KPT	GS04	42	83.96	[null]
105	1	43 0815GYR	GS02	60	120.54	2022-12-18 06:12

Total rows: 105 of 105 Query complete 00:00:00.071 Rows selected: 1 Ln 1, Col 1



c) Tal y como nos indica el ejercicio creamos las columnas, la función y el disparador

Screenshot of PgAdmin 4 showing the creation of a function and trigger.

```

PgAdmin  File  Object  Tools  Help
Browser  dbdw_pec3  dbdw_pec3/postgres@PostgreSQL 15  Dashboard  Properties  SQL  Statistics  Dependencies  Dependents  Processes  PEC3_Ej1.sql*  PEC3_Ej2.sql
Query  Query History
55
56 --c)
57 ALTER TABLE erp.tb_invoice ADD inv_updated_dt DATE NULL;
58 ALTER TABLE erp.tb_invoice ADD inv_update_counter INTEGER NULL DEFAULT 0;
59 ALTER TABLE erp.tb_invoice ADD inv_insert_counter INTEGER NULL DEFAULT 0;
60 --- function
61 CREATE OR REPLACE FUNCTION fn_invoice_updated()
62 RETURNS TRIGGER AS
63 $$
64 BEGIN
65   UPDATE erp.tb_invoice
66   SET inv_updated_dt = to_date(line_updated_dt_tm, 'YYYY-MM-DD') FROM erp.tb_lines_invoice
67   WHERE erp.tb_lines_invoice.inv_id = erp.tb_invoice.inv_id;
68   IF TG_OP = 'UPDATE' THEN
69     UPDATE erp.tb_invoice
70     SET inv_update_counter = inv_update_counter +1
71     WHERE NEW.inv_id = erp.tb_invoice.inv_id;
72   ELSE
73     UPDATE erp.tb_invoice
74     SET inv_insert_counter = inv_insert_counter +1
75     WHERE NEW.inv_id = erp.tb_invoice.inv_id;
76   END IF;
77   RETURN NEW;
78 END;
79 $$ LANGUAGE plpgsql;
80 --- trigger
81 CREATE TRIGGER tr_update_inv_updated_dt
82 BEFORE UPDATE OR INSERT ON erp.tb_lines_invoice
83 FOR EACH ROW
84 EXECUTE PROCEDURE fn_invoice_updated();
85
Data Output  Messages  Notifications
CREATE TRIGGER
Total rows: 0 of 0  Query complete 00:00:00.063
Ln 69, Col 30

```

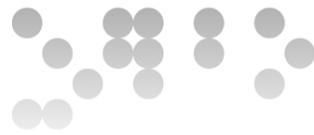
Podemos hacer un update para comprobar que funciona (también podríamos haber hecho un insert)

Screenshot of PgAdmin 4 showing an update query and its successful execution.

```

PgAdmin  File  Object  Tools  Help
Browser  dbdw_pec3  dbdw_pec3/postgres@PostgreSQL 15  Dashboard  Properties  SQL  Statistics  Dependencies  Dependents  Processes  PEC3_Ej1.sql*  PEC3_Ej2.sql  dbdw_pec3/po...
Query  Query History
61 $$
62 BEGIN
63   UPDATE erp.tb_invoice
64   SET inv_updated_dt = to_date(line_updated_dt_tm, 'YYYY-MM-DD') FROM erp.tb_lines_invoice
65   WHERE erp.tb_lines_invoice.inv_id = erp.tb_invoice.inv_id;
66   IF TG_OP = 'UPDATE' THEN
67     UPDATE erp.tb_invoice
68     SET inv_update_counter = inv_update_counter +1
69     WHERE NEW.inv_id = erp.tb_invoice.inv_id;
70   ELSE
71     UPDATE erp.tb_invoice
72     SET inv_insert_counter = inv_insert_counter +1
73     WHERE NEW.inv_id = erp.tb_invoice.inv_id;
74   END IF;
75   RETURN NEW;
76 END;
77 $$ LANGUAGE plpgsql;
78
79 --- trigger
80 CREATE TRIGGER tr_update_inv_updated_dt
81 BEFORE UPDATE OR INSERT ON erp.tb_lines_invoice
82 FOR EACH ROW
83 EXECUTE PROCEDURE fn_invoice_updated();
84
85 -- updates
86 UPDATE erp.tb_lines_invoice SET linv_amount = 1000.50 WHERE inv_id = 1 AND linv_id = 2;
87 UPDATE erp.tb_lines_invoice SET linv_amount = 1531.10 WHERE inv_id = 1 AND linv_id = 2;
Data Output  Messages  Notifications
UPDATE 1
Query returned successfully in 49 msec.
Total rows: 0 of 0  Query complete 00:00:00.049
Ln 85, Col 11

```



Si vemos la tabla tb_lines_invoice podemos comprobar que todo ha salido correctamente

pgAdmin

File Object Tools Help

Browser dbdw_pec3

- Casts
- Catalogs
- Event Triggers
- Extensions
- Foreign Data Wrappers
- Languages
- Publications
- Schemas (2)
 - erp
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
- Tables (7)
 - tb_cars
 - tb_fuel_price
 - tb_gas_station
 - tb_invoice
 - tb_lines_invoice
 - tb_pump
 - tb_refueling
- Trigger Functions
- Types

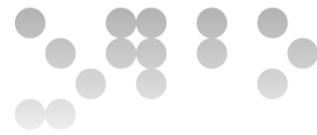
dbdw_pec3/postgres@PostgreSQL 15

Query History Data Output Messages Notifications

inv_id	linv_id	cars_registration	gas_id	linv_liters	linv_amount	line_updated_dt_tm
85	3	14	2093GSW	GS04	36	72.68 [null]
86	3	15	2093GSW	GS05	41	87.70 [null]
87	3	16	3685HDP	GS01	125	250.60 [null]
88	3	17	3685HDP	GS02	63	123.33 [null]
89	3	18	4273GFK	GS01	104	199.95 [null]
90	3	19	4273GFK	GS02	156	309.36 [null]
91	3	20	4273GFK	GS03	84	179.68 [null]
92	3	21	4273GFK	GS04	64	132.74 [null]
93	3	22	4273GFK	GS05	70	142.46 [null]
94	3	23	5649JSN	GS01	116	235.30 [null]
95	3	24	5649JSN	GS02	70	145.54 [null]
96	3	25	5649JSN	GS03	72	158.32 [null]
97	3	26	5649JSN	GS04	76	168.18 [null]
98	3	27	5649JSN	GS05	78	169.58 [null]
99	3	28	6392KPT	GS01	116	231.04 [null]
100	3	29	6392KPT	GS02	76	152.48 [null]
101	3	30	6392KPT	GS03	98	196.18 [null]
102	3	31	6392KPT	GS04	42	83.96 [null]
103	1	43	0019GYR	GS02	60	120.54 2022-12-18 06:12
104	1	1	0019GVM	GS01	80	1000.50 2022-12-18 07:12
105	1	2	0019GVM	GS02	96	1531.10 2022-12-18 07:12

Total rows: 105 of 105 Query complete 00:00:00.122 Rows selected: 2

Ln 2, Col 28



EJERCICIO 2 (55%)

- a) Tal y como nos indica el ejercicio creamos la columna, la función y el disparador

Screenshot of pgAdmin 4 showing the creation of a function and trigger.

```

--a)
-- add a new column
ALTER TABLE erp.tb_refueling ADD COLUMN rf_cost NUMERIC;
---- FUNCIÓN:
CREATE OR REPLACE FUNCTION fn_calc_cost()
RETURNS TRIGGER AS
$$
DECLARE
    fuel_type CHARACTER(10);
    fuel_import NUMERIC(12,3);
BEGIN
    fuel_type= (SELECT cars_fuel
                FROM erp.tb_cars
               WHERE cars_registration = NEW.cars_registration);

    fuel_import = (SELECT fp_import
                  FROM erp.tb_fuel_price
                 WHERE fp_date = NEW.rf_date AND fp_fuel = fuel_type);
    NEW.rf_cost = fuel_import * NEW.rf_liters;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
--- trigger
CREATE TRIGGER tg_refueling_cost
BEFORE INSERT ON erp.tb_refueling
FOR EACH ROW
EXECUTE PROCEDURE fn_calc_cost();

```

The query was successfully executed in 50 msec. Total rows: 26 of 26 Query complete 00:00:00.050 Ln 27, Col 34

Podemos hacer un insert para comprobar que funciona

Screenshot of pgAdmin 4 showing an insert operation into the tb_refueling table.

```

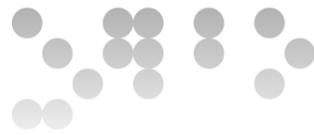
--a)
-- add a new column
ALTER TABLE erp.tb_refueling ADD COLUMN rf_cost NUMERIC;
---- FUNCIÓN:
CREATE OR REPLACE FUNCTION fn_calc_cost()
RETURNS TRIGGER AS
$$
DECLARE
    fuel_type CHARACTER(10);
    fuel_import NUMERIC(12,3);
BEGIN
    fuel_type= (SELECT cars_fuel
                FROM erp.tb_cars
               WHERE cars_registration = NEW.cars_registration);

    fuel_import = (SELECT fp_import
                  FROM erp.tb_fuel_price
                 WHERE fp_date = NEW.rf_date AND fp_fuel = fuel_type);
    NEW.rf_cost = fuel_import * NEW.rf_liters;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
--- trigger
CREATE TRIGGER tg_refueling_cost
BEFORE INSERT ON erp.tb_refueling
FOR EACH ROW
EXECUTE PROCEDURE fn_calc_cost();

-- insert
INSERT INTO erp.tb_refueling (gas_id,cars_registration,rf_liters,rf_date,pm_id,rf_km)
VALUES ('GS01','0019GVM',47,to_date('01-08-2022','DD-MM-YYYY'),23,234561);

```

INSERT 0 1 Query returned successfully in 102 msec. Total rows: 0 of 0 Query complete 00:00:00.102 Ln 31, Col 75



Si vemos la tabla tb_refueling podemos comprobar que todo ha salido correctamente

pgAdmin

File Object Tools Help

Browser

- > Catalogs
- > Event Triggers
- > Extensions
- > Foreign Data Wrappers
- > Languages
- > Publications
- > Schemas (2)
 - > erp
 - > Aggregates
 - > Collations
 - > Domains
 - > FTS Configurations
 - > FTS Dictionaries
 - > FTS Parsers
 - > FTS Templates
 - > Foreign Tables
 - > Functions
 - > Materialized Views
 - > Operators
 - > Procedures
 - > 1.3 Sequences
 - > Tables (8)
 - > tb_cars
 - > tb_fuel_price
 - > tb_gas_station
 - > tb_invoice
 - > tb_invoice_cost_sui
 - > tb_lines_invoice
 - > tb_pump
 - > tb_refueling
 - > Trigger Functions
 - > Types

Query Query History

```
1 SELECT gas_id, cars_registration, pm_id, rf_liters, rf_date, rf_km, rf_cost
2     FROM erp.tb_refueling;
```

Data Output Messages Notifications

gas_id	cars_registration	pm_id	rf_liters	rf_date	rf_km	rf_cost
100	GS01	9421GMT	23	39	2022-08-20	283490
101	GS01	9421GMT	23	38	2022-08-23	[null]
102	GS01	9421GMT	25	33	2022-09-07	284440
103	GS01	9421GMT	25	40	2022-09-13	305216
104	GS02	9421GMT	35	41	2022-08-02	305899
105	GS02	9421GMT	35	41	2022-08-17	306599
106	GS02	9421GMT	33	33	2022-08-26	307282
107	GS02	9421GMT	33	39	2022-09-10	307932
108	GS03	9421GMT	37	42	2022-08-05	308632
109	GS03	9421GMT	41	31	2022-08-29	309316
110	GS04	9421GMT	47	41	2022-08-08	309966
111	GS04	9421GMT	45	35	2022-09-01	[null]
112	GS05	9421GMT	53	39	2022-08-11	311149
113	GS05	9421GMT	55	41	2022-09-04	311666
114	GS01	0019GVM	23	47	2022-08-01	234561
						90.663

Total rows: 114 of 114 Query complete 00:00:00.070 Rows selected: 1

Ln 2, Col 24

b) Tal y como nos indica el ejercicio creamos la función

pgAdmin

File Object Tools Help

Browser

- > dbdw_pec3
 - > Casts
 - > Catalogs
 - > Event Triggers
 - > Extensions
 - > Foreign Data Wrappers
 - > Languages
 - > Publications
 - > Schemas (2)
 - > erp
 - > Aggregates
 - > Collations
 - > Domains
 - > FTS Configurations
 - > FTS Dictionaries
 - > FTS Parsers
 - > FTS Templates
 - > Foreign Tables
 - > Functions
 - > Materialized Views
 - > Operators
 - > Procedures
 - > 1.3 Sequences
 - > Tables (7)
 - > tb_cars
 - > tb_fuel_price
 - > tb_gas_station
 - > tb_invoice
 - > tb_lines_invoice
 - > tb_pump
 - > tb_refueling
 - > Trigger Functions
 - > Types

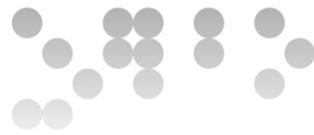
Query Query History

```
32
33 --b)
34 --function:
35 CREATE OR REPLACE FUNCTION fn_get_cost_by_car_type(initial_date DATE, final_date DATE, car_function CHARACTER(20))
36 RETURNS TABLE (cars_function CHARACTER(20), rf_date DATE, rf_liters INT, rf_cost NUMERIC)
37 LANGUAGE plpgsql
38 AS $$
39 BEGIN
40 IF length(car_function) > 20 THEN
41     RAISE EXCEPTION 'Longitud excesiva del valor introducido para el tipo de vehículo';
42 END IF;
43
44 RETURN QUERY
45 SELECT car.cars_function, refu.rf_date, refu.rf_liters, refu.rf_cost
46 FROM erp.tb_refueling refu
47 INNER JOIN erp.tb_cars car
48 ON car.cars_registration = refu.cars_registration
49 WHERE car.cars_function = car_function AND refu.rf_date BETWEEN initial_date AND final_date;
50 END; $$|
```

Create Function

Query returned successfully in 47 msec.

Total rows: 0 of 0 Query complete 00:00:00.047 Ln 50, Col 8



Podemos hacer una consulta buena y otra que debería dar error para comprobar que funciona:

- Consulta buena:

Screenshot of PgAdmin showing a successful query execution.

Query:

```

46   FROM erp.tb_refueling refu
47   INNER JOIN erp.tb_cars car
48   ON car.cars_registration = refu.cars_registration
49   WHERE car.cars_function = car_function AND refu.rf_date BETWEEN initial_date AND final_date;
50   END; $$

52 -- consulta
53   SELECT * FROM fn_get_cost_by_car_type('2022-08-01', '2022-08-015', 'reparto')

```

Data Output:

car_function	rf_date	rf_liters	rf_cost
reparto	2022-08-03	60	[null]
reparto	2022-08-12	42	[null]
reparto	2022-08-15	34	[null]
reparto	2022-08-10	60	[null]
reparto	2022-08-15	33	[null]
reparto	2022-08-05	40	[null]
reparto	2022-08-12	39	[null]
reparto	2022-08-15	25	[null]
reparto	2022-08-09	35	[null]
reparto	2022-08-01	44	[null]
reparto	2022-08-14	42	[null]
reparto	2022-08-02	41	[null]
reparto	2022-08-05	42	[null]
reparto	2022-08-08	41	[null]
reparto	2022-08-11	39	[null]
reparto	2022-08-01	47	40 AÑOS

Total rows: 26 of 26 Query complete 00:00:00.052 Ln 53, Col 1

- Consulta con error:

Screenshot of PgAdmin showing an error query execution.

Query:

```

31
32
33 --b)
34 --function:
35 CREATE OR REPLACE FUNCTION fn_get_cost_by_car_type(initial_date DATE, final_date DATE, car_function CHARACTER(20))
36 RETURNS TABLE (cars_function CHARACTER(20), rf_date DATE, rf_liters INT, rf_cost NUMERIC)
37 language plpgsql
38 AS $$
39 BEGIN
40 IF length(car_function) > 20 THEN
41   RAISE EXCEPTION 'Longitud excesiva del valor introducido para el tipo de vehículo';
42 END IF;
43
44 RETURN QUERY
45   SELECT car.cars_function, refu.rf_date, refu.rf_liters, refu.rf_cost
46   FROM erp.tb_refueling refu
47   INNER JOIN erp.tb_cars car
48   ON car.cars_registration = refu.cars_registration
49   WHERE car.cars_function = car_function AND refu.rf_date BETWEEN initial_date AND final_date;
50 END; $$

52 -- consulta
53   SELECT * FROM fn_get_cost_by_car_type('2022-08-01', '2022-08-015', 'reparto')
54 -- error
55   SELECT * FROM fn_get_cost_by_car_type('2022-08-01', '2022-08-015', 'vehiculos destinados al reparto de mercancías')
56
57

```

Messages:

ERROR: Longitud excesiva del valor introducido para el tipo de vehículo
CONTEXT: PL/pgSQL function fn_get_cost_by_car_type(date,date,charater) line 4 at RAISE
SQL state: P0001

Total rows: 26 of 26 Query complete 00:00:00.095 Ln 55, Col 1



c) Tal y como nos indica el ejercicio creamos el procedimiento y la tabla

Screenshot of pgAdmin 4 showing the creation of a new table and its data insertion.

```

CREATE TABLE erp.tb_invoice_cost_summary (
    cars_registration CHARACTER(7) NOT NULL,
    cars_fuel CHARACTER(10) NOT NULL,
    invoice_year INTEGER NOT NULL,
    invoice_quarter INTEGER NOT NULL,
    invoice_month VARCHAR(7) NOT NULL,
    total_liters NUMERIC NOT NULL,
    total_cost NUMERIC NOT NULL,
    number_of_lines INTEGER NOT NULL,
    timestamp TIMESTAMP NOT NULL
);

DO $$$
DECLARE
    row_record record;
BEGIN
    FOR row_record IN
        SELECT tb_cars.cars_registration, tb_cars.cars_fuel, EXTRACT(YEAR FROM tb_invoice.inv_date_start) AS invoice_year, EXTRACT(QUARTER FROM tb_invoice.inv_date_start) AS invoice_quarter, EXTRACT(MONTH FROM tb_invoice.inv_date_start) AS invoice_month, tb_lines.invoice.cars_registration, tb_lines.invoice.inv_id
        FROM tb_cars
        INNER JOIN tb_lines_invoice ON tb_cars.cars_registration = tb_lines.invoice.cars_registration
        INNER JOIN tb_invoice ON tb_invoice.inv_id = tb_lines.invoice.inv_id
    LOOP
        INSERT INTO erp.tb_invoice_cost_summary(cars_registration, cars_fuel, invoice_year, invoice_quarter, invoice_month, total_liters, total_cost, number_of_lines)
        VALUES (row_record.cars_registration, row_record.cars_fuel, row_record.invoice_year, row_record.invoice_quarter, row_record.invoice_month, row_record.total_liters, row_record.total_cost, row_record.number_of_lines);
    END LOOP;
END $$;

```

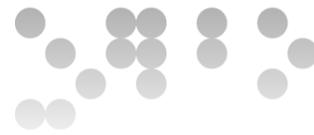
Total rows: 26 of 26 Query complete 00:00:00.049

Podemos ver la nueva tabla que se crea para comprobar que funciona

Screenshot of pgAdmin 4 showing the results of a query against the newly created table.

	cars_registration	cars_fuel	invoice_year	invoice_quarter	invoice_month	total_liters	total_cost	number_of_lines	timestamp
1	3685HUP	gasoil	2022	3	2022-8	188	373.93	2	2022-12-19 10:25:11.035989
2	0815GYR	gasoil	2022	3	2022-8	368	735.96	6	2022-12-19 10:25:11.035989
3	0815GYR	gasoil	2022	4	2022-10	616	1230.84	5	2022-12-19 10:25:11.035989
4	4273GFK	gasoil	2022	3	2022-9	291	582.07	5	2022-12-19 10:25:11.035989
5	0815GYR	gasoil	2022	3	2022-9	308	615.42	5	2022-12-19 10:25:11.035989
6	0019GVM	gasoil	2022	4	2022-10	301	601.43	5	2022-12-19 10:25:11.035989
7	9421GMT	gasoil	2022	3	2022-8	431	856.07	5	2022-12-19 10:25:11.035989
8	7045KDM	gasoil	2022	3	2022-8	64	125.53	2	2022-12-19 10:25:11.035989
9	3685HDP	gasoil	2022	3	2022-8	188	373.93	2	2022-12-19 10:25:11.035989
10	8806KZN	gasolina	2022	3	2022-8	108	235.58	3	2022-12-19 10:25:11.035989
11	2093GSW	gasoli	2022	4	2022-10	394	787.19	5	2022-12-19 10:25:11.035989
12	5649JSN	gasolina	2022	4	2022-10	412	876.92	5	2022-12-19 10:25:11.035989
13	6392KPT	gasoli	2022	4	2022-10	332	663.66	4	2022-12-19 10:25:11.035989
14	3685HDP	gasoli	2022	4	2022-10	188	373.93	2	2022-12-19 10:25:11.035989
15	5649JSN	gasolina	2022	3	2022-9	264	556.11	5	2022-12-19 10:25:11.035989
16	4273GFK	gasoli	2022	4	2022-10	478	964.19	5	2022-12-19 10:25:11.035989
17	2093GSW	gasoli	2022	3	2022-9	394	787.19	5	2022-12-19 10:25:11.035989
18	4273GFK	gasoli	2022	3	2022-8	291	582.07	5	2022-12-19 10:25:11.035989
19	2093GSW	gasoli	2022	3	2022-8	394	787.19	5	2022-12-19 10:25:11.035989
20	6392KPT	gasoli	2022	3	2022-9	166	331.83	4	2022-12-19 10:25:11.035989

Total rows: 24 of 24 Query complete 00:00:00.062



EJERCICIO 3 (10%)

- a) Para saber cuántos triggers tenemos definidos podemos utilizar la siguiente función:

```
SELECT * FROM information_schema.triggers WHERE event_object_schema = 'erp';
```

Este Código nos muestra los triggers que tenemos definidos en el momento en el que lo ejecutamos. Estos son los triggers que yo tengo definidos en mi base de datos:

The screenshot shows the pgAdmin 4 interface with the following details:

- Servers:** PostgreSQL 15, dbdw_pec3
- Browser:** Shows 'Servers (1)', 'Databases (3)', 'Casts', 'Catalogs', 'Event Triggers', 'Extensions', 'Foreign Data Wrappers', 'Languages', 'Publications', 'Schemas (2)' (selected), 'Aggregates', 'Collations', 'Domains', 'FTS Configurations', 'FTS Dictionaries', 'FTS Parsers', 'FTS Templates', 'Foreign Tables', 'Functions', 'Materialized Views', 'Operators', 'Procedures', 'Sequences', and 'Tables (8)'.
- Query Editor:** Contains the SQL query: `--a)
SELECT * FROM information_schema.triggers WHERE event_object_schema = 'erp';`
- Data Output:** A table showing the results of the query. The columns are: trigger_catalog, trigger_schema, trigger_name, event_manipulation, event_object_catalog, event_object_schema, event_object_table, action_order, and action_condition. The data is as follows:

trigger_catalog	trigger_schema	trigger_name	event_manipulation	event_object_catalog	event_object_schema	event_object_table	action_order	action_condition
dbdw_pec3	erp	tg_line_inserted	INSERT	dbdw_pec3	erp	tb_lines_invoice	1	[null]
dbdw_pec3	erp	tr_update_inv_updated...	INSERT	dbdw_pec3	erp	tb_lines_invoice	2	[null]
dbdw_pec3	erp	tg_line_inserted	UPDATE	dbdw_pec3	erp	tb_lines_invoice	1	[null]
dbdw_pec3	erp	tr_update_inv_updated...	UPDATE	dbdw_pec3	erp	tb_lines_invoice	2	[null]
dbdw_pec3		update_timestamp	INSERT	dbdw_pec3	erp	tb_pump	1	[null]
dbdw_pec3	erp	tg_refueling_cost	INSERT	dbdw_pec3	erp	tb_refueling	1	[null]

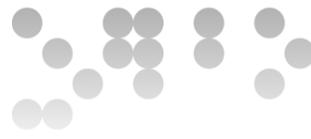
- b)

En PostgreSQL, una tabla interna es un tipo especial de tabla que no se almacena en el disco y no es accesible para los usuarios. Son utilizados internamente por el servidor PostgreSQL para almacenar datos temporales o resultados intermedios.

Una vista, por otro lado, es una tabla virtual que está definida por una instrucción SELECT. Una vista no almacena datos en sí misma, sino que recupera datos de una o más tablas o vistas subyacentes cuando se consulta. Las vistas se utilizan a menudo para simplificar consultas complejas dividiéndolas en partes más pequeñas y manejables, o para proporcionar una vista personalizada de los datos para diferentes usuarios.

Las tablas internas se utilizan para el almacenamiento temporal y el procesamiento de datos, mientras que las vistas se utilizan para presentar los datos a los usuarios de forma personalizada o simplificada.

A continuación detallo un par de ejemplos de tablas y consultas que pueden ser útiles cuando se trabaja con el “Statistics Collector”:



1. Para ver información estadística sobre las tablas en una base de datos específica, incluida la cantidad de filas en cada tabla y la cantidad de espacio en disco utilizado por cada tabla, se puede usar la siguiente consulta:

```
SELECT * FROM pg_stat_user_tables WHERE schemaname = 'nombre_esquema';
```

En mi base de datos se vería así:

relid	oid	name	schemaname	relname	seq_scan	seq_tup_read	idx_scan	idx_tup_fetch	n_tup_ins	n_tup_upd	n_tup_del	n_tup_hot_upd	n_live_tup
1	33341	erp	erp	tb_invoice	1	0	120	120	3	12	0	0	12
2	33341	erp	erp	tb_invoice_cost_summary	1	24	[null]	[null]	24	0	0	0	0
3	33327	erp	erp	tb_refueling	3	342	[null]	[null]	114	0	0	0	0
4	33310	erp	erp	tb_pump	10	282	164	164	59	2	0	0	2
5	33291	erp	erp	tb_cars	2	10	230	230	10	0	0	0	0
6	33305	erp	erp	tb_fuel_price	2	122	0	0	122	0	0	0	0
7	33298	erp	erp	tb_gas_station	2	0	165	165	5	0	0	0	0
8	33346	erp	erp	tb_lines_invoice	12	1155	0	0	105	3	0	0	3

2. Para ver la actividad actual en el sistema de la base de datos, incluido el PID (ID de proceso) de cada proceso backend activo, el usuario con el que está conectado y la consulta actual que ejecuta cada proceso, se puede usar la siguiente consulta:

```
SELECT * FROM pg_stat_activity;
```

En mi base de datos se vería así:

datid	oid	datname	pid	leader_pid	usessoid	username	application_name	client_addr	client_hostname	client_port	backend_start	
1	[null]	[null]	366	[null]	[null]	[null]	[null]	[null]	[null]	[null]	2022-12-18 11:01:47.609	
2	[null]	[null]	368	[null]	10	postgres	pgAdmin 4 - DB postgres	[null]	[null]	[null]	2022-12-18 11:01:47.610	
3	5	postgres	3909	[null]	10	postgres	pgAdmin 4 - DB postgres	127.0.0.1	[null]	[null]	54054	2022-12-18 17:43:33.233
4	33289	dbdw_pec3	3931	[null]	10	postgres	pgAdmin 4 - DB dbdw_pec3	127.0.0.1	[null]	[null]	54101	2022-12-18 17:43:44.771
5	33289	dbdw_pec3	3935	[null]	10	postgres	pgAdmin 4 - CONN41434...	127.0.0.1	[null]	[null]	54137	2022-12-18 17:44:54.268
6	24993	pec3	3912	[null]	10	postgres	pgAdmin 4 - DB pec3	127.0.0.1	[null]	[null]	54089	2022-12-18 17:43:41.708
7	33289	dbdw_pec3	4522	[null]	10	postgres	pgAdmin 4 - CONN44739...	127.0.0.1	[null]	[null]	54242	2022-12-18 17:45:58.339
8	33289	dbdw_pec3	4541	[null]	10	postgres	pgAdmin 4 - CONN15154...	127.0.0.1	[null]	[null]	54285	2022-12-18 17:48:55.794
9	33289	dbdw_pec3	4570	[null]	10	postgres	pgAdmin 4 - CONN47791...	127.0.0.1	[null]	[null]	54354	2022-12-18 17:50:29.486
10	33289	dbdw_pec3	4691	[null]	10	postgres	pgAdmin 4 - CONN68909...	127.0.0.1	[null]	[null]	54711	2022-12-18 17:59:49.551
11	33289	dbdw_pec3	4802	[null]	10	postgres	pgAdmin 4 - CONN38620...	127.0.0.1	[null]	[null]	55191	2022-12-18 18:00:46.787
12	33289	dbdw_pec3	5634	[null]	10	postgres	pgAdmin 4 - CONN33416...	127.0.0.1	[null]	[null]	58106	2022-12-18 18:55:02.103
13	33289	dbdw_pec3	5845	[null]	10	postgres	pgAdmin 4 - CONN54545...	127.0.0.1	[null]	[null]	59171	2022-12-18 19:09:30.684
14	33289	dbdw_pec3	11600	[null]	10	postgres	pgAdmin 4 - CONN42153...	127.0.0.1	[null]	[null]	64988	2022-12-19 10:14:00.475