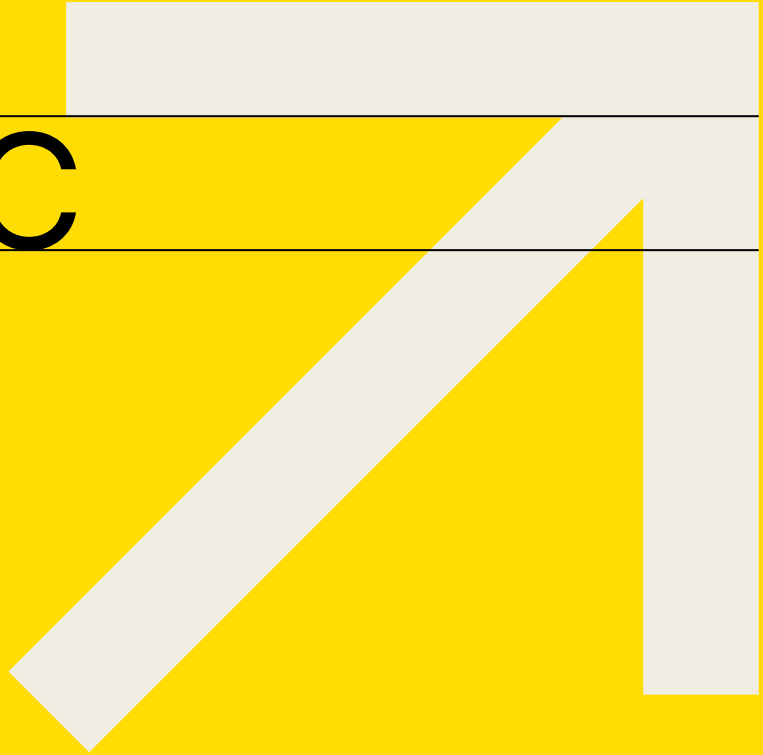
ATELIER STORED PROC



```
DROP TABLE public.journal;
CREATE TABLE public.journal (
    id serial NOT NULL,
    moment timestamp with time zone,
    operation text NOT NULL,
    description text,
    objet text,
    CONSTRAINT journal_pkey PRIMARY KEY (id)
);

ALTER TABLE public.journal OWNER TO postgres;
```

```
CREATE or REPLACE FUNCTION journaliser()
RETURNS trigger
LANGUAGE 'plpgsql'
AS $$
BEGIN
   INSERT into journal(operation) VALUES('test');
   return NEW;
END
$$
;
```

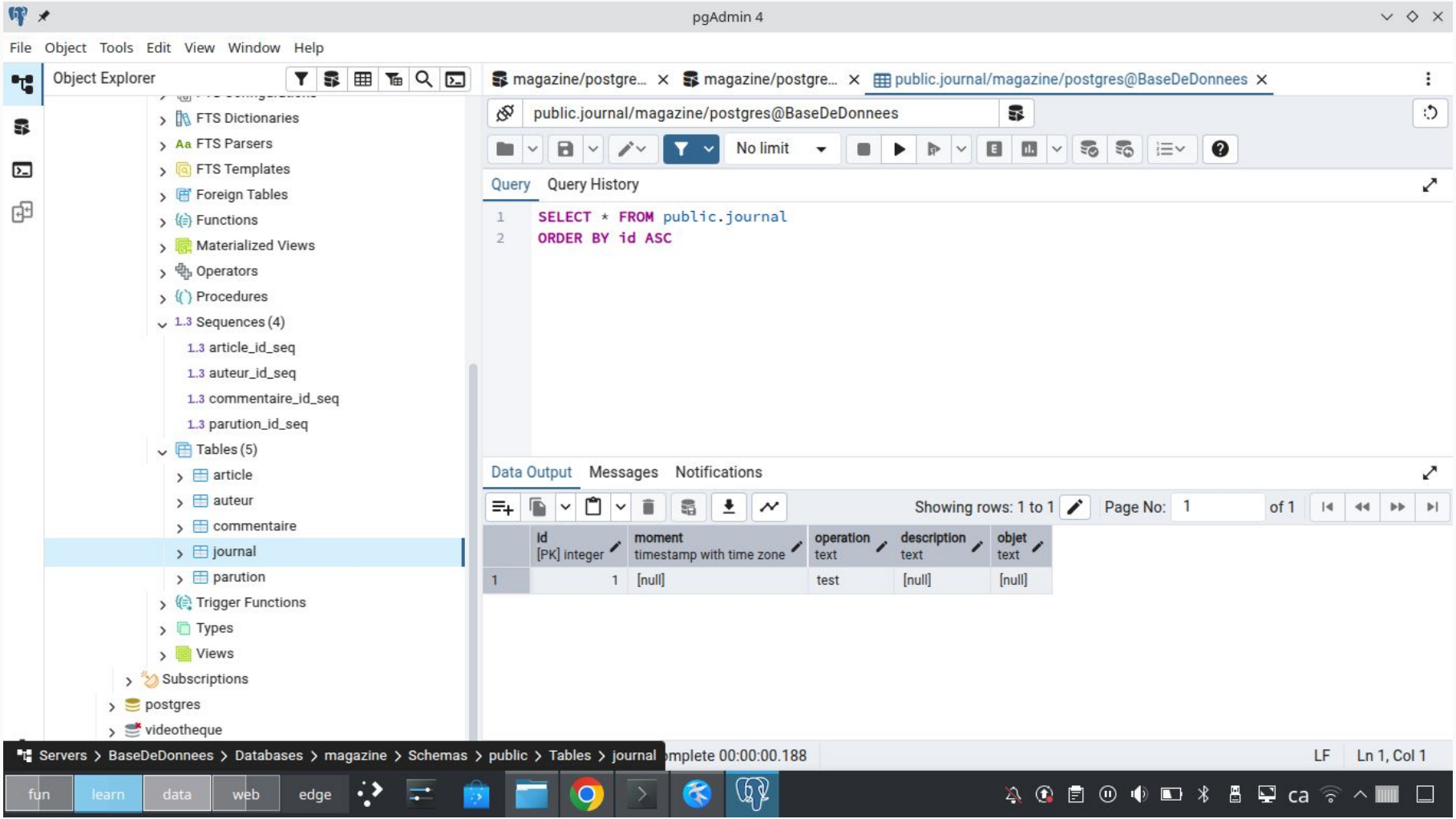
```
CREATE TRIGGER evenementAjoutArticle BEFORE INSERT ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
INSERT INTO public.article (titre, contenu, image, date_soumission,
date_modification, approuver, id_parution, id_auteur, id_editeur)
VALUES ('Soupes et veloutés : champignons', 'Des potages simples mais pleins
de caractère, à base de légumes de saison et d'un soupçon de créativité. De
quoi transformer chaque bol en moment cocooning.', 'soupes_hiver.jpg',
'2026-01-02', '2025-12-24', false, 3, 4, 1);
```

```
SELECT * from journal
```

Showing rows: 1 to 1

	id [PK] integer	moment timestamp with time zone	operation text	description text	objet text
1	1	[null]	test	[null]	[null]


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences (4)
article_id_seq
auteur_id_seq
commentaire_id_seq
parution_id_seq
Tables (5)
article
auteur
commentaire
journal
parution
Trigger Functions
Types
Views
Subscriptions
postgres
videotheque
magazine/postgre...
magazine/postgre...
public.journal/magazine/postgres@BaseDeDonnees
public.journal/magazine/postgres@BaseDeDonnees
No limit
Query
Query History
1 SELECT * FROM public.journal
2 ORDER BY id ASC
Data Output
Messages
Notifications
Showing rows: 1 to 1
Page No: 1
of 1
id [PK] integer
moment timestamp with time zone
operation text
description text
objet text
1 1 [null] test [null] [null]
Servers > BaseDeDonnees > Databases > magazine > Schemas > public > Tables > journal
mplete 00:00:00.188
LF
Ln 1, Col 1
fun
learn
data
web
edge
ca

```
CREATE or REPLACE FUNCTION journaliser()
RETURNS trigger
LANGUAGE 'plpgsql'

AS $$
BEGIN
        IF TG_OP = 'INSERT' THEN
                INSERT into journal(operation) VALUES('ajout');
        END IF;
    return NEW;
END
$$
;
```

```
CREATE TRIGGER evenementAjoutArticle BEFORE INSERT ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
SELECT * from journal
```

Showing rows: 1 to 2

	id [PK] integer	moment timestamp with time zone	operation text	description text	objet text
1	1	[null]	test	[null]	[null]
2	2	[null]	ajout	[null]	[null]

MARCHE (ajout dans journal)

```
INSERT INTO public.article (titre, contenu
date_modification, approuver, id_parution,
VALUES ('Soupes et veloutés : champignons', 'Des
de caractère, à base de légumes de saison et d'u
quoi transformer chaque bol en moment cocooning.', 'soupes_hiver.jpg',
'2026-01-02', '2025-12-24', false, 3, 4, 1);
```

NE FAIT RIEN dans journal

```
UPDATE article SET titre = 'truc' WHERE id = 1
```

```
CREATE or REPLACE FUNCTION journaliser()
RETURNS trigger
LANGUAGE 'plpgsql'

AS $$
BEGIN
        IF TG_OP = 'INSERT' THEN
                INSERT into journal(operation) VALUES('AJOUTER');
        END IF;
        IF TG_OP = 'UPDATE' THEN
                INSERT into journal(operation) VALUES('M
        END IF;
    return NEW;
END
$$
;
```

```
CREATE TRIGGER evenementAjoutArticle BEFORE INSERT ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
CREATE TRIGGER evenementChangementArticle BEFORE UPDATE ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

MARCHE

```
INSERT INTO public.article (titre, contenu, image, date_
date_modification, approuver, id_parution, id_auteur, id_
VALUES ('Soupes et veloutés : champignons', 'Des potages simples mais pleins
de caractère, à base de légumes de saison et d'un soupçon de créativité. De
quoi transformer chaque bol en moment cocooning.', 'soupes_hiver.jpg',
'2026-01-02', '2025-12-24', false, 3, 4, 1);
```

```
SELECT * from journal
```

Data Output | Messages | Notifications

Showing rows: 1 to 4

	id [PK] integer	moment timestamp with time zone	operation text	description text	objet text
1	3	[null]	AJOUTER	[null]	[null]
2	4	[null]	MODIFIER		

MARCHE

```
UPDATE article SET titre = 'truc' WHERE id = 1
```

```
CREATE or REPLACE FUNCTION journaliser()
RETURNS trigger
LANGUAGE 'plpgsql'

AS $$
BEGIN
        IF TG_OP = 'INSERT' THEN
                INSERT into journal(operation) VALUES('AJOUTER');
        END IF;
        IF TG_OP = 'UPDATE' THEN
                INSERT into journal(operation) VALUES('MODIFIER');
        END IF;
        IF TG_OP = 'DELETE' THEN
                INSERT into journal(operation) VALUES('EFFACER');
        END IF;
    return NEW;
END
$$
;
```

```
CREATE TRIGGER evenementAjoutArticle BEFORE INSERT ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
CREATE TRIGGER evenementChangementArticle BEFORE UPDATE ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
CREATE TRIGGER evenementEffacementArticle BEFORE DELETE ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
INSERT INTO article ....
```

MARCHE

```
UPDATE article SET titre = 'truc' WHERE id = 1
```

MARCHE

```
DELETE FROM article WHERE id = 21
```

MARCHE

```
SELECT * from journal
```

	id [PK] integer	moment timestamp with time zone	operation text	description text	objet text
1	7	[null]	AJOUTER	[null]	[null]
2	8	[null]	MODIFIER	[null]	[null]
3	9	[null]	EFFACER	[null]	[null]

```
CREATE or REPLACE FUNCTION journaliser()
RETURNS trigger LANGUAGE 'plpgsql'

AS $$
DECLARE
	operationJournaliste text;
BEGIN
	IF TG_OP = 'INSERT' THEN
		operationJournaliste := 'AJOUTER';
	END IF;
	IF TG_OP = 'UPDATE' THEN
		operationJournaliste := 'MODIFIER';
	END IF;
	IF TG_OP = 'DELETE' THEN
		operationJournaliste := 'EFFACER';
	END IF;
	INSERT into journal(operation) VALUES(operationJournaliste);
   return NEW;
END
$$
;
```

```
CREATE TRIGGER evenementAjoutArticle BEFORE INSERT ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
CREATE TRIGGER evenementChangementArticle BEFORE UPDATE ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
CREATE TRIGGER evenementEffacementArticle BEFORE DELETE ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
INSERT INTO article ....
```

```
UPDATE article SET titre = 'truc' WHERE id = 1
```

```
DELETE FROM article WHERE id = 21
```

```
SELECT * from journal
```

Showing rows: 1 to

	id [PK] integer	moment timestamp with time zone	operation text	description text	objet
1	7	[null]	AJOUTER	[null]	
2	8	[null]	MODIFIER	[null]	

MARCH

```
CREATE or REPLACE FUNCTION journaliser()
RETURNS trigger
LANGUAGE 'plpgsql'

AS $$
DECLARE
	operationJournaliste text;
	objetAvant text;
	objetApres text;
	description text;
BEGIN
	objetAvant = '{}';
	objetApres = '{}';
	IF TG_OP = 'INSERT' THEN
		operationJournaliste := 'AJOUTER';
		objetApres := '{' || NEW.titre || '}';
	END IF;
	IF TG_OP = 'UPDATE' THEN
		operationJournaliste := 'MODIFIER';
		objetApres := '{' || NEW.titre || '}';
	END IF;
```

```
CREATE TRIGGER evenementAjoutArticle BEFORE INSERT ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
CREATE TRIGGER evenementChangementArticle BEFORE UPDATE ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
CREATE TRIGGER evenementEffacementArticle BEFORE DELETE ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
INSERT INTO article ....
```

```
UPDATE article SET titre = 'truc' WHERE id = 1
```

```
DELETE FROM article WHERE id = 21
```

MARC

```
SELECT * from journal
```

Showing rows: 1 to 7 Page No: 1

	id [PK] integer	moment timestamp with time zone	operation text	description text	objet text
1	22	[null]	AJOUTER	{} => {Soupes et veloutés : chaleur dans le ...	[null]
2	23	[null]	MODIFIER	{} => {truc}	[null]
3	24	[null]	EFFACER	{} => {}	[null]

```
CREATE or REPLACE FUNCTION journaliser()
RETURNS trigger
LANGUAGE 'plpgsql'

AS $$
DECLARE
        operationJournaliste text;
        objetAvant text;
        objetApres text;
        description text;
BEGIN
        objetAvant := '{}';
        objetApres := '{}';
        IF TG_OP = 'INSERT' THEN
                operationJournaliste := 'AJOUTER';
                objetApres := '{' || NEW.titre || ',' || NEW.contenu || ',' || N
        END IF;
        IF TG_OP = 'UPDATE' THEN
                operationJournaliste := 'MODIFIER';
                objetApres := '{' || NEW.titre || ',' || NEW.contenu || ',' || NEW.date_soumission || '}';
                objetAvant := '{' || OLD.titre || ',' || OLD.contenu || ',' || OLD.date_soumission || '}';
        END IF;
        IF TG_OP = 'DELETE' THEN
                operationJournaliste := 'EFFACER';
                objetAvant := '{' || OLD.titre || ',' || OLD.contenu || ',' || OLD.da
        END IF;
        description = objetAvant || ' => ' || objetApres;
        INSERT into journal(operation, description) VALUES(operationJourna
        IF TG_OP = 'DELETE' THEN
            return OLD;
```

```
CREATE TRIGGER evenementAjoutArticle BEFORE INSERT ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
CREATE TRIGGER evenementChangementArticle BEFORE UPDATE ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
CREATE TRIGGER evenementEffacementArticle BEFORE DELETE ON public.article
FOR EACH ROW EXECUTE PROCEDURE public.journaliser();
```

```
INSERT INTO article ....
```

```
UPDATE article SET titre = 'truc' WHERE id = 1
```

```
DELETE FROM article WHERE id = 21
```

MARC

```
SELECT * from journal
```

Showing rows: 1 to 7 Page No: 1

	id [PK] integer	moment timestamp with time zone	operation text	description text	objet text
1	22	[null]	AJOUTER	{} => {Soupes et veloutés : chaleur dans le ...	[null]
2	23	[null]	MODIFIER	{} => {truc}	[null]
3	24	[null]	EFFACER	{} => {}	[null]