CORRECTION EXERCICEs CH4

Exercice 1

- **Korentin Georget**

```
SELECT
        indexname AS nom_de_l_index,
        tablename AS table_concernee,
        indexdef AS definition_complete
FROM
        pg_indexes
WHERE
        schemaname = 'public'
ORDER BY
        tablename,
        indexname;
```

Exercice 2

- **Furfaro Thomas**

```
ANALYZE skippers;

SELECT

        relname AS table_name,

        reltuples AS number_of_rows,

        relpages AS number_of_blocks

FROM pg_class

WHERE relname = 'skippers';
```

Idem pour les autres tables

Clément Lorieau

Pour BDCroisieresBig

```
ANALYZE;

SELECT relname AS table_name,
    reltuples::bigint AS nb_lignes,
    relpages  AS nb_blocs
FROM pg_class
WHERE relname IN ('skippers', 'croisieres', 'bateaux')
ORDER BY relname;
```

Exercice 3

- **SANNA Thomas**

(((PETITE BDD)))

```
EXPLAIN ANALYZE SELECT sknom FROM skippers ORDER BY salaire;

--"Sort  (cost=23.60..24.28 rows=270 width=122) (actual time=0.018..0.019 rows=8 loops=1)"

--"  Sort Key: salaire"

--"  Sort Method: quicksort  Memory: 25kB"

--"  ->  Seq Scan on skippers  (cost=0.00..12.70 rows=270 width=122) (actual time=0.005..0.006 rows=8 loops=1)"

--"Planning Time: 0.114 ms"

--"Execution Time: 0.030 ms"


EXPLAIN ANALYZE SELECT MIN(salaire) FROM skippers;


--"Aggregate  (cost=13.38..13.38 rows=1 width=4) (actual time=0.012..0.013 rows=1 loops=1)"

--"  ->  Seq Scan on skippers  (cost=0.00..12.70 rows=270 width=4) (actual time=0.006..0.007 rows=8 loops=1)"

--"Planning Time: 0.065 ms"

--"Execution Time: 0.027 ms"


EXPLAIN ANALYZE SELECT DISTINCT SALAIRE FROM SKIPPERS;
```

```
--"HashAggregate  (cost=13.38..15.38 rows=200 width=4) (actual time=0.022..0.024 rows=4 loops=1)"
--"  Group Key: salaire"
--"  Batches: 1  Memory Usage: 40kB"
--"  ->  Seq Scan on skippers  (cost=0.00..12.70 rows=270 width=4) (actual time=0.012..0.013 rows=8 loops=1)"
--"Planning Time: 0.056 ms"
--"Execution Time: 0.060 ms"

CREATE INDEX idx_salaire ON skippers (salaire);

EXPLAIN ANALYZE SELECT sknom FROM skippers ORDER BY salaire;

--"Sort  (cost=1.20..1.22 rows=8 width=122) (actual time=0.016..0.017 rows=8 loops=1)"
--"  Sort Key: salaire"
--"  Sort Method: quicksort  Memory: 25kB"
--"  ->  Seq Scan on skippers  (cost=0.00..1.08 rows=8 width=122) (actual time=0.008..0.009 rows=8 loops=1)"
--"Planning Time: 0.165 ms"
--"Execution Time: 0.028 ms"

EXPLAIN ANALYZE SELECT MIN(salaire) FROM skippers;

--"Aggregate  (cost=1.10..1.11 rows=1 width=4) (actual time=0.017..0.018 rows=1 loops=1)"
--"  ->  Seq Scan on skippers  (cost=0.00..1.08 rows=8 width=4) (actual time=0.011..0.012 rows=8 loops=1)"
--"Planning Time: 0.091 ms"
--"Execution Time: 0.129 ms"

EXPLAIN ANALYZE SELECT DISTINCT salaire FROM skippers;

--"HashAggregate  (cost=1.10..1.18 rows=8 width=4) (actual time=0.017..0.018 rows=4 loops=1)"
--"  Group Key: salaire"
--"  Batches: 1  Memory Usage: 24kB"
--"  ->  Seq Scan on skippers  (cost=0.00..1.08 rows=8 width=4) (actual time=0.009..0.010 rows=8 loops=1)"
--"Planning Time: 0.053 ms"
--"Execution Time: 0.035 ms"
```

(((GROSSE BDD)))

```
EXPLAIN ANALYZE SELECT sknom FROM skippers ORDER BY salaire;
```

```
--"Sort  (cost=67.83..70.33 rows=1000 width=17) (actual time=0.493..0.545 rows=1000 loops=1)"

--"  Sort Key: salaire"

--"  Sort Method: quicksort  Memory: 64kB"

--"  ->  Seq Scan on skippers  (cost=0.00..18.00 rows=1000 width=17) (actual time=0.008..0.125 rows=1000 loops=1)"

--"Planning Time: 0.173 ms"

--"Execution Time: 0.625 ms"


EXPLAIN ANALYZE SELECT MIN(salaire) FROM skippers;


--"Aggregate  (cost=20.50..20.51 rows=1 width=32) (actual time=0.335..0.336 rows=1 loops=1)"

--"  ->  Seq Scan on skippers  (cost=0.00..18.00 rows=1000 width=6) (actual time=0.012..0.136 rows=1000 loops=1)"

--"Planning Time: 0.160 ms"

--"Execution Time: 0.394 ms"


EXPLAIN ANALYZE SELECT DISTINCT SALAIRE FROM SKIPPERS;


--"HashAggregate  (cost=20.50..30.49 rows=999 width=6) (actual time=0.240..0.324 rows=999 loops=1)"

--"  Group Key: salaire"

--"  Batches: 1  Memory Usage: 129kB"

--"  ->  Seq Scan on skippers  (cost=0.00..18.00 rows=1000 width=6) (actual time=0.007..0.068 rows=1000 loops=1)"

--"Planning Time: 0.051 ms"

--"Execution Time: 0.403 ms"


CREATE INDEX idx_salaire ON skippers (salaire);


EXPLAIN ANALYZE SELECT sknom FROM skippers ORDER BY salaire;


--"Index Scan using idx_salaire on skippers  (cost=0.28..67.27 rows=1000 width=17) (actual time=0.028..0.355 rows=1000 loops=1)"

--"Planning Time: 0.201 ms"
```

```
--"Execution Time: 0.417 ms"

 EXPLAIN ANALYZE SELECT MIN(salaire) FROM skippers;

--"Result  (cost=0.34..0.35 rows=1 width=32) (actual time=0.049..0.050 rows=1 loops=1)"

--"  InitPlan 1"

--"    ->  Limit  (cost=0.28..0.34 rows=1 width=6) (actual time=0.045..0.046 rows=1 loops=1)"

--"          ->  Index Only Scan using idx_salaire on skippers  (cost=0.28..67.27 rows=1000 width=6) (actual time=0.044..0.044 rows=1
loops=1)"

--"                Heap Fetches: 1"

--"Planning Time: 0.087 ms"

--"Execution Time: 0.066 ms"



EXPLAIN ANALYZE SELECT DISTINCT salaire FROM skippers;



--"HashAggregate  (cost=20.50..30.49 rows=999 width=6) (actual time=0.546..0.731 rows=999 loops=1)"

--"  Group Key: salaire"

--"  Batches: 1  Memory Usage: 129kB"

--"  ->  Seq Scan on skippers  (cost=0.00..18.00 rows=1000 width=6) (actual time=0.017..0.148 rows=1000 loops=1)"

--"Planning Time: 0.092 ms"

--"Execution Time: 0.837 ms"
```

EXERCICE 8

1. Sur croisieresBig

- Solution avec not exist
- Solution avec IN
- Solution avec Group BY Having

Fabbri Yohann

EXPLAIN ANALYSE

```
SELECT DISTINCT SKNOM

FROM CROISIERES NATURAL JOIN SKIPPERS

WHERE BATNUM IN (

SELECT BATNUM

FROM BATEAUX

WHERE CAPACITE <= (

SELECT MIN(CAPACITE)

      FROM BATEAUX

      WHERE BATPORT = 'AJACCIO'

   )

);
```

```
"HashAggregate  (cost=193.24..203.24 rows=1000 width=11) (actual time=0.462..0.463
rows=0 loops=1)"

"  Group Key: skippers.sknom"

"  Batches: 1  Memory Usage: 73kB"

"  InitPlan 1"

"    ->  Aggregate  (cost=21.33..21.34 rows=1 width=4) (actual time=0.112..0.112 rows=1
loops=1)"

"          ->  Seq Scan on bateaux bateaux_1  (cost=0.00..20.50 rows=333 width=4) (actual
time=0.003..0.092 rows=333 loops=1)"

"                Filter: ((batport)::text = 'AJACCIO'::text)"

"                Rows Removed by Filter: 667"

"  ->  Hash Join  (cost=55.16..167.74 rows=1665 width=11) (actual time=0.457..0.457
rows=0 loops=1)"

"        Hash Cond: (croisieres.sknum = skippers.sknum)"
"      ->  Hash Join  (cost=24.66..132.84 rows=1665 width=4) (actual time=0.194..0.194 rows=0 loops=1)"

"            Hash Cond: ((croisieres.batnum)::text = (bateaux.batnum)::text)"

"            ->  Seq Scan on croisieres  (cost=0.00..95.00 rows=5000 width=10) (actual time=0.003..0.003 rows=1 loops=1)"

"            ->  Hash  (cost=20.50..20.50 rows=333 width=6) (actual time=0.189..0.190 rows=0 loops=1)"

"                  Buckets: 1024  Batches: 1  Memory Usage: 8kB"

"                  ->  Seq Scan on bateaux  (cost=0.00..20.50 rows=333 width=6) (actual time=0.189..0.189 rows=0 loops=1)"

"                        Filter: (capacite < (InitPlan 1).col1)"

"                        Rows Removed by Filter: 1000"

"      ->  Hash  (cost=18.00..18.00 rows=1000 width=15) (actual time=0.259..0.259 rows=1000 loops=1)"

"            Buckets: 1024  Batches: 1  Memory Usage: 55kB"

"            ->  Seq Scan on skippers  (cost=0.00..18.00 rows=1000 width=15) (actual time=0.011..0.133 rows=1000 loops=1)"

"Planning Time: 0.363 ms"

"Execution Time: 0.501 ms"
```

```
EXPLAIN ANALYSE

SELECT DISTINCT s.SKNOM

FROM SKIPPERS s

JOIN CROISIERES c ON c.SKNUM = s.SKNUM

JOIN BATEAUX b ON b.BATNUM = c.BATNUM

WHERE NOT EXISTS (

  SELECT 1
  FROM BATEAUX a
  WHERE a.BATPORT = 'AJACCIO'
    AND a.CAPACITE <= b.CAPACITE
);
```

```
"HashAggregate  (cost=3584.01..3594.01 rows=1000 width=11) (actual time=1.453..1.455 rows=0 loops=1)"
"  Group Key: s.sknom"
"  Batches: 1  Memory Usage: 73kB"
"  ->  Hash Join  (cost=3419.80..3575.68 rows=3333 width=11) (actual time=1.447..1.449 rows=0 loops=1)"
"        Hash Cond: (c.sknum = s.sknum)"
"        ->  Hash Join  (cost=3389.30..3536.40 rows=3333 width=4) (actual time=1.207..1.209 rows=0 loops=1)"
"              Hash Cond: ((c.batnum)::text = (b.batnum)::text)"
"              ->  Seq Scan on croisieres c  (cost=0.00..95.00 rows=5000 width=10) (actual time=0.002..0.003 rows=1 loops=1)"
"              ->  Hash  (cost=3380.96..3380.96 rows=667 width=6) (actual time=1.202..1.203 rows=0 loops=1)"
"                    Buckets: 1024  Batches: 1  Memory Usage: 8kB"
"                    ->  Nested Loop Anti Join  (cost=0.00..3380.96 rows=667 width=6) (actual time=1.202..1.203 rows=0 loops=1)"
"                          Join Filter: (a.capacite <= b.capacite)"
"                          Rows Removed by Join Filter: 8227"
"                          ->  Seq Scan on bateaux b  (cost=0.00..18.00 rows=1000 width=10) (actual time=0.005..0.065 rows=1000 loops=1)"
"                    ->  Materialize  (cost=0.00..22.16 rows=333 width=4) (actual time=0.000..0.000 rows=9 loops=1000)"
"                          ->  Seq Scan on bateaux a  (cost=0.00..20.50 rows=333 width=4) (actual time=0.004..0.085 rows=259 loops=1)"
"                                Filter: ((batport)::text = 'AJACCIO'::text)"
"                                Rows Removed by Filter: 518"
"      ->  Hash  (cost=18.00..18.00 rows=1000 width=15) (actual time=0.237..0.237 rows=1000 loops=1)"
"            Buckets: 1024  Batches: 1  Memory Usage: 55kB"
"            ->  Seq Scan on skippers s  (cost=0.00..18.00 rows=1000 width=15) (actual time=0.009..0.104 rows=1000 loops=1)"
"Planning Time: 0.334 ms"
"Execution Time: 1.497 ms"
```

```
EXPLAIN ANALYSE
SELECT s.SKNOM
FROM SKIPPERS s
JOIN CROISIERES c ON c.SKNUM = s.SKNUM
JOIN BATEAUX b ON b.BATNUM = c.BATNUM
GROUP BY s.SKNOM
HAVING MIN(b.CAPACITE) <= (
  SELECT MIN(CAPACITE)
  FROM BATEAUX
  WHERE BATPORT = 'AJACCIO'
);
```

```
"HashAggregate  (cost=228.70..241.20 rows=333 width=11) (actual time=4.555..4.560
rows=0 loops=1)"

"  Group Key: s.sknom"

"  Filter: (min(b.capacite) < (InitPlan 1).col1)"

"  Batches: 1  Memory Usage: 129kB"

"  Rows Removed by Filter: 1000"

"  InitPlan 1"

"    ->  Aggregate  (cost=21.33..21.34 rows=1 width=4) (actual time=0.144..0.145 rows=1
loops=1)"

"          ->  Seq Scan on bateaux  (cost=0.00..20.50 rows=333 width=4) (actual
time=0.013..0.124 rows=333 loops=1)"

"                Filter: ((batport)::text = 'AJACCIO'::text)"

"                Rows Removed by Filter: 667"

"  ->  Hash Join  (cost=61.00..182.36 rows=5000 width=15) (actual time=0.568..3.289
rows=5000 loops=1)"

"        Hash Cond: ((c.batnum)::text = (b.batnum)::text)"

"        ->  Hash Join  (cost=30.50..138.68 rows=5000 width=17) (actual time=0.289..1.867
rows=5000 loops=1)"

"              Hash Cond: (c.sknum = s.sknum)"

"              ->  Seq Scan on croisieres c  (cost=0.00..95.00 rows=5000 width=10) (actual
time=0.002..0.417 rows=5000 loops=1)"

"              ->  Hash  (cost=18.00..18.00 rows=1000 width=15) (actual time=0.268..0.269
rows=1000 loops=1)"

"                    Buckets: 1024  Batches: 1  Memory Usage: 55kB"

"                    ->  Seq Scan on skippers s  (cost=0.00..18.00 rows=1000 width=15) (actual
time=0.006..0.120 rows=1000 loops=1)"

"        ->  Hash  (cost=18.00..18.00 rows=1000 width=10) (actual time=0.274..0.275
rows=1000 loops=1)"
```

```
"           Buckets: 1024  Batches: 1  Memory Usage: 50kB"
"           ->  Seq Scan on bateaux b  (cost=0.00..18.00 rows=1000 width=10) (actual time=0.011..0.135 rows=1000 loops=1)"
"Planning Time: 0.424 ms"
"Execution Time: 4.611 ms"
```

Si l’on rajoute un index sur batnum ou sknum les résultats restent inchangés.

Si l’on rajoute CREATE INDEX idx_bateaux_batport_capacite ON BATEAUX (BATPORT, CAPACITE); les résultats s’améliorent.

```
EXPLAIN ANALYSE
SELECT DISTINCT SKNOM
FROM CROISIERES
NATURAL JOIN SKIPPERS
WHERE BATNUM IN
( SELECT BATNUM FROM BATEAUX WHERE CAPACITE <=
( SELECT MIN(CAPACITE) FROM BATEAUX WHERE BATPORT = 'AJACCIO' ) );
```

```
"HashAggregate (cost=172.31..182.31 rows=1000 width=11) (actual time=4.597..4.621 rows=5 loops=1)" " Group Key: skippers.sknom" " Batches: 1 Memory Usage: 73kB" " InitPlan 2" " -> Result (cost=0.40..0.41 rows=1 width=4) (actual time=0.097..0.100 rows=1 loops=1)" " InitPlan 1" " -> Limit (cost=0.28..0.40 rows=1 width=4) (actual time=0.090..0.093 rows=1 loops=1)" " -> Index Only Scan using idx_bateaux_batport_capacite on bateaux bateaux_1 (cost=0.28..41.60 rows=333 width=4) (actual time=0.088..0.089 rows=1 loops=1)" " Index Cond: (batport = 'AJACCIO'::text)" " Heap Fetches: 1" " -> Hash Join (cost=55.16..167.74 rows=1665 width=11) (actual time=1.853..4.557 rows=25 loops=1)" " Hash Cond: (croisieres.sknum = skippers.sknum)" " -> Hash Join (cost=24.66..132.84 rows=1665 width=4) (actual time=0.663..3.344 rows=25 loops=1)" " Hash Cond: ((croisieres.batnum)::text = (bateaux.batnum)::text)" " -> Seq Scan on croisieres (cost=0.00..95.00 rows=5000 width=10) (actual time=0.016..0.938 rows=5000 loops=1)" " -> Hash (cost=20.50..20.50 rows=333 width=6) (actual time=0.525..0.526 rows=5 loops=1)" " Buckets: 1024 Batches: 1 Memory Usage: 9kB" " -> Seq Scan on bateaux (cost=0.00..20.50 rows=333 width=6) (actual time=0.233..0.513 rows=5 loops=1)" " Filter: (capacite <= (InitPlan 2).col1)" " Rows Removed by Filter: 995" " -> Hash (cost=18.00..18.00 rows=1000 width=15) (actual time=1.175..1.176
```

rows=1000 loops=1)" " Buckets: 1024 Batches: 1 Memory Usage: 55kB" " -> Seq Scan on skippers (cost=0.00..18.00 rows=1000 width=15) (actual time=0.040..0.573 rows=1000 loops=1)" "Planning Time: 2.487 ms" "Execution Time: 4.837 ms"

Exercice Bilan

Question 1

```
SELECT relname, relpages, reltuples FROM pg_class
WHERE relname IN ('banque', 'client')
```

Question 2

1.

```
EXPLAIN ANALYZE
SELECT id_client, nom_client FROM client
WHERE date_naissance > '1965-01-01'
```

```
"Seq Scan on client  (cost=0.00..4817.00 rows=175913 width=20) (actual
time=0.011..18.373 rows=175425 loops=1)"
"  Filter: (date_naissance > '1965-01-01'::date)"
"  Rows Removed by Filter: 24575"
"Planning Time: 0.065 ms"
"Execution Time: 23.516 ms"
```

2.

```
CREATE INDEX idx_date_naissance ON client(date_naissance)
```

3.

```
"Seq Scan on client  (cost=0.00..4817.00 rows=175913 width=20) (actual
time=0.013..22.143 rows=175425 loops=1)"
"  Filter: (date_naissance > '1965-01-01'::date)"
"  Rows Removed by Filter: 24575"
"Planning Time: 0.416 ms"
"Execution Time: 29.099 ms"
```

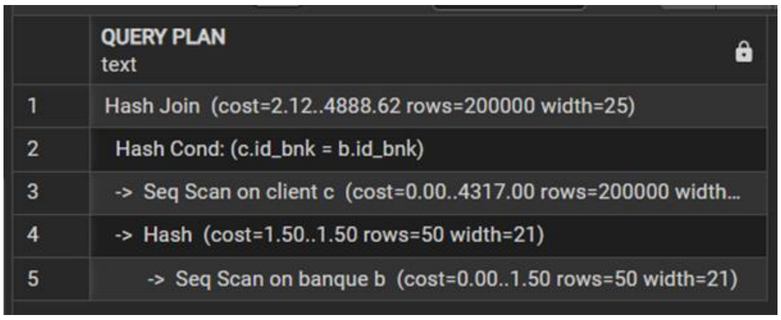
On peut voir qu'après création de l'index, le plan d'exécution n'a pas changé. En effet, PostgreSQL a estimé que le scan séquentiel était plus efficace que l'utilisation de l'index dans ce cas précis. Cela peut être dû au fait qu'une grande partie des lignes (175425 sur 200000) satisfont la condition, rendant l'utilisation de l'index moins avantageuse.

Question 3 :

1)

```
EXPLAIN
select b.nom_bnk
from banque b
join client c on b.id_bnk = c.id_bnk
```

Resultat :



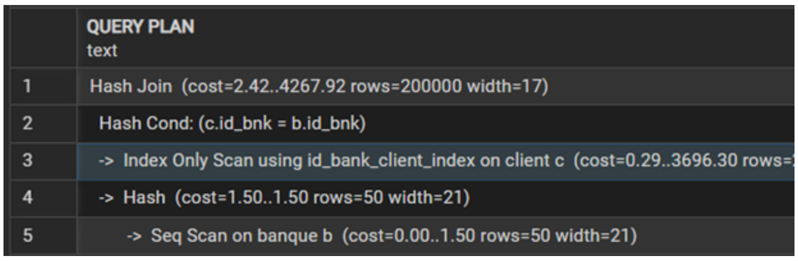
	QUERY PLAN text
1	Hash Join (cost=2.12..4888.62 rows=200000 width=25)
2	Hash Cond: (c.id_bnk = b.id_bnk)
3	-> Seq Scan on client c (cost=0.00..4317.00 rows=200000 width…
4	-> Hash (cost=1.50..1.50 rows=50 width=21)
5	-> Seq Scan on banque b (cost=0.00..1.50 rows=50 width=21)

2) Le type d’algorithme utilisé pour la jointure est un Hash Join, et le coût total de l’exécution est de 4888.62

3)

```
create index id_bank_client_index on client (id_bnk)
```

4) Il l’utilise l’index, et donc le coût total passe à 4267.92



	QUERY PLAN text
1	Hash Join (cost=2.42..4267.92 rows=200000 width=17)
2	Hash Cond: (c.id_bnk = b.id_bnk)
3	-> Index Only Scan using id_bank_client_index on client c (cost=0.29..3696.30 rows=
4	-> Hash (cost=1.50..1.50 rows=50 width=21)
5	-> Seq Scan on banque b (cost=0.00..1.50 rows=50 width=21)

Question 4

1.

```
EXPLAIN ANALYZE
SELECT id_client, nom_client
FROM client
JOIN banque USING(id_bnk)
WHERE nom_bnk = 'BNP Paribas'
"Nested Loop  (cost=47.30..2526.71 rows=4000 width=20) (actual
time=1.994..5.209 rows=4054 loops=1)"
"  ->  Seq Scan on banque  (cost=0.00..1.62 rows=1 width=4) (actual
time=0.048..0.057 rows=1 loops=1)"
"        Filter: ((nom_bnk)::text = 'BNP Paribas'::text)"
"        Rows Removed by Filter: 49"
"  ->  Bitmap Heap Scan on client  (cost=47.30..2485.09 rows=4000 width=24)
(actual time=1.934..4.587 rows=4054 loops=1)"
"        Recheck Cond: (id_bnk = banque.id_bnk)"
"        Heap Blocks: exact=1944"
"        ->  Bitmap Index Scan on idx_id_bnk  (cost=0.00..46.30 rows=4000
width=0) (actual time=1.106..1.106 rows=4054 loops=1)"
"              Index Cond: (id_bnk = banque.id_bnk)"
"Planning Time: 10.715 ms"
"Execution Time: 5.515 ms"
```

2.

L'algorithme de jointure utilisé est une jointure par hachage (Hash Join).

Le coût total de l'exécution est de 4888.14 en coût estimé et 26.879 ms en temps réel.

3.

```
CREATE INDEX idx_nom_bnk ON banque(nom_bnk)
```

4.

```
EXPLAIN ANALYZE
SELECT id_client, nom_client
FROM client
JOIN banque USING(id_bnk)
WHERE nom_bnk = 'BNP Paribas'
```

```
"Hash Join  (cost=1.64..4888.14 rows=4000 width=20) (actual time=0.031..28.776
rows=4054 loops=1)"
"  Hash Cond: (client.id_bnk = banque.id_bnk)"
"  ->  Seq Scan on client  (cost=0.00..4317.00 rows=200000 width=24) (actual
time=0.007..11.284 rows=200000 loops=1)"
"  ->  Hash  (cost=1.62..1.62 rows=1 width=4) (actual time=0.014..0.016 rows=1
loops=1)"
"        Buckets: 1024  Batches: 1  Memory Usage: 9kB"
"        ->  Seq Scan on banque  (cost=0.00..1.62 rows=1 width=4) (actual
time=0.007..0.011 rows=1 loops=1)"
"              Filter: ((nom_bnk)::text = 'BNP Paribas'::text)"
"              Rows Removed by Filter: 49"
"Planning Time: 0.194 ms"
"Execution Time: 28.948 ms"
```

L'algorithme de jointure utilisé est toujours une jointure par hachage (Hash Join).

Le coût total de l'exécution est toujours de 4888.14 en coût estimé et 28.948 ms en temps réel, soit une légère dégradation par rapport à l'exécution précédente. Cela peut s'expliquer par le fait que l'index créé n'est pas utilisé dans ce cas précis, car la table banque est très petite (seulement 50 lignes) et un scan séquentiel est plus efficace.

5.

Tous les noms de banques sont différents entre eux, donc un index sur nom_bnkserait dense, et donc pratique à utiliser. Cependant, la table banque est très petite (seulement 50 lignes), donc l'impact de l'index est négligeable dans ce cas précis.

Question 5

1.

REQ4.0

```
EXPLAIN ANALYZE
SELECT id_client
FROM client
WHERE nom_client = 'client10';
"Seq Scan on client  (cost=0.00..4817.00 rows=1 width=8) (actual
time=0.035..16.980 rows=1 loops=1)"
"  Filter: ((nom_client)::text = 'client10'::text)"
"  Rows Removed by Filter: 199999"
"Planning Time: 0.251 ms"
"Execution Time: 17.035 ms"
```

REQ4.1

```
EXPLAIN ANALYZE
SELECT id_client
FROM client
WHERE upper(nom_client) = 'CLIENT10';
"Gather  (cost=1000.00..5181.71 rows=1000 width=8) (actual time=0.214..33.261
rows=1 loops=1)"
"  Workers Planned: 1"
"  Workers Launched: 1"
"  ->  Parallel Seq Scan on client  (cost=0.00..4081.71 rows=588 width=8)
(actual time=11.631..27.023 rows=0 loops=2)"
"        Filter: (upper((nom_client)::text) = 'CLIENT10'::text)"
"        Rows Removed by Filter: 100000"
"Planning Time: 0.050 ms"
"Execution Time: 33.279 ms"
```

REQ4.2

```
EXPLAIN ANALYZE
SELECT id_client
FROM client
WHERE nom_client = lower('CLIENT10');
"Seq Scan on client  (cost=0.00..4817.00 rows=1 width=8) (actual
time=0.012..9.994 rows=1 loops=1)"
"  Filter: ((nom_client)::text = 'client10'::text)"
"  Rows Removed by Filter: 199999"
"Planning Time: 0.165 ms"
"Execution Time: 10.009 ms"
```

La requête 4.1 est bien plus coûteuse car on effectue une fonction sur chaque ligne de la colonne nom_client, ce qui empêche l'utilisation d'un index. Les requêtes 4.0 et 4.2 sont similaires en termes de coût.

2.

```
CREATE INDEX idnom ON client(nom_client)
```

3.

REQ4.0

```
EXPLAIN ANALYZE
SELECT id_client
FROM client
WHERE nom_client = 'client10';
"Index Scan using idnom on client  (cost=0.42..8.44 rows=1 width=8) (actual
time=0.070..0.072 rows=1 loops=1)"
"  Index Cond: ((nom_client)::text = 'client10'::text)"
"Planning Time: 0.729 ms"
"Execution Time: 0.089 ms"
```

REQ4.1

```
EXPLAIN ANALYZE
SELECT id_client
```

```
FROM client
WHERE upper(nom_client) = 'CLIENT10';
"Gather  (cost=1000.00..5181.71 rows=1000 width=8) (actual time=0.198..39.577
rows=1 loops=1)"
"  Workers Planned: 1"
"  Workers Launched: 1"
"  ->  Parallel Seq Scan on client  (cost=0.00..4081.71 rows=588 width=8)
(actual time=12.545..31.161 rows=0 loops=2)"
"        Filter: (upper((nom_client)::text) = 'CLIENT10'::text)"
"        Rows Removed by Filter: 100000"
"Planning Time: 0.052 ms"
"Execution Time: 39.595 ms"
```

REQ4.2

```
EXPLAIN ANALYZE
SELECT id_client
FROM client
WHERE nom_client = lower('CLIENT10');
"Seq Scan on client  (cost=0.00..4817.00 rows=1 width=8) (actual
time=0.012..9.994 rows=1 loops=1)"
"  Filter: ((nom_client)::text = 'client10'::text)"
"  Rows Removed by Filter: 199999"
"Planning Time: 0.165 ms"
"Execution Time: 10.009 ms"
```

4.

```
CREATE INDEX idnom ON client(upper(nom_client));
```