École Polytechnique

INF553 - Projet

Étape 2

Auteurs:
Mohamed Babana[1]
Thy Vathana[2]

[1]mohamed.babana@polytechnique.edu
[2]thy.vathana@polytechnique.edu

1 Requêtes

1.1

```
SELECT c.name, a.name
FROM artist a, country c
WHERE a.area=c.id
ORDER BY (c.name, a.name) ASC;
```

1.2

```
SELECT c.name, a.name
FROM country c, artist a
WHERE c.id = (SELECT c.id
              FROM country c, artist a
              WHERE c.id = a.area
              GROUP BY (c.id)
              ORDER BY COUNT(1) DESC LIMIT 1);
```

1.3

```
SELECT a.id
FROM artist a, release_country rc, country c, release_has_artist rha
WHERE
          c.name LIKE 'A%'
          AND c.id = rc.country
          AND rc.release=rha.release
          AND rha.artist=a.id
GROUP BY a.id
ORDER BY a.id ASC;
```

1.4

```
SELECT c.id, COUNT(*) AS num_release
FROM country c, release r, release_country rc
WHERE c.id = rc.country AND r.id = rc.release
GROUP BY c.id
ORDER BY num_release DESC;
```

1.5

```
SELECT r.id, a.id
FROM release r, artist a, release_has_artist rha
WHERE
          rha.contribution = 0
          AND rha.release = r.id
          AND rha.artist = a.id
```

```
ORDER BY r.id ASC;
```

1.6

```
SELECT rha.artist, rhb.artist, COUNT(*) AS num_release
FROM release_has_artist rha, release_has_artist rhb
WHERE rha.artist != rhb.artist AND rha.release = rhb.release
GROUP BY (rha.artist, rhb.artist)
ORDER BY rha.artist, rhb.artist ASC;
```

1.7

```
SELECT rc.country, num_track.rid
FROM release_country rc, (
        SELECT r.id AS rid, COUNT(*) AS num
        FROM track tr, release r
        WHERE tr.release = r.id
        GROUP BY rid) num_track
WHERE rc.release = num_track.rid  AND num_track.num >=2
GROUP BY rc.country, num_track.rid
ORDER BY num_track.rid ASC;
```

1.8

```
SELECT rc.release, rc.country
FROM release_country rc
WHERE rc.country != (
        SELECT rc1.country
        FROM release_country rc1
        WHERE rc1.release = rc.release
        GROUP BY (rc1.release, rc1.country, rc1.year, rc1.month, rc1.day)
        HAVING (
                SELECT COUNT(*) AS nb
                FROM release_country rc2
                WHERE rc1.release = rc2.release) > 1
                ORDER BY (rc1.year, rc1.month, rc1.day) ASC LIMIT 1);
```

2 Profilage de requête

2.1 Mesure le temps d'exécution

Le tableau ci-dessous récapitule les durées de 5 exécutions de chaque requête.

Requêtes	Temps 1	Temps 2	Temps 3	Temps 4	Temps 5
Requête 1	7687 ms	7659 ms	7670 ms	7523 ms	7615 ms
Requête 2	662 ms	690 ms	663 ms	670 ms	669 ms
Requête 3	3750 ms	185 ms	185 ms	186 ms	187 ms
Requête 4	4174 ms	1526 ms	1516 ms	1508 ms	1503 ms
Requête 5	3686 ms	3635 ms	3698 ms	3808 ms	3674 ms
Requête 6	2147 ms	2085 ms	2226 ms	2110 ms	2114 ms
Requête 7	39529 ms	14806 ms	14494 ms	14531 ms	14149 ms
Requête 8	4731 ms	4651 ms	4775 ms	4773 ms	4782 ms

Les temps d'exécution de chaque requête sont du même ordre de grandeur. Toutefois, nous constatons que le temps de la première exécution est plus grand que ceux des exécutions suivantes. En effet, la première fois que le serveur reçoit une requête, le serveur doit compiler un plan d'exécution pour déterminer le meilleur moyen d'exécuter la requête. Ce plan est ensuite stocké en mémoire dans le cache au cas où on exécute cette requête.

2.2 Analyse des plans des requêtes

Dans ce qui suit nous fournissons les plans d'exécutions des 8 requêtes.

2.2.1 Requête 1

```
                                   QUERY PLAN
------------------------------------------------------------------------------------------
Gather Merge  (cost=417279.11..537893.33 rows=1033764 width=587)
   Workers Planned: 2
   ->  Sort  (cost=416279.08..417571.29 rows=516882 width=587)
         Sort Key: (ROW(c.name, a.name))
         ->  Hash Join  (cost=8.78..95152.34 rows=516882 width=587)
               Hash Cond: (a.area = c.id)
               ->  Parallel Seq Scan on artist a  (cost=0.00..93778.82 rows=516882 width=508)
               ->  Hash  (cost=5.57..5.57 rows=257 width=55)
                     ->  Seq Scan on country c  (cost=0.00..5.57 rows=257 width=55)
(9 rows)
```

2.2.2 Requête 2

```
                                   QUERY PLAN
------------------------------------------------------------------------------------------
 Nested Loop  (cost=98816.02..212242.56 rows=1240516 width=555)
   InitPlan 1 (returns $1)
     ->  Limit  (cost=98816.02..98816.02 rows=1 width=12)
           ->  Sort  (cost=98816.02..98816.66 rows=257 width=12)
                 Sort Key: (count(1)) DESC
                 ->  Finalize GroupAggregate  (cost=98749.63..98814.74 rows=257 width=12)
                       Group Key: c_1.id
                       ->  Gather Merge  (cost=98749.63..98809.60 rows=514 width=12)
                             Workers Planned: 2
                             ->  Sort  (cost=97749.60..97750.25 rows=257 width=12)
                                   Sort Key: c_1.id
                                   ->  Partial HashAggregate  (cost=97736.75..97739.32 rows=257 width=12)
                                         Group Key: c_1.id
                                         ->  Hash Join  (cost=8.78..95152.34 rows=516882 width=4)
                                               Hash Cond: (a_1.area = c_1.id)
                                               ->  Parallel Seq Scan on artist a_1  (cost=0.00..93778.82 rows=516882 width=
                                               ->  Hash  (cost=5.57..5.57 rows=257 width=4)
                                                     ->  Seq Scan on country c_1  (cost=0.00..5.57 rows=257 width=4)
   ->  Seq Scan on country c  (cost=0.00..6.21 rows=1 width=51)
         Filter: (id = $1)
   ->  Seq Scan on artist a  (cost=0.00..101015.16 rows=1240516 width=504)
(21 rows)
```

2.2.3 Requête 3

```
                                   QUERY PLAN
--------------------------------------------------------------------------------
Group  (cost=78854.20..92396.69 rows=134946 width=4)
  Group Key: a.id
  ->  Gather Merge  (cost=78854.20..92115.55 rows=112456 width=4)
        Workers Planned: 2
        ->  Group  (cost=77854.17..78135.31 rows=56228 width=4)
              Group Key: a.id
              ->  Sort  (cost=77854.17..77994.74 rows=56228 width=4)
                    Sort Key: a.id
                    ->  Nested Loop  (cost=7.27..73418.07 rows=56228 width=4)
                          ->  Nested Loop  (cost=6.84..38222.86 rows=56228 width=4)
                                ->  Hash Join  (cost=6.41..17737.75 rows=41189 width=4)
                                      Hash Cond: (rc.country = c.id)
                                      ->  Parallel Seq Scan on release_country rc  (cost=0.00..15966.00 rows=661600 width=8)
                                      ->  Hash  (cost=6.21..6.21 rows=16 width=4)
                                            ->  Seq Scan on country c  (cost=0.00..6.21 rows=16 width=4)
                                                  Filter: (name ~~ 'A%'::text)
                                ->  Index Only Scan using release_has_artist_pkey on release_has_artist rha  (cost=0.43..0.49 rows=1 width=
                                      Index Cond: (release = rc.release)
                          ->  Index Only Scan using artist_pkey on artist a  (cost=0.43..0.63 rows=1 width=4)
                                Index Cond: (id = rha.artist)
(20 rows)
```

2.2.4 Requête 4

```
                                   QUERY PLAN
--------------------------------------------------------------------------------
 Gather  (cost=99828.47..195454.73 rows=4827 width=555)
   Workers Planned: 2
   Params Evaluated: $2
   InitPlan 2 (returns $2)
     ->  Seq Scan on country c_2  (cost=98816.03..98822.25 rows=1 width=51)
           Filter: (name = $1)
           InitPlan 1 (returns $1)
             ->  Subquery Scan on num_artist  (cost=98816.02..98816.03 rows=1 width=51)
                   ->  Limit  (cost=98816.02..98816.02 rows=1 width=59)
                         ->  Sort  (cost=98816.02..98816.66 rows=257 width=59)
                               Sort Key: (count(*)) DESC
                               ->  Finalize GroupAggregate  (cost=98749.63..98814.74 rows=257 width=59)
                                     Group Key: c_1.name
                                     ->  Gather Merge  (cost=98749.63..98809.60 rows=514 width=59)
                                           Workers Planned: 2
                                           ->  Sort  (cost=97749.60..97750.25 rows=257 width=59)
                                                 Sort Key: c_1.name
                                                 ->  Partial HashAggregate  (cost=97736.75..97739.32 rows=257 width=59)
                                                       Group Key: c_1.name
                                                       ->  Hash Join  (cost=8.78..95152.34 rows=516882 width=51)
                                                             Hash Cond: (a_1.area = c_1.id)
                                                             ->  Parallel Seq Scan on artist a_1  (cost=0.00..93778.82 rows
                                                             ->  Hash  (cost=5.57..5.57 rows=257 width=55)
                                                                   ->  Seq Scan on country c_1  (cost=0.00..5.57 rows=257 w
   ->  Hash Join  (cost=6.23..95149.78 rows=2011 width=555)
         Hash Cond: (a.area = c.id)
         ->  Parallel Seq Scan on artist a  (cost=0.00..93778.82 rows=516882 width=508)
         ->  Hash  (cost=6.21..6.21 rows=1 width=55)
               ->  Seq Scan on country c  (cost=0.00..6.21 rows=1 width=55)
                     Filter: (name = $2)
(30 rows)
```

2.2.5 Requête 5

```
                                   QUERY PLAN
--------------------------------------------------------------------------------
 Gather Merge  (cost=291530.01..471174.45 rows=1539702 width=8)
   Workers Planned: 2
   ->  Sort  (cost=290529.98..292454.61 rows=769851 width=8)
         Sort Key: r.id
         ->  Parallel Hash Join  (cost=161743.37..204732.80 rows=769851 width=8)
               Hash Cond: (rha.artist = a.id)
               ->  Parallel Hash Join  (cost=59483.53..92416.09 rows=769851 width=8)
                     Hash Cond: (rha.release = r.id)
                     ->  Parallel Seq Scan on release_has_artist rha  (cost=0.00..21894.71 rows=769851 width=8)
                           Filter: (contribution = 0)
                     ->  Parallel Hash  (cost=46881.12..46881.12 rows=768112 width=4)
                           ->  Parallel Seq Scan on release r  (cost=0.00..46881.12 rows=768112 width=4)
               ->  Parallel Hash  (cost=93778.82..93778.82 rows=516882 width=4)
                     ->  Parallel Seq Scan on artist a  (cost=0.00..93778.82 rows=516882 width=4)
(14 rows)
```

2.2.6 Requête 6

```
                                   QUERY PLAN
--------------------------------------------------------------------------------
Finalize GroupAggregate  (cost=213936.85..566734.57 rows=2921220 width=16)
   Group Key: rha.artist, rhb.artist
```

```
  ->  Gather Merge  (cost=213936.85..519264.74 rows=2434350 width=16)
        Workers Planned: 2
        ->  Partial GroupAggregate  (cost=212936.83..237280.33 rows=1217175 width=16)
              Group Key: rha.artist, rhb.artist
              ->  Sort  (cost=212936.83..215979.76 rows=1217175 width=8)
                    Sort Key: rha.artist, rhb.artist
                    ->  Parallel Hash Join  (cost=33847.68..73267.72 rows=1217175 width=8)
                          Hash Cond: (rha.release = rhb.release)
                          Join Filter: (rha.artist <> rhb.artist)
                          ->  Parallel Seq Scan on release_has_artist rha  (cost=0.00..19745.97 rows=859497 width=8)
                          ->  Parallel Hash  (cost=19745.97..19745.97 rows=859497 width=8)
                                ->  Parallel Seq Scan on release_has_artist rhb  (cost=0.00..19745.97 rows=859497 width=8)
(14 rows)
```

2.2.7 Requête 7

```
                                                   QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------
 Group  (cost=1920038.16..2575208.36 rows=25000 width=8)
   Group Key: r.id, rc.country
   ->  Merge Join  (cost=1920038.16..2571979.85 rows=645702 width=8)
         Merge Cond: (rc.release = r.id)
         ->  Index Only Scan using release_country_pkey on release_country rc  (cost=0.43..76658.95 rows=1587840 width=8)
         ->  Finalize GroupAggregate  (cost=1920037.74..2477213.16 rows=614490 width=12)
               Group Key: r.id
               Filter: (count(*) >= 2)
               ->  Gather Merge  (cost=1920037.74..2435735.08 rows=3686940 width=12)
                     Workers Planned: 2
                     ->  Partial GroupAggregate  (cost=1919037.71..2009170.70 rows=1843470 width=12)
                           Group Key: r.id
                           ->  Sort  (cost=1919037.71..1942937.14 rows=9559772 width=4)
                                 Sort Key: r.id
                                 ->  Parallel Hash Join  (cost=59483.53..549250.71 rows=9559772 width=4)
                                       Hash Cond: (tr.release = r.id)
                                       ->  Parallel Seq Scan on track tr  (cost=0.00..386985.72 rows=9559772 width=4)
                                       ->  Parallel Hash  (cost=46881.12..46881.12 rows=768112 width=4)
                                             ->  Parallel Seq Scan on release r  (cost=0.00..46881.12 rows=768112 width=4)
(19 rows)
```

2.2.8 Requête 8

```
                                                   QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------
Seq Scan on release_country rc  (cost=0.00..26903390.00 rows=1579901 width=8)
 Filter: (country <> (SubPlan 2))
 SubPlan 2
   ->  Limit  (cost=16.92..16.93 rows=1 width=52)
     ->  Sort  (cost=16.92..16.93 rows=1 width=52)
       Sort Key: (ROW(rc1.year, rc1.month, rc1.day))
       ->  Group  (cost=0.43..16.91 rows=1 width=52)
             Group Key: rc1.release, rc1.country
             Filter: ((SubPlan 1) > 1)
            ->  Index Scan using release_country_pkey on release_country rc1  (cost=0.43..8.45 rows=1 width=20)
                  Index Cond: (release = rc.release)
             SubPlan 1
               ->  Aggregate  (cost=8.45..8.46 rows=1 width=8)
                 ->  Index Only Scan using release_country_pkey on release_country rc2  (cost=0.43..8.45 rows=1 width=0)
                       Index Cond: (release = rc1.release)
(15 rows)
```

3 Amélioration des performances

Pour cette partie nous avons choisit d'améliorer les requettes 2, 3, et 4.

3.1 Requête 2

```sql
SELECT c.name, a.name
FROM country c, artist a
WHERE c.id = (SELECT c.id
              FROM country c, artist a
              WHERE c.id = a.area
              GROUP BY (c.id)
              ORDER BY COUNT(1) DESC LIMIT 1);
```

Nous proposons la version améliorée suivante :

```
SELECT c.name, a.name
FROM artist a, country c WHERE c.id = a.area AND c.name = (
        SELECT c.name
        FROM country c
        WHERE c.name = (
                SELECT cname
                FROM (
                        SELECT c.name AS cname, COUNT(*) AS num
                        FROM artist a, country c
                        WHERE a.area = c.id
                        GROUP BY cname
                        ORDER BY num DESC
                        LIMIT 1 )
                AS num_artist)
);
```

Le plan d'exécution de cette requête est :

```
QUERY PLAN
------------------------------------------------------------------------------------------------------------------------------------------------
 Gather  (cost=99828.54..195454.86 rows=4827 width=555)
   Workers Planned: 2
   Params Evaluated: $2
   InitPlan 2 (returns $2)
     ->  Seq Scan on country c_2  (cost=98816.10..98822.31 rows=1 width=51)
           Filter: (name = $1)
           InitPlan 1 (returns $1)
             ->  Subquery Scan on num_artist  (cost=98816.09..98816.10 rows=1 width=51)
                   ->  Limit  (cost=98816.09..98816.09 rows=1 width=59)
                         ->  Sort  (cost=98816.09..98816.73 rows=257 width=59)
                               Sort Key: (count(1)) DESC
                               ->  Finalize GroupAggregate  (cost=98749.69..98814.80 rows=257 width=59)
                                     Group Key: c_1.name
                                     ->  Gather Merge  (cost=98749.69..98809.66 rows=514 width=59)
                                           Workers Planned: 2
                                           ->  Sort  (cost=97749.67..97750.31 rows=257 width=59)
                                                 Sort Key: c_1.name
                                                 ->  Partial HashAggregate  (cost=97736.81..97739.38 rows=257 width=59)
                                                       Group Key: c_1.name
                                                       ->  Hash Join  (cost=8.78..95152.40 rows=516882 width=51)
                                                             Hash Cond: (a_1.area = c_1.id)
                                                             ->  Parallel Seq Scan on artist a_1  (cost=0.00..93778.82 rows
                                                             ->  Hash  (cost=5.57..5.57 rows=257 width=55)
                                                                   ->  Seq Scan on country c_1  (cost=0.00..5.57 rows=257 w
   ->  Hash Join  (cost=6.23..95149.84 rows=2011 width=555)
         Hash Cond: (a.area = c.id)
         ->  Parallel Seq Scan on artist a  (cost=0.00..93778.82 rows=516882 width=508)
         ->  Hash  (cost=6.21..6.21 rows=1 width=55)
               ->  Seq Scan on country c  (cost=0.00..6.21 rows=1 width=55)
                     Filter: (name = $2)
(30 rows)
```

Le temps d'exécution de cette requête est : **257 ms**

3.2 Requête 3

```sql
SELECT a.id
FROM artist a, release_country rc, country c, release_has_artist rha
WHERE
        c.name LIKE 'A%'
        AND c.id = rc.country
        AND rc.release=rha.release
        AND rha.artist=a.id
GROUP BY a.id
ORDER BY a.id ASC;
```

Nous proposons la version améliorée suivante :

```sql
SELECT rha.artist
FROM release_has_artist rha
INNER JOIN (
        SELECT rc.release as release
        FROM release_country rc
        INNER JOIN (
                SELECT c1.id AS id
                FROM country c1
                WHERE
                        c1.name LIKE 'A%') AS c2
                        ON (rc.country = c2.id)) AS re
                        ON (rha.release = re.release)
GROUP BY rha.artist
ORDER BY rha.artist ASC;
```

Le plan d'exécution de cette requête est :

```
                                              QUERY PLAN
------------------------------------------------------------------------------------------------------------------------
Sort  (cost=56491.79..56633.45 rows=56663 width=4)
  Sort Key: rha.artist
  ->  Finalize HashAggregate  (cost=51451.58..52018.21 rows=56663 width=4)
        Group Key: rha.artist
        ->  Gather  (cost=39363.42..51170.46 rows=112448 width=4)
              Workers Planned: 2
              ->  Partial HashAggregate  (cost=38363.42..38925.66 rows=56224 width=4)
                    Group Key: rha.artist
                    ->  Nested Loop  (cost=6.84..38222.86 rows=56224 width=4)
                          ->  Hash Join  (cost=6.41..17737.75 rows=41189 width=4)
                                Hash Cond: (rc.country = c1.id)
                                ->  Parallel Seq Scan on release_country rc  (cost=0.00..15966.00 rows=661600 width=8)
                                ->  Hash  (cost=6.21..6.21 rows=16 width=4)
                                      ->  Seq Scan on country c1  (cost=0.00..6.21 rows=16 width=4)
                                            Filter: (name ~~ 'A%'::text)
                          ->  Index Only Scan using release_has_artist_pkey on release_has_artist rha  (cost=0.43..0.49 ro
                                Index Cond: (release = rc.release)
(17 rows)
```

Le temps d'exécution de cette requête est : **102 ms**

3.3 Requête 4

```
SELECT c.id, COUNT(*) AS num_release
FROM country c, release r, release_country rc
WHERE c.id = rc.country AND r.id = rc.release
GROUP BY c.id
ORDER BY num_release DESC;
```

Nous proposons la version améliorée suivante :

```
SELECT rc.country, COUNT(*) AS num
FROM release_country rc
INNER JOIN country c ON (rc.country = c.id)
GROUP BY rc.country
ORDER BY num DESC;
```

Le plan d'exécution de cette requête est :

```
                                                QUERY PLAN
--------------------------------------------------------------------------------------------------------------
Sort  (cost=22088.38..22088.68 rows=121 width=12)
   Sort Key: (count(*)) DESC
   ->  Finalize GroupAggregate  (cost=22053.54..22084.19 rows=121 width=12)
         Group Key: rc.country
         ->  Gather Merge  (cost=22053.54..22081.77 rows=242 width=12)
               Workers Planned: 2
               ->  Sort  (cost=21053.51..21053.82 rows=121 width=12)
                     Sort Key: rc.country
                     ->  Partial HashAggregate  (cost=21048.12..21049.33 rows=121 width=12)
                           Group Key: rc.country
                           ->  Hash Join  (cost=8.78..17740.12 rows=661600 width=4)
                                 Hash Cond: (rc.country = c.id)
                                 ->  Parallel Seq Scan on release_country rc  (cost=0.00..15966.00 rows=661600 width=4)
                                 ->  Hash  (cost=5.57..5.57 rows=257 width=4)
                                       ->  Seq Scan on country c  (cost=0.00..5.57 rows=257 width=4)
(15 rows)
```

Le temps d'exécution de cette requête est : **117 ms**