

Indexation

Structures arborescentes et hachage

P. Rigaux

CNAM Paris

May 10, 2011

L'optimiseur

L'optimiseur ORACLE suit une approche classique :

- Génération de plusieurs plans d'exécution.
- Estimation du coût de chaque plan généré.
- Choix du meilleur et exécution.

Tout ceci est automatique, **mais il est possible d'influer, voire de forcer le plan d'exécution.**

Estimation du coût d'un plan d'exécution

Beaucoup de paramètres entrent dans l'estimation du coût :

- Les chemins d'accès disponibles.
- Les opérations physiques de traitement des résultats intermédiaires.
- Des statistiques sur les tables concernées (taille, sélectivité). Les statistiques sont calculées par appel explicite à l'outil ANALYSE.
- Les ressources disponibles.

L'optimiseur basé sur les règles

Priorité	Description
1	Accès à un enregistrement par ROWID
2	Accès à un enregistrement dans un <i>cluster</i>
3	Accès à un enregistrement dans une table de hachage (<i>hash cluster</i>)
4	Accès à un enregistrement par clé unique (avec un index)
...	...
15	Balayage complet de la table

L'optimiseur basé sur les coûts

Principaux paramètres :

- ❶ OPTIMIZER_MODE (RULE, CHOOSE, FIRST_ROW, ALL_ROWS).
- ❷ SORT_AREA_SIZE (taille de la zone de tri).
- ❸ HASH_AREA_SIZE (taille de la zone de hachage).
- ❹ HASH_JOIN_ENABLED considère les jointures par hachage.

Création des statistiques

- 1 Calcul de la taille et du nombre de lignes :

```
ANALYSE TABLE Film COMPUTE STATISTICS FOR TABLE
```

- 2 Analyse des index :

```
ANALYSE TABLE Film COMPUTE STATISTICS FOR ALL INDEX
```

- 3 Analyse de la distribution des valeurs :

```
ANALYSE TABLE Film COMPUTE STATISTICS FOR COLUMNS titre,  
genre
```

Les chemins d'accès

- **Parcours séquentiel** (*FULL TABLE sSCAN*).
- **Par adresse** (*ACCESS BY ROWID*).
- **Parcours de regroupement** (*CLUSTER SCAN*). On récupère alors dans une même lecture les n-uplets des 2 tables du *cluster*.
- **Recherche par hachage** (*HASH SCAN*).
- **Parcours d'index** (*INDEX SCAN*).

Opérations physiques

Voici les principales :

- *INTERSECTION* : intersection de deux ensembles de n-uplet.
- *CONCATENATION* : union de deux ensembles.
- *FILTER* : élimination de n-uplets (sélection).
- *PROJECTION* : opération de l'algèbre relationnelle.

D'autres opérations sont liées aux algorithmes de jointures.

Algorithmes de jointure sous ORACLE

ORACLE utilise trois algorithmes de jointures :

- **Boucles imbriquées** quand il y a au moins un index.
Opération NESTED LOOP.
- **Tri/fusion** quand il n'y a pas d'index.
Opération SORT et MERGE.
- **Jointure par hachage** quand il n'y a pas d'index.
Opération HASH JOIN

L'outil EXPLAIN

L'outil EXPLAIN donne le plan d'exécution d'une requête. La description comprend :

- Le chemin d'accès utilisé.
- Les opérations physiques (tri, fusion, intersection, ...).
- L'ordre des opérations.

Il est représentable par un arbre.

Rappel du schéma

- Film (**idFilm**, titre, année, genre, résumé, *idMES*, *codePays*)
- Artiste (**idArtiste**, nom, prénom, annéeNaissance)
- Role (*idActeur*, *idFilm*, nomRôle)
- Internaute (**email**, nom, prénom, région)
- Notation (**email**, *idFilm*, note)
- Pays (**code**, nom, langue)

Sélection sans index

La requête :

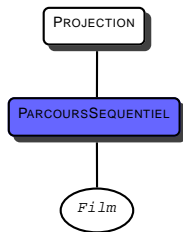
```
EXPLAIN PLAN  
SET STATEMENT_ID='SelSansInd' FOR  
SELECT * FROM Film  
WHERE titre = 'Vertigo'
```

Le résultat de EXPLAIN :

```
0 SELECT STATEMENT  
  1 TABLE ACCESS FULL FILM
```

Plan d'exécution

On ne fait pas plus simple !



Sélection avec index

La requête :

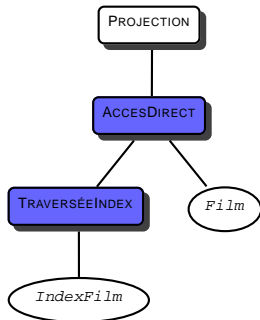
```
EXPLAIN PLAN
SET STATEMENT_ID='SelInd' FOR
SELECT *
FROM Film
WHERE idFilm=21;
```

Le résultat de EXPLAIN :

```
0 SELECT STATEMENT
  1 TABLE ACCESS BY ROWID FILM
    2 INDEX UNIQUE SCAN IDX-FILM-ID
```

Plan d'exécution

Accès à l'index, puis à la table.



Jointure avec index

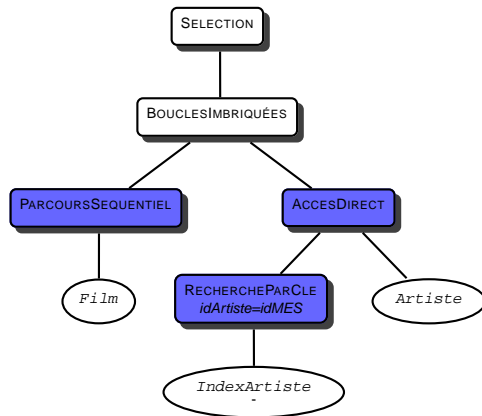
La requête :

```
EXPLAIN PLAN
SET STATEMENT_ID='JoinIndex' FOR
SELECT titre, nom, prenom
FROM   Film f, Artiste a
WHERE  idMES = idArtiste;
```

Le résultat de EXPLAIN :

```
0 SELECT STATEMENT
  1 NESTED LOOPS
    2 TABLE ACCESS FULL FILM
    3 TABLE ACCESS BY ROWID ARTISTE
      4 INDEX UNIQUE SCAN IDXARTISTE
```


Plan d'exécution



Jointure avec index et sélection

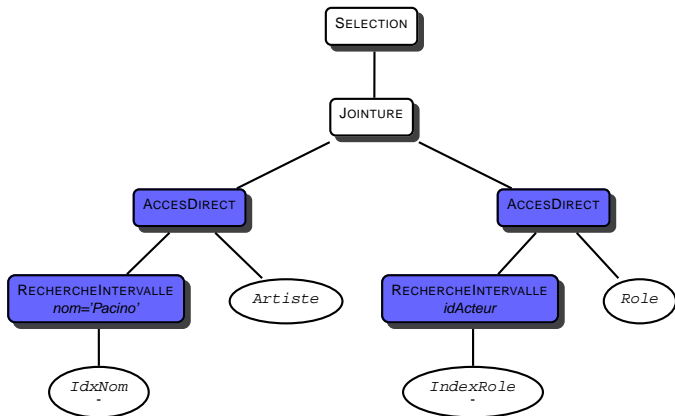
La requête :

```
EXPLAIN PLAN SET  
STATEMENT_ID='JoinSelIndex' FOR  
SELECT nomRole  
FROM   Role r, Artiste a  
WHERE  r.idActeur = a.idArtiste  
AND    nom = 'Pacino';
```

Le résultat de EXPLAIN :

```
0 SELECT STATEMENT  
  1 NESTED LOOPS  
    2 TABLE ACCESS BY ROWID ARTISTE  
      3 INDEX RANGE SCAN IDX-NOM  
    4 TABLE ACCESS BY ROWID ROLE  
      5 INDEX RANGE SCAN IDX-ROLE
```

Plan d'exécution



Jointure sans index

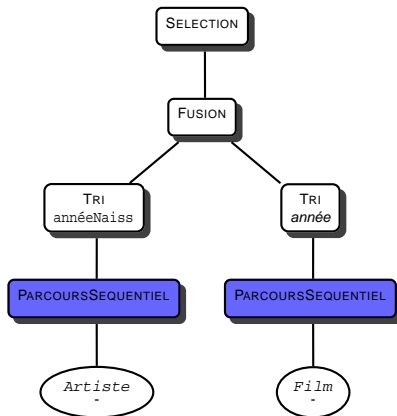
La requête :

```
EXPLAIN PLAN SET  
STATEMENT_ID='JoinSansIndex' FOR  
SELECT nom, prenom  
FROM Film f, Artiste a  
WHERE f.annee = a.anneeNaiss  
AND titre = 'Vertigo';
```

Le résultat de EXPLAIN :

```
0 SELECT STATEMENT  
  1 MERGE JOIN  
    2 SORT JOIN  
      3 TABLE ACCESS FULL ARTISTE  
    4 SORT JOIN  
      5 TABLE ACCESS FULL FILM
```

Plan d'exécution



Analyse des coûts

Avec l'utilitaire TKPROF, on obtient :

- Le coût CPU
- Le nombre d'entrées/sorties physiques
- Le nombre d'entrées/sorties en mémoire
- Le nombre de tris
- etc

Un excellent outil à utiliser : la trace automatique qui donne à la fois le EXPLAIN et les coûts d'exécution.