

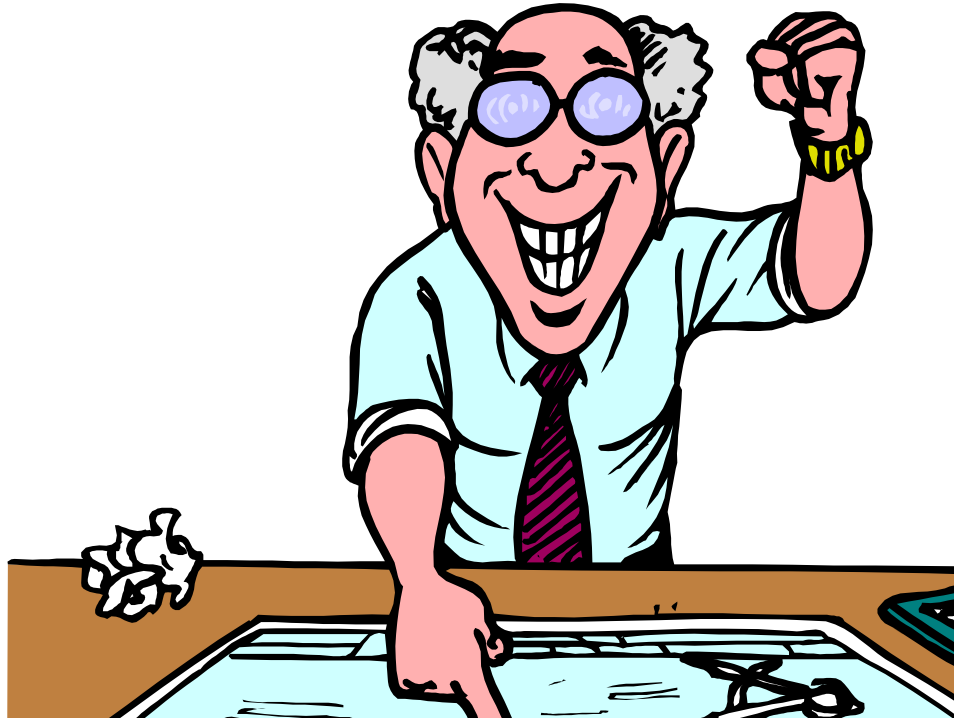


Conception Avancée de Base de Données

Objectifs



Objectif du cours



Objectif du cours : Data Base Optimisation



■ Data Base Model Optimisation

- Functional Dependencies
- Normalization
- Physical Design
- Indexing

=> **Master 1**

■ Database Management Sytem Performances

- Request Execution Plan
- Explain Plan
- Relationnal Operators implementation
- Index Implementation
- Buffer Management

Performances des bases de données



■ Explan Plan

- Cost
- Seq Scan
- Row ID
- Nested Loop
- Hash Join



■ Analyse



Explain Plan

- Mysql
- Postgres
- Oracle



ORACLE®



localhost / localhost / stones / stones | phpMyAdmin 3.1.3.1 - Microsoft Internet Explorer

Fichier Edition Affichage Favoris Outils ?

Adresse <http://localhost/phpmyadmin/index.php?db=alg2sql&token=a7a9a20c198c13c401869e1d5d32d868> OK Google Search Sign In

Google This page is in French. Translate it using Google Toolbar? The content of this intranet page will be sent to Google for translation using a secure connection. [Learn more](#) Translate Turn off French translation

phpMyAdmin

Base de données
stones (3)

stones (3)

- albums
- auteurs
- stones

Serveur: localhost Base de données: stones Table: stones

Afficher Structure SQL Rechercher Insérer Exporter Importer Opérations Vider Supprimer

✓ Votre requête SQL a été exécutée avec succès

```
EXPLAIN SELECT Titre, Groupe, Prenom, Nom, Album
FROM stones.auteurs, albums
WHERE stones.auteurs_id = auteurs.auteurs_id
AND stones.album_id = albums.album_id
```

[Modifier] [Ne pas expliquer SQL] [Créer source PHP]

+ Options

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	auteurs	ALL	NULL	NULL	NULL	NULL	3	
1	SIMPLE	albums	ALL	NULL	NULL	NULL	NULL	4	Using join buffer
1	SIMPLE	stones	ALL	NULL	NULL	NULL	NULL	7	Using where; Using join buffer

Opérations sur les résultats de la requête

Version imprimable Version imprimable (avec textes complets) CREATE VIEW

MySQL



```
ex d:\pgm\wamp\bin\mysql\mysql5.1.33\bin\mysql.exe
mysql> use stones;
Database changed
mysql> SELECT
-> Titre,
-> Groupe,
-> Prenom,
-> Nom,
-> Album
-> FROM
-> stones,
-> auteurs,
-> albums
-> WHERE
-> stones.auteurs_id = auteurs.auteurs_id
-> AND
-> stones.album_id = albums.album_id;
```

Titre	Groupe	Prenom	Nom	Album
Rough Justice	The Rolling Stones	Michael Philip	Jagger	A Bigger Bang
Street Fighting Man	The Rolling Stones	Michael Philip	Jagger	Beggars Banquet
Ventilator Blues	The Rolling Stones	Michael Philip	Jagger	Exile On Main Street
Rough Justice	The Rolling Stones	Keith	Richard	A Bigger Bang
Street Fighting Man	The Rolling Stones	Keith	Richard	Beggars Banquet
Ventilator Blues	The Rolling Stones	Keith	Richard	Exile On Main Street
Ventilator Blues	The Rolling Stones	Mick	Taylor	Exile On Main Street

```
7 rows in set (0.00 sec)

mysql> explain
-> SELECT
-> Titre,
-> Groupe,
-> Prenom,
-> Nom,
-> Album
-> FROM
-> stones,
-> auteurs,
-> albums
-> WHERE
-> stones.auteurs_id = auteurs.auteurs_id
-> AND
-> stones.album_id = albums.album_id;
```

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	auteurs	ALL	NULL	NULL	NULL	NULL	3	
1	SIMPLE	albums	ALL	NULL	NULL	NULL	NULL	4	Using join buffer
1	SIMPLE	stones	ALL	NULL	NULL	NULL	NULL	7	Using where; Using join buffer

```
3 rows in set (0.00 sec)

mysql> _
```

Postgres



Query - stones sur postgres@localhost : 5432 *

Fichier Édition Requêtes Favoris Macros Affichage Aide

Éditeur SQL Constructeur graphique de requêtes

```
SELECT
  auteurs.prenom,
  auteurs.nom,
  albums.album,
  albums.annee
FROM
  public.albums,
  public.auteurs,
  public.stones
WHERE
  albums.album_id = stones.album_id AND
  stones.auteurs_id = auteurs.auteurs_id;
```

Panneau sortie

Sortie de données Expliquer (Explain) Messages Historique

The diagram illustrates the execution plan for the provided SQL query. It shows three input tables: 'albums', 'auteurs', and 'stones'. The 'albums' table is joined with the 'stones' table using a Hash Join operation. The result of this join is then joined with the 'auteurs' table using another Hash Join operation. The final output is a list of columns: auteurs.prenom, auteurs.nom, albums.album, and albums.annee.

OK. Unix Ligne 13 Col 1 Caract. 224 10 lignes. 16 ms

Postgres



PostgreSQL



Query - stones sur postgres@localhost : 5432 *

Fichier Édition Requêtes Favoris Macros Affichage Aide

Éditeur SQL Constructeur graphique de requêtes

```
SELECT
  auteurs.prenom,
  auteurs.nom,
  albums.album,
  albums.annee
FROM
  public.albums,
  public.auteurs,
  public.stones
WHERE
  albums.album_id = stones.album_id AND
  stones.auteurs_id = auteurs.auteurs_id;
```

Bloc notes

Panneau sortie

Sortie de données Expliquer (Explain) Messages Historique

QUERY PLAN

text

1	Hash Join (cost=2.14..3.40 rows=7 width=38) (actual time=0.146..0.240 rows=7 loops=1)
2	Output: auteurs.prenom, auteurs.nom, albums.album, albums.annee
3	Hash Cond: (stones.auteurs_id = auteurs.auteurs_id)
4	-> Hash Join (cost=1.07..2.23 rows=7 width=26) (actual time=0.070..0.126 rows=7 loops=1)
5	Output: albums.album, albums.annee, stones.auteurs_id
6	Hash Cond: (stones.album_id = albums.album_id)
7	-> Seq Scan on stones (cost=0.00..1.07 rows=7 width=8) (actual time=0.006..0.022 rows=7 loops=1)
8	Output: stones.titre, stones.groupe, stones.auteurs_id, stones.album_id
9	-> Hash (cost=1.03..1.03 rows=3 width=26) (actual time=0.030..0.030 rows=3 loops=1)
10	Output: albums.album, albums.annee, albums.album_id
11	-> Seq Scan on albums (cost=0.00..1.03 rows=3 width=26) (actual time=0.005..0.013 rows=3 loops=1)
12	Output: albums.album, albums.annee, albums.album_id
13	-> Hash (cost=1.03..1.03 rows=3 width=20) (actual time=0.042..0.042 rows=3 loops=1)
14	Output: auteurs.prenom, auteurs.nom, auteurs.auteurs_id
15	-> Seq Scan on auteurs (cost=0.00..1.03 rows=3 width=20) (actual time=0.014..0.023 rows=3 loops=1)
16	Output: auteurs.prenom, auteurs.nom, auteurs.auteurs_id
17	Total runtime: 0.409 ms

OK.

Unix Ligne 13 Col 1 Caract. 224 17 lignes. 16 ms



```

Oracle SQL*Plus
File Edit Search Options Help

SQL>
SQL> 1
  1 EXPLAIN PLAN SET STATEMENT_ID = 'HOTKA' for SELECT count(*)
  2 from B, C, A
  3 WHERE A.STATUS = B.STATUS
  4 AND   A.B_ID = B.ID
  5 AND   B.STATUS = 'OPEN'
  6 AND   B.ID = C.B_ID
  7* AND  C.STATUS = 'OPEN'
SQL> /

Explained.

SQL> start c:\temp\SHOW_PLAN.sql HOTKA
old  4: where statement_id = '&1'
new  4: where statement_id = 'HOTKA'

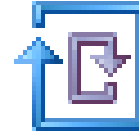
Cost      ID P_ID Access                Access          Object
-----  - - - - Path                Name
-----  - - - -
          0      SELECT STATEMENT
          1      0      SORT                AGGREGATE
          2      1      TABLE ACCESS       BY INDEX ROWID  C
          3      2      NESTED LOOPS
          4      3      NESTED LOOPS
          5      4      TABLE ACCESS       BY INDEX ROWID  B
          6      5      INDEX               RANGE SCAN      B_STATUS_IDX
          7      4      TABLE ACCESS       BY INDEX ROWID  A
          8      7      INDEX               RANGE SCAN      A_STATUS_IDX
          9      3      INDEX               RANGE SCAN      C_B_ID_IDX

10 rows selected.
    
```

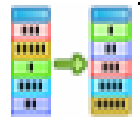
Join Algorithmn Implementation



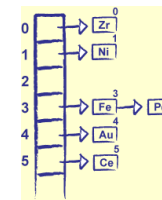
- Nested loop



- Merge join



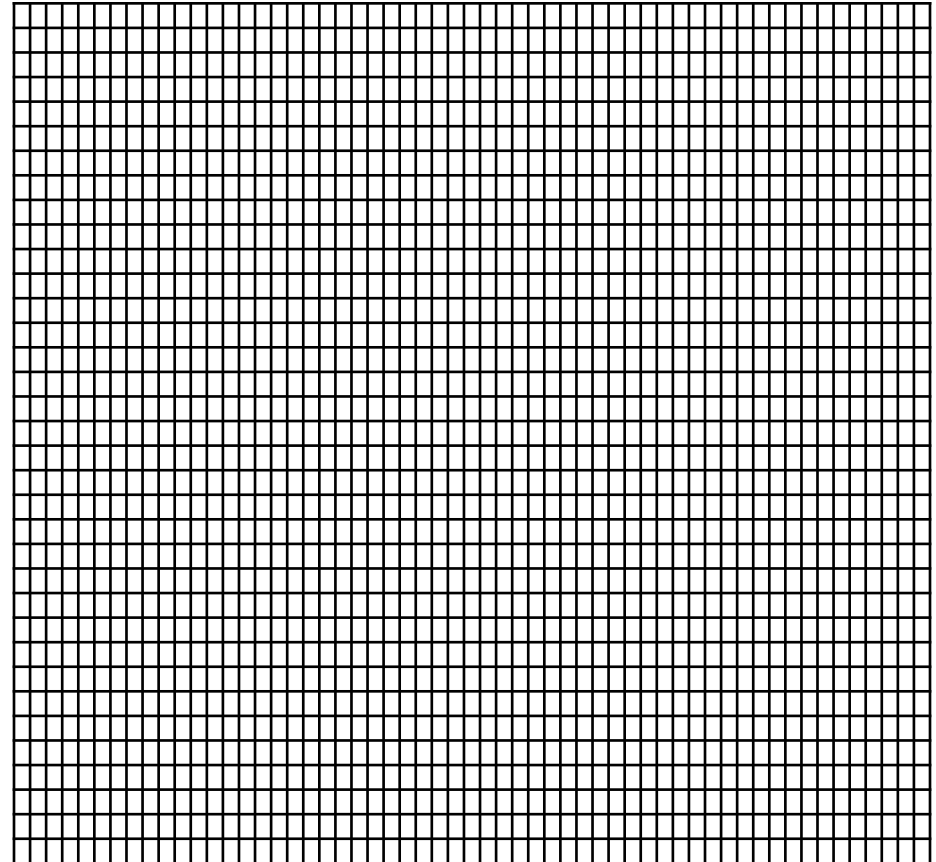
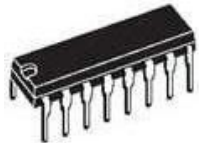
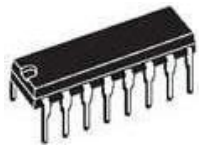
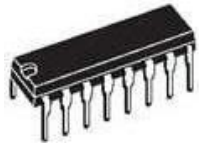
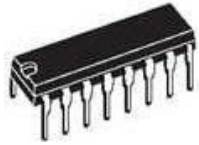
- Hash join



Big Relation => Big Data

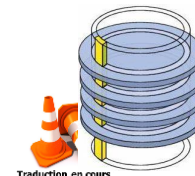


Relation



Memory

Disc

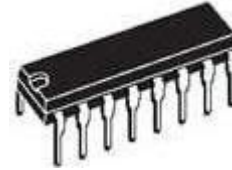


Jointures Physiques



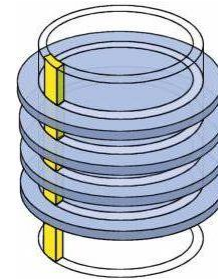
■ Jointure en mémoire

- Nested Loop
- Merge join
- Hash join

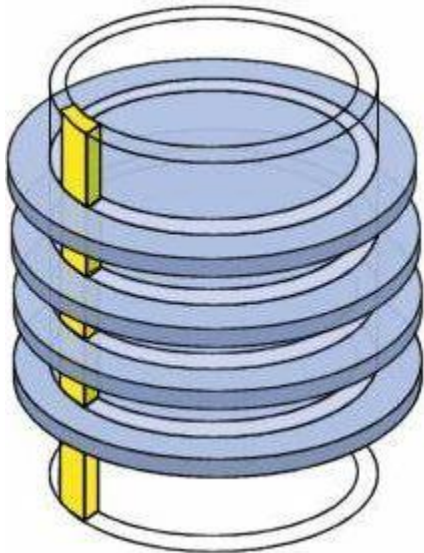
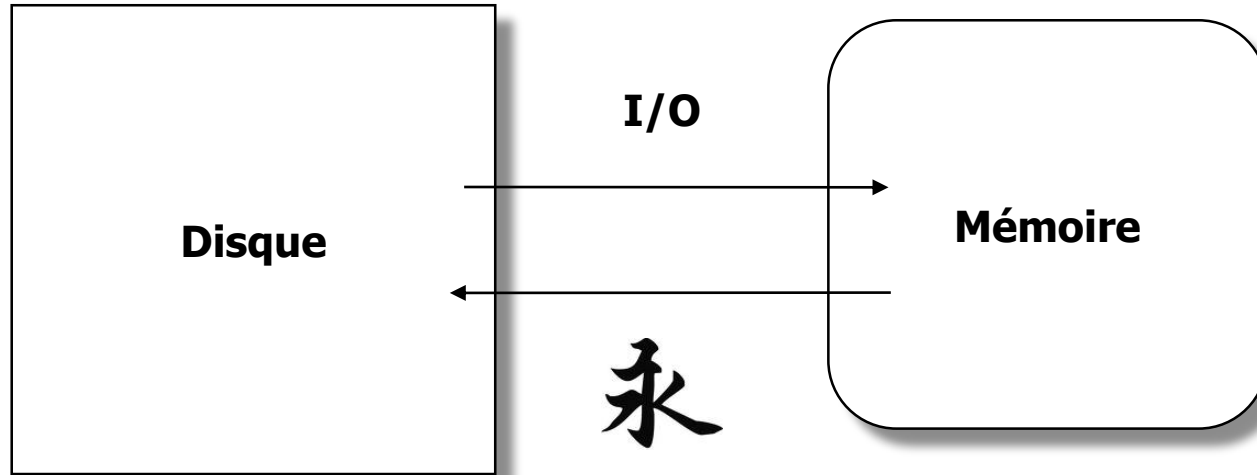


■ Jointure sur disque

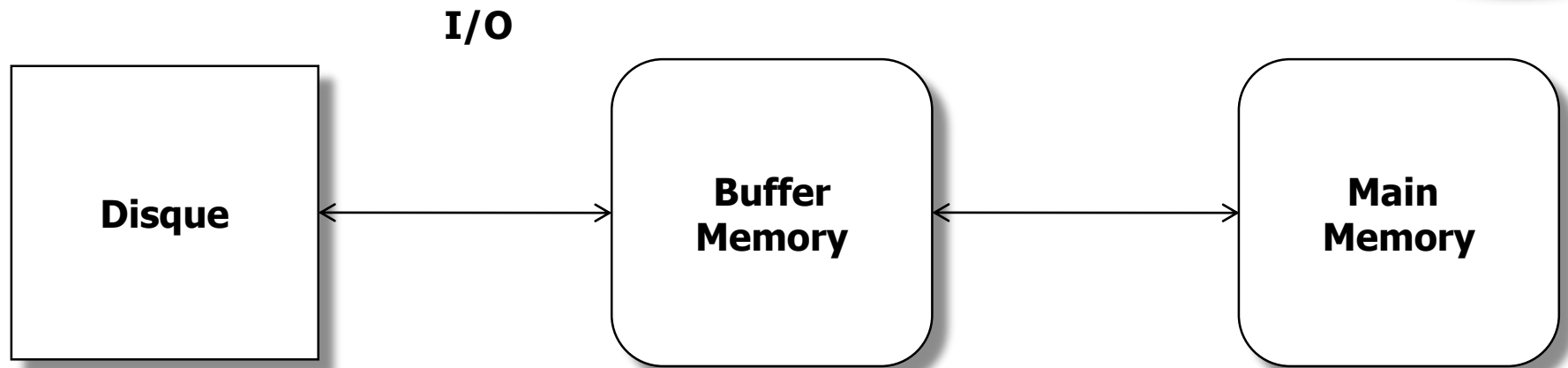
- Nested Loop
- Sort Merge Join
- Hash join



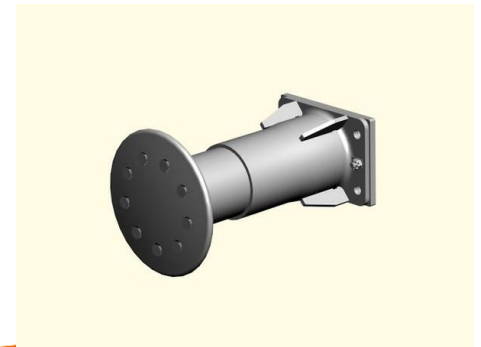
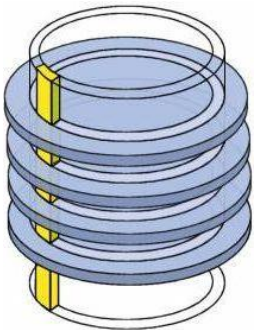
Entrées/Sorties, Lectures écritures disques, IO



Entrées/Sorties, Lectures écritures disques, IO



永



Gartner Hype Cycle



Hype Cycle for the Digital Workplace, 2020



gartner.com/SmarterWithGartner

Source: Gartner
© 2020 Gartner, Inc. and/or its affiliates. All rights reserved. Gartner and Hype Cycle are registered trademarks of Gartner, Inc. and its affiliates in the U.S.

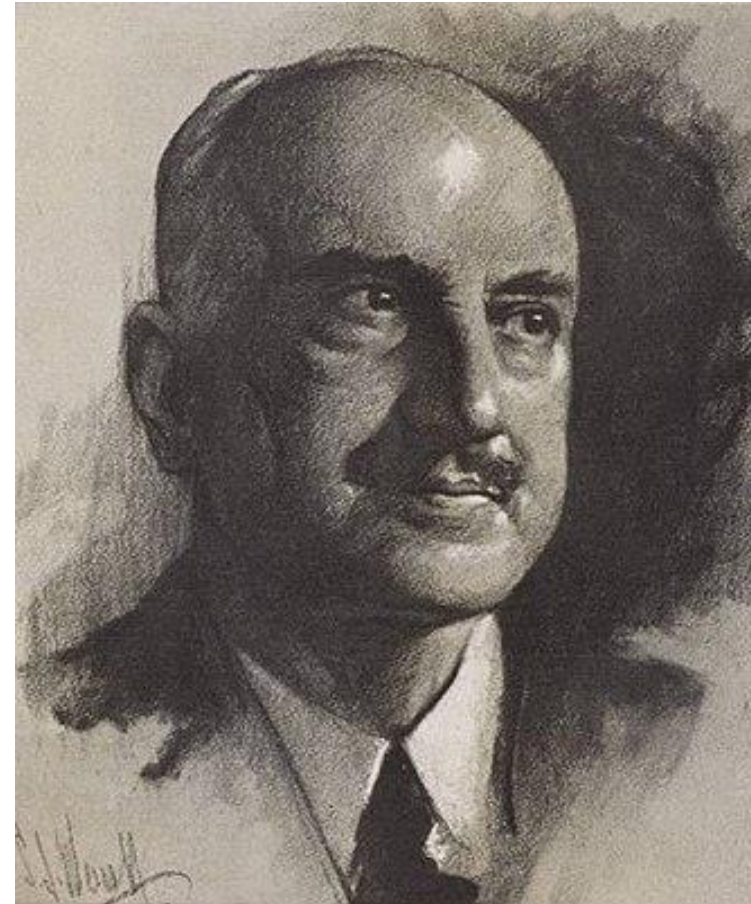
Gartner

George Santayana



George Santayana (1905) Reason in Common Sense, p. 284, volume 1
of The Life of Reason

"Those who cannot
remember the past are
condemned to repeat it"



BIG DATA

- NoSQL
- MAP REDUCE
-



