

Cours

Mise en œuvre des Framework

D'Intelligence Artificielle et Big data

Cycle Ingénieur
INDIA, Semestre 5

Pr. Abderrahim El Qadi
Département Mathématique Appliquée et Génie Informatique
ENSAM, Université Mohammed V de Rabat

A.U. 2023/2024

3eme partie

6. Hive

7. PIG & OOZIE

8. Scoop

Références

- <https://www.simplilearn.com/tutorials/hadoop-tutorial/>
- SQL to Hive Cheat Sheet ([hortonworks.com](https://www.hortonworks.com))
- <https://www.tutorialspoint.com/hive/>
- https://www.tutorialspoint.com/apache_pig/
- <https://www.edureka.co/blog/pig-tutorial/>
- <https://meritis.fr/big-data-analyse-donnees-apache-hiv>

6. Hive



- Apache Hive est un data warehouse pour Hadoop
- Il a été développé par Facebook, puis l'Apache Software Foundation l'a repris et l'a développé en open source sous le nom d'Apache Hive.
- Il ne s'agit pas d'une base de données relationnelle ni d'un data warehouse classique
- Il s'agit d'un système qui maintient des métadonnées décrivant les données stockées dans HDFS.
- Il utilise une base de données relationnelle appelée metastore (**Derby** par défaut) pour assurer la persistance des métadonnées.
- Une table dans Hive est composée essentiellement :
 - D'un schéma stocké dans le metastore,
 - De données stockées dans HDFS.

- Afin de faciliter l'analyse de données stockées dans HDFS sans passer par la complexité de MapReduce, certains frameworks comme **Pig**, **Hive** sont apparus. Ils proposent des langages de haut niveau pour lancer des requêtes ad-hoc sur HDFS.
- Chaque requête écrite par l'utilisateur est convertie en code MapReduce qui interagit ensuite avec HDFS.



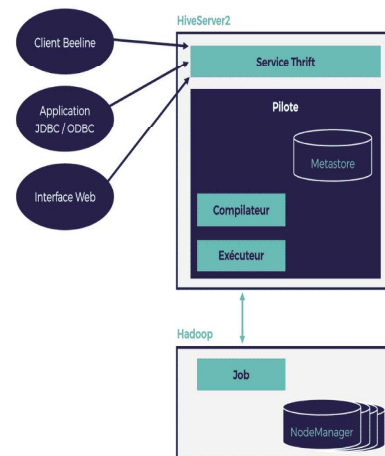
Différence entre Hive et SGBDR

Hive	SGBDR
Hive applique le schéma lors de la lecture	Le SGBDR applique le schéma à l'écriture
La taille des données Hive est exprimée en pétaoctets	La taille des données est exprimée en téraoctets
Hive est basé sur la notion d'écrire une fois et de lire plusieurs fois	Le SGBDR est basé sur la notion de lecture et d'écriture plusieurs fois
Hive ressemble à un entrepôt de données	Le SGBDR est un type de système de gestion de base de données, qui est basé sur le modèle relationnel des données

6.1. Architecture de Hive

L'interaction Hive/Hadoop s'effectue selon les trois étapes suivantes :

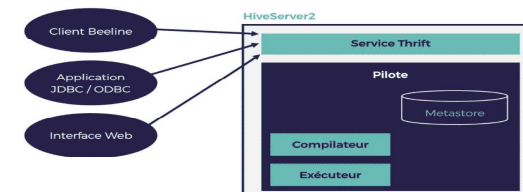
1. **Envoi de la requête HiveQL** : en utilisant un client Hive (le client shell, un client JDBC/ODBC ou une interface web), la requête est envoyée au serveur Hive,
2. **Planification de la requête** : la requête est reçue par le driver (pilote). Elle est compilée, optimisée et planifiée comme un job,
3. **Exécution du job** : le job est exécuté sur le cluster Hadoop.



Partie (1) : la partie client

Il est possible de **soumettre des requêtes au serveur Hive de différentes manières**. En utilisant :

- Le **client Hive CLI** (Hive Command Line Interface) qui permet d'entrer des commandes directement depuis le shell hive ou d'exécuter un ensemble de commandes Hive écrites dans un fichier texte.
 - Ce client n'est pas compatible avec la nouvelle version de Hive (HiveServer2) et a été remplacé par **Beeline** qui est le nouveau client en mode ligne de commande de Hive.
 - Il communique avec HiveServer2 via thrift,
- Le **client JDBC/ODBC**,
- Le **client web**.



Partie (2) : la partie serveur

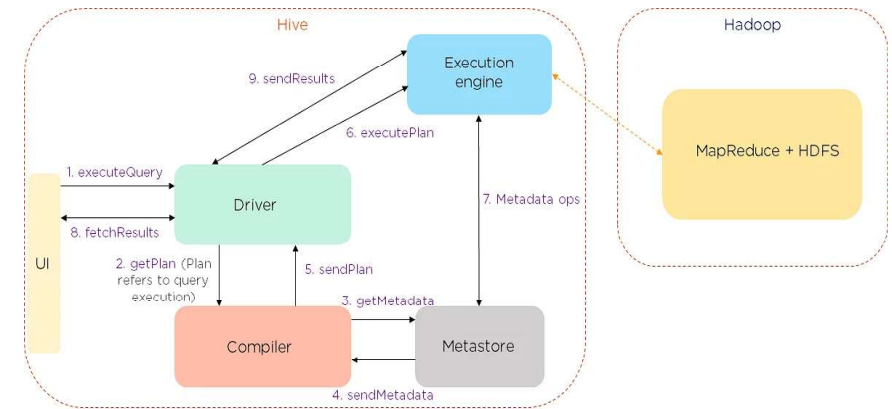
- Appelé **HiveServer2** qui succède à HiveServer.
- Il s'agit du **conteneur du moteur d'exécution de Hive** et appelé couramment **pilote** (ou driver).
- Il se compose du **metastore**, du **compilateur** et de **l'exécuteur**.
- HiveServer2 assure deux nouvelles fonctionnalités : la gestion de l'authentification client et la gestion des requêtes concurrentes.
 - Pour **chaque connexion client, HiveServer2 crée un nouveau contexte d'exécution** (connexion + session).
 - La nouvelle interface RPC de HiveServer2 permet au serveur d'associer le contexte d'exécution Hive avec le thread qui sert la requête client. Cette interface implémente un service thrift pour communiquer avec les clients et exécuter leurs requêtes.

Partie (3) : Hadoop

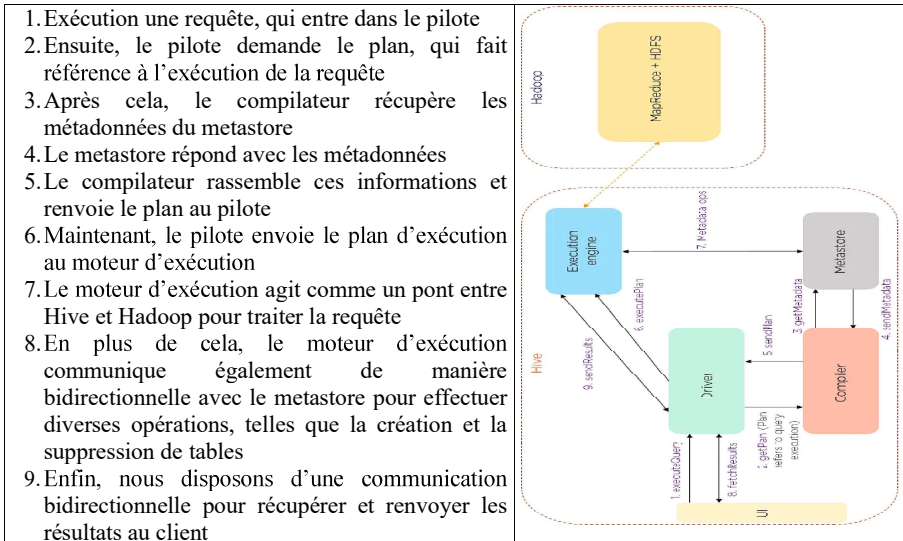
- Correspond à **l'exécution du job sur le cluster Hadoop**.
 - Tables, partitions et requêtes dans Hive

6.2. Flux de données dans Hive

- Le flux de données dans Hive contient le système Hive et Hadoop.
- Sous l'interface utilisateur, Il y a le pilote, le compilateur, le moteur d'exécution et le metastore.



- Le flux de données se déroule dans l'ordre suivant :



6.3. Modélisation des données Hive

- La modélisation des données Hive se compose de tables, de partitions et de compartiments :
 - Tables : les tables dans Hive sont créées de la même manière que dans le SGBDR.
 - Partitions : ici, les tables sont organisées en partitions pour regrouper des types de données similaires en fonction de la clé de partition
 - Compartiments : les données présentes dans les partitions peuvent être divisées en compartiments pour une interrogation efficace.



6.3.1. Tables et chargement des données dans Hive

- Une **table dans Hive permet d'associer une structure à des données non structurées dans HDFS**.
- La création d'une table dans Hive est similaire à la création d'une table dans un RDBMS et s'effectue avec la commande CREATE TABLE.
- Il existe dans Hive deux types de tables :
 - 1) Managed table,
 - 2) External table.
- La différence entre une Managed table et une External table est la gestion des données lorsque la table est supprimée.

1. Managed Table

- La table est similaire à une table au sens RDBMS
- Dans ce type de tableau, nous devons d'abord créer un tableau et charger les données.
- Nous pouvons appeler celui-ci comme **données sur le schéma**.
- En supprimant cette table, les données et le schéma seront supprimés.
- L'emplacement stocké de cette table sera dans /user/hive/warehouse.
- **Quand choisir une table interne ?**
 - Si les données de traitement disponibles dans le système de fichiers local
 - Si on veut que Hive gère le cycle de vie complet des données, y compris la suppression.

Exemple d'extrait de code pour la table interne

1. Création la table interne

```
CREATE TABLE livre_interne (  
    codelivre int,  
    isbn string,  
    titre string,  
    auteur string,  
    editeur string,  
    dateedition date,  
    prixunite float)  
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' ;
```

2. Chargement des données dans la table interne

```
LOAD DATA INPATH '/user/root/input/dataLivre.csv' INTO table livre_interne;
```

3. Suppression de la table interne

```
DROP TABLE livre_interne;
```

Après la suppression de la table, ses métadonnées et ses données seront supprimées de Hive.

2. External Table

- La table externe est de nature faiblement couplée.
- Les données seront disponibles dans HDFS.
- Le tableau va être créé sur les données HDFS.
- En d'autres termes, c'est comme si cela créait **schéma sur les données**.
- Au moment de la suppression de la table, seul le schéma est supprimé, les données seront toujours disponibles dans HDFS comme auparavant.
- Les tables externes offrent la possibilité de créer plusieurs schémas pour les données stockées dans HDFS au lieu de supprimer les données à chaque fois que le schéma est mis à jour.
- **Quand choisir une table externe ?**
 - Si les données de traitement sont disponibles dans HDFS
 - Utile lorsque les fichiers sont utilisés en dehors de Hive

1. Création la table externe

```
CREATE EXTERNAL TABLE livre_externe (  
    codelivre int,  
    isbn string,  
    titre string,  
    auteur string,  
    editeur string,  
    dateedition date,  
    prixunite float)  
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\';  
LOCATION '/user/hive/data/db';
```

2. Chargement des données dans la table externe

```
root@c25780d97f98:/tmp# hdfs dfs -mkdir -p /user/hive/data/db  
root@c25780d97f98:/tmp# hdfs dfs -put dataLivre.csv /user/hive/data/db
```

Si pas de données au moment de la création de la table, possibilité de charger les données manuellement

```
LOAD DATA INPATH '/user/root/input/dataLivre.csv' INTO table livre_externe;
```

6.3.3. Langage HiveQL

- C'est un langage de requête d'Apache Hive pour le traitement et l'analyse de données structurées.
- La syntaxe HiveQL est similaire à SQL
- Hive utilise une base de données Derby pour le stockage des métadonnées d'un seul utilisateur, et pour les métadonnées de plusieurs utilisateurs ou les cas de métadonnées partagées, Hive utilise MySQL.
- Syntaxe d'une requête HQL :

```
SELECT [ALL | DISTINCT] select_expr, select_expr, ...  
FROM table_reference  
[WHERE where_condition]  
[GROUP BY col_list]  
[HAVING having_condition]  
[CLUSTER BY col_list | [DISTRIBUTE BY col_list] [SORT BY col_list]]  
[LIMIT number];
```

6.3.2. Gestion des partitions

- Partitionner une table dans Hive implique une séparation des fichiers selon la colonne (ou les colonnes) définissant la clé de partition.
- Le **partitionnement peut améliorer les performances des requêtes HiveQL** puisque les fichiers dans HDFS sont déjà séparés en se basant sur la valeur de la colonne.
- Cette séparation peut **réduire le nombre de mappers** et réduire ainsi le nombre des opérations de shuffle/sort du job résultant.
- La définition d'une partition est similaire à sa définition en SQL :

```
CREATE TABLE product_partitioned (  
    productId INT,  
    productName STRING,  
    valuationDate TIMESTAMP,  
    validTillDate TIMESTAMP)  
partitioned by (productType  
STRING);
```

Dans HDFS, nous aurons une structure de données en sous-répertoire par productType sous
/apps/hive/warehouse/product_partitioned :
/apps/hive/warehouse/product_partitioned
/productType=bond/
/productType=future/
/productType=forward/
/productType=option/

- Limites de Hive :
 - Hive n'est pas conçu pour le traitement des transactions en ligne (OLTP), il n'est utilisé que pour le traitement analytique en ligne.
 - Dans Hive, les sous-requêtes ne sont pas prises en charge.

7. Apache Pig

- Il s'agit d'un outil/plate-forme qui est utilisé pour analyser de plus grands ensembles de données en les représentant sous forme de flux de données.



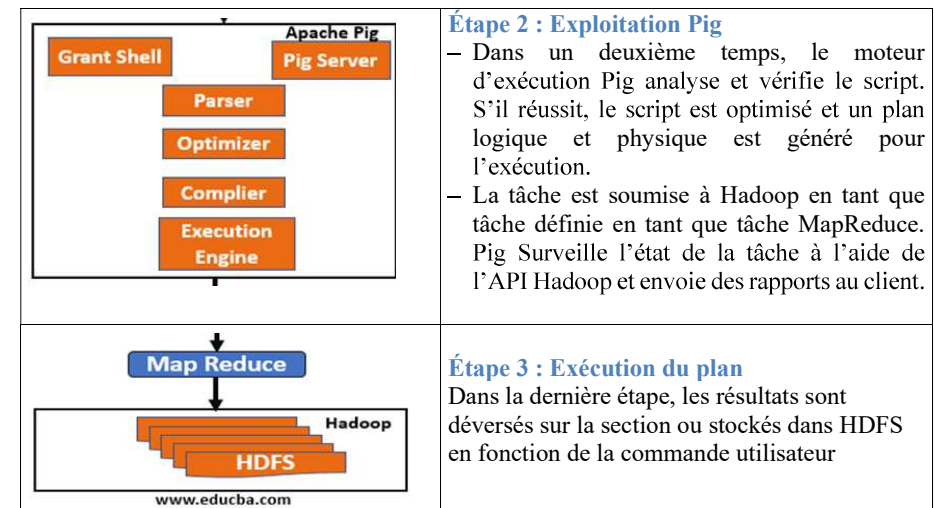
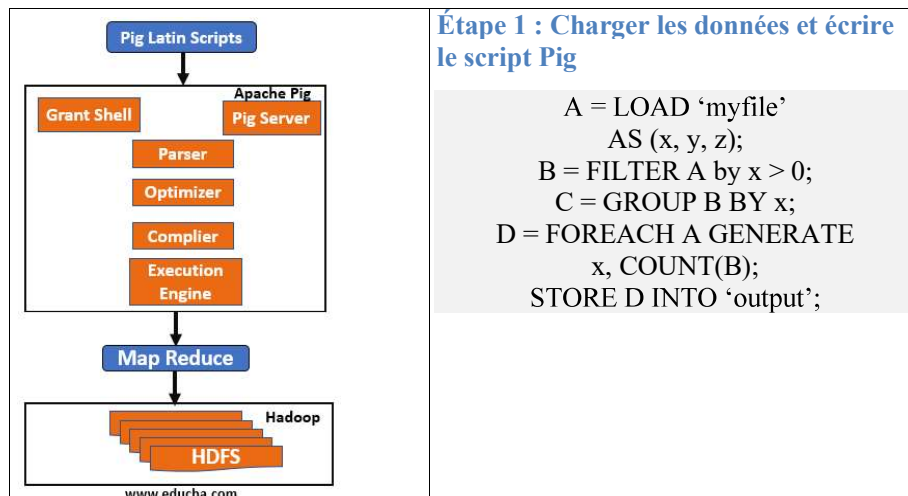
- Pig est extensible dans le sens où, si une fonction n'est pas disponible, il est possible de l'enrichir via des développements spécifiques dans un langage bas niveau (Java, Python...).
- Pour écrire des programmes d'analyse de données, Pig fournit un langage de haut niveau connu sous le nom de **Pig Latin**.

Caractéristiques de Pig

Apache Pig est livré avec les fonctionnalités suivantes :

- **Riche ensemble d'opérateurs** – Il fournit de nombreux opérateurs pour effectuer des opérations telles que jointure, tri, fichier, etc.
- **Facilité de programmation** – Pig Latin est similaire à SQL
- **Possibilités d'optimisation** – Les tâches d'Apache Pig optimisent leur exécution automatiquement, de sorte que les programmeurs doivent se concentrer uniquement sur la sémantique du langage.
- **Extensibilité** – À l'aide des opérateurs existants, les utilisateurs peuvent développer leurs propres fonctions de lecture, de traitement et d'écriture de données.
- Gère toutes sortes de données : Apache Pig analyse **toutes sortes de données**, qu'elles soient structurées ou non. Il stocke les résultats dans HDFS.

7.1. Apache Pig – Architecture



7.2. Pig – Modèle de données

- **Pig Latin** est le langage utilisé pour analyser les données dans Hadoop à l'aide d'Apache Pig.
- Lors du traitement des données à l'aide de Pig Latin, **les instructions** sont les constructions de base.
 - Les instructions Pig Latin prennent une relation en entrée et produisent une autre relation en sortie.
 - Dès que on saisit l'instruction **Load** dans le shell Grunt, sa vérification sémantique est effectuée.
 - L'opérateur **Dump** permet d'afficher le contenu du schéma.

Exemple :

```
grunt> Student_data = LOAD 'student_data.txt' USING PigStorage(',') as
(id:int, firstname:chararray, lastname:chararray, phone:chararray,
city:chararray );
```

– Pig Latin – Type Opérateurs/opératrices de construction

Opérateur	Description	Exemple
()	Opérateur de constructeur de tuple – Cet opérateur est utilisé pour construire un tuple.	(Voiture, 120)
{}	Opérateur de constructeur de bags (sacs) – Cet opérateur est utilisé pour construire un sac.	{('développeur', 1500, ('java', 'sql')), ('voiture', 120)}
[]	Opérateur de constructeur de carte – Cet opérateur est utilisé pour construire un tuple.	[marque#Voiture, km#30]

– Pig Latin – Types de données

- Types de données simples : Une valeur atomique simple (int, long, float, double, chararray, bytearray, boolean). Exemple : 'voiture', 10.11
- Types de données complexes :
 - **Tuple** (ensemble ordonné de champs, exemple. ('Voiture', 120))
 - **Bag** (ensemble de tuple), Exemple : {('développeur', 1500, ('java', 'sql')), ('voiture', 120)}
 - **Map** (ensemble de paires clé-valeur).

– Pig Latin – Opérations relationnelles

Opérateur	Description
Chargement et stockage	
LOAD	Pour charger les données du système de fichiers (local/HDFS) dans une relation.
STORE	Permet d'enregistrer une relation avec le système de fichiers (local/HDFS).
Filtrage	
FILTER	Pour supprimer les lignes indésirables d'une relation.
DISTINCT	Pour supprimer les lignes dupliquées d'une relation.
FOREACH, GENERATE	Pour générer des transformations de données basées sur des colonnes de données.
STREAM	Pour transformer une relation à l'aide d'un programme externe.

Regroupement et jonction	
JOIN	Pour joindre deux relations ou plus.
COGROUP	Pour regrouper les données dans deux relations ou plus.
GROUP	Pour regrouper les données dans une seule relation.
CROSS	Créer le produit vectoriel de deux ou plusieurs relations.
Classement	
ORDER	Permet d'organiser une relation dans un ordre trié en fonction d'un ou de plusieurs champs (croissants ou décroissants).
LIMIT	Pour obtenir un nombre limité de tuples à partir d'une relation.
Combinaison et fractionnement	
UNION	Combiner deux relations ou plus en une seule relation.
SPLIT	Pour fractionner une seule relation en deux relations ou plus.

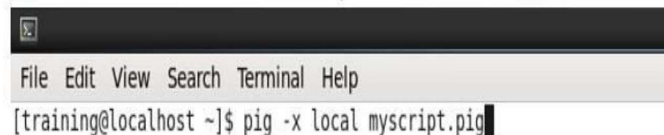
Opérateurs de diagnostic	
DUMP	Permet d'imprimer le contenu d'une relation sur la console.
DESCRIBE	Décrire le schéma d'une relation.
EXPLAIN	Pour afficher les plans d'exécution logiques, physiques ou MapReduce pour calculer une relation.
ILLUSTRATE	Permet d'afficher l'exécution pas à pas d'une série d'instructions.

7.3. Modes d'exécution des Pig

– Pig fonctionne en deux modes d'exécution : Local et MapReduce.

- **Mode local**

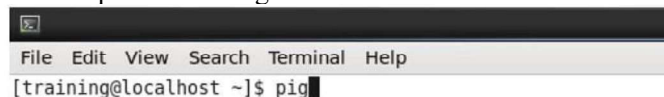
le moteur Pig prend l'entrée du système de fichiers Linux et la sortie est stockée dans le même système de fichiers. Le mode local d'exécution de Pig est expliqué ci-dessous.



```
File Edit View Search Terminal Help
[training@localhost ~]$ pig -x local myscript.pig
```

- **Mode MapReduce**

le moteur Pig interagit et s'exécute directement dans HDFS et MapReduce, comme indiqué dans le diagramme ci-dessous.



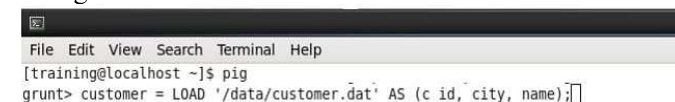
```
File Edit View Search Terminal Help
[training@localhost ~]$ pig
```

– Modes interactifs de Pig

Les deux modes dans lesquels un programme Pig Latin peut être écrit sont Interactif et Batch.

- **Mode interactif**

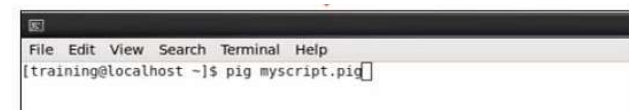
Le mode interactif consiste à coder et à exécuter le script, ligne par ligne, comme le montre l'image ci-dessous.



```
File Edit View Search Terminal Help
[training@localhost ~]$ pig
grunt> customer = LOAD '/data/customer.dat' AS (c_id, city, name);
```

- **Mode batch**

En mode Batch, tous les scripts sont codés dans un fichier avec l'extension. pig et le fichier est directement exécuté comme le montre le schéma ci-dessous.



```
File Edit View Search Terminal Help
[training@localhost ~]$ pig myscript.pig
```


7.4. Pig et SQL

	Pig	SQL
Définition	est un langage de script utilisé pour interagir avec HDFS.	est un langage de requête utilisé pour interagir les bases de données résidant
Style de requête	propose un style d'exécution étape par étape.	propose le style d'exécution d'un seul bloc.
Évaluation	effectue une évaluation différée, ce qui signifie que les données ne sont traitées que lorsque la commande STORE ou DUMP est rencontrée.	offre une évaluation immédiate d'une requête.
Fractionnements de pipelines	Les fractionnements de pipeline sont pris en charge dans Pig.	Il faut exécuter deux fois la commande « join » pour que le résultat soit matérialisé en tant que résultat intermédiaire.

Pig et SQL – Example

Pig	SQL
<pre>customer = LOAD '/data/customer.dat' AS (c_id,name,city); sales = LOAD '/data/sales.dat' AS (s_id,c_id,date,amount); salesBLR = FILTER customer BY city == 'Texas'; joined= JOIN customer BY c_id, salesTAX BY c_id; grouped = GROUP joined BY c_id; summed= FOREACH grouped GENERATE GROUP, SUM(joined.salesTX::amount); spenders= FILTER summed BY \$1 > 2000; sorted = ORDER spenders BY \$1 DESC; DUMP sorted;</pre>	<pre>SELECT c_id , SUM(amount) AS CTotal FROM customers c JOIN sales s ON c.c_id = s.c_id WHERE c.city = 'Texas' GROUP BY c_id HAVING SUM(amount) > 2000 ORDER BY CTotal DESC</pre>