

TP01 - Manipulation de données en SQL sous PostgreSQL

Ce TP consiste à interroger une base de données *PostgreSQL*. Cette base de données est implantée sur le serveur `servbdd`. Il faudra donc utiliser l'URL suivante : « `servbdd.iutlan.etu.univ-rennes1.fr` », ou plus simplement « `servbdd` ». La base de données s'appelle : « `pg_tp1a` ».

Vérification de l'accès à la base et exploration

Connectez vous à cette base à l'aide de la commande :

```
psql -U <monlogin> -h servbdd pg_tp1a
```

Listez les nom des tables(Relations) présentes dans le schéma « `public` » de cette base de données, vous devriez obtenir :

```
      List of relations
Schema | Name      | Type  | Owner
-----+-----+-----+-----
public | client    | table | delhay
public | commande  | table | delhay
public | stock     | table | delhay
(3 rows)
```

Les relations présentées sont la propriété du rôle « `delhay` », mais vous avez le droit de les lire, donc, d'afficher leur valeur. Affichez maintenant le type des relations « `client` », « `commande` » et « `stock` ». Pour la relation clients vous obtenez :

```
pg_tp1a=> \d clients
          Table "public.client"
   Column      |          Type          | Modifiers
-----+-----+-----+-----
 num_cli       | numeric(3,0)           | not null
 nom_cli       | character varying(20)  |
 adresse_cli   | character varying(20)  |
Indexes:
    "client_pk" PRIMARY KEY, btree (num_cli)
Referenced by:
    TABLE "commande" CONSTRAINT "commande_fk2" FOREIGN KEY (num_cli_c) REFERENCES clients(num_cli)
```

Procédez de même pour les relations « `stock` » et « `commande` » et indiquez pour chaque relation quelles sont les clés candidates et étrangères.

Requêtes

Exercice 1 Restrictions, projections, jointures

Configurez ensuite une nouvelle connexion dans SQLWorkbench/J en tenant compte du changement de base : vous vous connectez **avec votre login** mais à la base **pg_tp1a**. Pour travailler, vous pouvez charger dans un onglet *statement* le fichier « `tp01.sql` » que vous avez téléchargé à partir de l'ENT (disponible au même endroit que ce document). Cela permettra de garder une trace de votre travail.

Pour chaque interrogation :

1. Ecrivez la requête SQL correspondante. Prenez soin de reproduire en commentaire l'intitulé de la requête,
2. Soumettez-la à l'interpréteur,
3. Contrôlez la validité des réponses obtenues en regardant la valeur de la base (recommencez si elles ne sont pas valides).

On souhaite obtenir :

1. tous les articles
2. tous les clients
3. toutes les commandes
4. le nom de chaque article
5. le nom et l'adresse de chaque client
6. le nom et le numéro des clients dont l'adresse n'est pas connue (valeur NULL)
7. les articles de type Philosophie
8. le nom, le type des articles de type Philosophie ou Litterature
9. le nom, le type des articles dont la quantité en stock est ≥ 100
10. les articles de type Sciences dont la quantité en stock est < 100
11. les noms des articles de type Sciences dont la quantité en stock est < 100
12. les noms des articles dont la valeur du stock est > 1000
13. le nom, type des articles qui sont en rupture de stock
14. le nom, type des articles qui sont approvisionnés
15. les noms des articles commandés
16. les noms des clients ayant commandé
17. le numéro des clients n'ayant jamais commandé
18. les commandes concernant les articles qui ne sont pas de type Philosophie
19. le nom des articles de type Sciences dont au moins une commande a un montant > 800
20. la valeur du stock
21. la somme des montants des commandes
22. les noms des clients ayant commandé (au moins un) article de type Philosophie
23. les numéros des articles sujets à au moins 2 commandes par deux clients différents
24. les couples de numéros de clients (n1,n2) tels que les clients n1 et n2 soient différents et aient la même adresse.
25. le nombre de clients
26. la quantité minimale en stock
27. la quantité maximale en stock des articles de type Philosophie
28. la moyenne des quantités en stock
29. le nombre de commandes du client n°1
30. le nom des articles commandés par les clients n°4 ou n°2
31. le nom des articles non commandés par le client n°4 et n°2
32. le nom des articles tels qu'il existe au moins une commande de quantité supérieure à celle en stock
33. le numéro des articles qui ne sont commandés qu'une seule fois
34. la moyenne des quantités commandées des articles de type Littérature

Exercice 2 Le partitionnement

Cet exercice porte sur la base de données déjà exploitée dans le TP précédent. La clause *GROUP BY* d'une requête *SQL* permet de faire du partitionnement en regroupant les n-uplets suite à l'évaluation des clauses *FROM* et *WHERE*. Elle peut être complétée par une clause *HAVING* qui définit une condition de restriction ensembliste portant sur les parties obtenues. Les requêtes suivantes sont à écrire en utilisant la clause *GROUP BY*.

36. Par numéro article : la somme et le maximum des quantités commandées.
37. Pour chaque article, son nom, la plus petite quantité commandée, la plus grande et la moyenne des quantités commandées.
38. Le numéro des articles sujets à au moins 2 commandes.
39. Le numéro des articles sujets à au plus 3 commandes.
40. Le numéro des articles sujets à exactement 2 commandes.
41. Le nom des articles dont la somme des quantités commandées excède 100.
42. Le nom et le numéro des clients ayant passé au moins 2 commandes du même produit.
43. Le nom et le numéro des clients ayant passé au moins 2 commandes de produits différents.
44. La moyenne des quantités commandées de chaque article de type 'Littérature'.
45. La moyenne des quantités commandées pour les articles sujets à au moins 3 commandes.
46. Le numéro des articles dont la moyenne des quantités commandées est supérieure à 15.
47. Le nom des clients ayant effectué au moins 3 commandes.
48. Combien de jours séparent la première et la dernière commande de chaque client ?.
49. Quelle est la date de la dernière commande de chaque client ?.

Exercice 3 Requêtes complexes

Formulez en langage *SQL* les requêtes suivantes et exécutez-les en contrôlant la validité du résultat obtenu.

50. le nom des articles dont la quantité en stock est maximale.
51. le nom des articles dont au moins une commande est de quantité commandée supérieure à la quantité en stock.
52. le nom des articles dont la somme des quantités commandées est supérieure à la quantité en stock.
53. le nom des articles tels que toutes les commandes sont de quantité supérieure à celle en stock .
54. le numéro des articles qui ne sont commandés qu'une seule fois.
55. le numéro des clients qui ont commandé tous les livres de Philosophie.
56. le plus ancien client (numéro et nom).
57. le premier client à avoir acheté un exemplaire de '1984'.
58. la dernière commande en date de chaque client.

Exercice 4 manipulation de dates

Les requêtes que l'on demande de formuler dans cet exercice nécessitent l'usage de fonctions permettant de manipuler des expressions de type « date » dans l'environnement *Postgresql*.

Dans une base de données "PostgreSQL", une valeur de type « Date » exprime une date du calendrier, une valeur de type « timestamp » représente l'instant (date + heure) avec une précision de l'ordre de la microseconde alors qu'une valeur de type « time » détermine une heure à la microseconde près. Un ensemble de fonctions est mis à disposition des usagers, permettant l'accès aux éléments constitutifs d'une date (jour, mois, année, heure, minute, secondes...) selon le type employé.

Parmi ces fonctions, on peut citer :

```
to_char(une_date, modèle de format)
to_date(une_chaine, modèle de format)
```

qui permettent de convertir des données de type date au format chaîne de caractères et vice versa. Les modèles de format sont décrits dans la documentation dans la rubrique *fonctions de formatage*.

Ou encore :

```
extract(champ from timestamp)
```

qui permet l'accès aux champs constitutifs d'une date.

Une liste exhaustive de ces fonctions est disponible dans la documentation dans la rubrique *fonctions de traitement de la date et de l'heure*

Exemple d'utilisation de ces fonctions :

Afficher la date des commandes passées par le client numéro 6 selon le format "Mardi 24 janvier 2022".

```
SELECT to_char(date_com, 'day dd month yyyy')
FROM commandes
WHERE num_cli_c = 6;
```

...En Français dans le texte :

```
SELECT to_char(date_com, 'tmday dd tmmonth yyyy')
FROM commandes
WHERE num_cli_c = 6;
```

Si l'on préfère la forme "31/03/2022", la requête deviendra :

```
SELECT to_char(date_com, 'dd/mm/yyyy')
FROM commandes
WHERE num_cli_c = 6;
```

Insérer dans la table "commandes" la commande de 25 unités de l'article numéro 3 par le client numéro 6 passée le Mardi 24 janvier 2022.

```
INSERT INTO commandes
VALUES ( to_date('mardi 24 janvier 2012','day dd month yyyy'), 3, 6, 25);
```

Opérations sur les données de type "date" (liste non exhaustive).

Travail à réaliser :

Formulez les requêtes suivantes en langage SQL :

59. La date d'aujourd'hui selon le format : « lundi 16 décembre 2013 ».
60. L'heure courante sous la forme : « 13 :24 :52 ».
61. Quelles sont les commandes passées en octobre 2013 ?
62. Quelles sont les commandes passées il y a au plus trente jours ?
63. Nom des clients ayant passé des commandes il y a au plus trente jours ?
64. Quelles sont les commandes passées entre le 16/08/2013 et le 16/11/2013 ?
65. Nom des clients ayant passé une commande un vendredi ?
66. Nom des clients ayant passé une commande un jeudi ou un vendredi ?
67. Quels sont les clients (numéro et nom) qui ont passé des commandes au cours du premier trimestre de l'année courante ?
68. Quels sont les clients (numéro et nom) qui n'ont pas passé de commande durant les deux derniers mois (60 jours) ?
69. Quels sont les clients qui ont passé toutes leurs commandes durant le même mois ?