|  |  |
| --- | --- |
| logo_IUT_Annecy_RVB | **Bases de données – TD1 Correction**  **Requêtes SQL (SELECT)** |

*Requêtes simples*

1. Afficher les employés (nom, prénom, adresse, ville) dont le nom commence par un D et habitant Seattle ou Londres. Classement alphabétique.

SELECT nom, prenom, adresse, ville

FROM employe

WHERE ville IN ('Seattle','London')

AND nom LIKE 'D%'

ORDER BY 1, 2;

1. Afficher les employés sans supérieur hiérarchique (i.e. la colonne est vide) et faire afficher au niveau de chaque enregistrement ‘Est chef’.

SELECT nom, prenom, fonction, 'Est chef' AS "Type"

FROM employe

WHERE superieurhierarchique IS NULL;

1. Sachant que les frais de port des commandes sont en $ dans la base, afficher les commandes et leur frais de port en euro (1 €=1.2$). Les dates seront affichées au format français (utilisez la fonction to\_char).

SELECT C.commandeid, Cl.nomsociete, to\_char(C.datecommande,'DD/MM/YYYY') As datecommande, C.fraisports/1.2 AS "frais de port en euros"

FROM Commande C

JOIN Client Cl ON C.clientid=Cl.clientid;

1. Afficher le nom et la fonction des employés ainsi que le nom et la fonction de leur supérieur hiérarchique (respecter le nom des colonnes).

Auto-jointure :

SELECT emp.Nom, emp.Prenom, emp.Fonction, sup.Nom As "Nom supérieur", sup.Prenom As "Prénom supérieur", sup.Fonction As "Fonction supérieur"

FROM Employe emp

JOIN Employe sup ON emp.SuperieurHierarchique = sup.employeid

1. Afficher le nom des employés plus âgés que leur supérieur hiérarchique.

Auto-jointure :

SELECT emp.Nom, emp.Prenom, to\_char(emp.datenaissance, 'DD/MM/YYYY') as datenaiss, sup.Nom As "Nom supérieur", sup.Prenom As "Prénom supérieur", to\_char(sup.datenaissance, 'DD/MM/YYYY') AS "Date naiss. Sup hier."

FROM Employe emp

JOIN Employe sup ON emp.SuperieurHierarchique = sup.employeid

**AND emp.datenaissance < sup.datenaissance**

1. Afficher les clients qui ont acheté des produits du fournisseur ‘Exotic Liquids’.

SELECT DISTINCT Cl.nomsociete

FROM Client Cl

JOIN Commande Co ON Cl.clientid=Co.clientid

JOIN LignesCommande LC ON Co.commandeid=LC.commandeid

JOIN Produit P ON LC.produitid=P.produitid

JOIN Fournisseur F ON P.fournisseurid=F.fournisseurid

WHERE

F.NomFournisseur='Exotic Liquids'

*Requêtes avec jointure externe*

1. Afficher toutes les catégories de produits et les noms de produit associés à chaque catégorie

SELECT C.nomcategorie, P.nomproduit

FROM categorie C

LEFT JOIN produit P ON C.categorieid=P.categorieid

ORDER BY 1,2;

-- LEFT JOIN ou LEFT OUTER JOIN : pareil !!

1. Afficher les territoires et leurs employés (y compris les territoires sans employé)

SELECT T.NomTerritoire, E.Nom

FROM territoire T

LEFT JOIN TerritoireEmploye TE ON T.territoireid=TE.territoireid

LEFT JOIN Employe E ON TE.employeid=E.employeid;

**-- On est obligé de mettre LEFT JOIN partout, sinon on perd le bénéfice de la jointure externe**

*Requêtes avec fonctions d’agrégat*

1. Calculer le nombre de produits hors catalogue (le champ HorsCatalogue est un booléen. Valeurs possible : True, False ou NULL (non renseigné))

SELECT COUNT(\*) AS "Nb pdts hors catalogue"

FROM Produit

WHERE HorsCatalogue=True

OU

SELECT COUNT(ProduitID) AS "Nb pdts hors catalogue"

FROM Produit

WHERE HorsCatalogue=True

1. Nom des catégories et le prix moyen des produits de chaque catégorie.

SELECT C.NomCategorie, AVG(P.PrixUnitaire) AS "Prix moyen"

FROM Produit P

JOIN Categorie C ON P.CategorieID=C.CategorieID

GROUP BY C.NomCategorie

ORDER BY C.NomCategorie;

1. Sélectionner l’identifiant et le nom des sociétés qui fournissent plus de 3 produits.

SELECT P.FournisseurID, F.NomFournisseur

FROM Produit P

JOIN Fournisseur F ON P.FournisseurID=F.FournisseurID

GROUP BY P.FournisseurID, F.NomFournisseur

HAVING COUNT(P.ProduitID)>3;

OU :

SELECT FournisseurID, NomFournisseur

FROM Fournisseur

WHERE FournisseurID IN

(

SELECT FournisseurID

FROM Produit

GROUP BY FournisseurID

HAVING COUNT(\*)>3

);

1. Sélectionner pour chaque employé, l’identifiant, le nom ainsi que le total des ventes réalisées, ordonnée par identifiant d'employé.

SELECT E.EmployeID, E.Nom,

SUM(LC.PrixUnitaire\*LC.Quantite\*(1-LC.Remise)) AS "Total ventes"

FROM Employe E

JOIN Commande C ON E.EmployeID=C.EmployeID

JOIN LignesCommande LC ON C.CommandeID = LC.CommandeID

GROUP BY E.EmployeID, E.Nom

ORDER BY E.EmployeID

1. Sélectionner le nom des employés qui vendent les produits de plus de 7 fournisseurs.

SELECT E.Nom, E.Prenom

FROM Employe E

JOIN Commande C ON E.EmployeID=C.EmployeID

JOIN LignesCommande LC ON C.CommandeID = LC.CommandeID

JOIN Produit P ON LC.ProduitID=P. ProduitID

GROUP BY E.Nom, E.Prenom

HAVING COUNT(DISTINCT P.FournisseurID)>7

*Requêtes imbriquées*

1. Sélectionner le nom des produits qui n’ont jamais été commandés. Répondre à cette requête de 2 façons : avec IN (et/ou NOT IN) et avec EXISTS (ou NOT EXISTS).

NOT IN :

SELECT NomProduit

FROM Produit

WHERE ProduitID NOT IN (

SELECT ProduitID

FROM LignesCommande

);

NOT EXISTS :

SELECT NomProduit

FROM Produit P

WHERE NOT EXISTS (

SELECT 'X'

FROM LignesCommande LC

WHERE LC.produitid=P.produitid

);

1. Sélectionner le nom des produits qui ont été achetés ou vendus par des sociétés basées à Londres ('London'). Ecrire deux versions de la requête, une avec UNION et l'autre avec OR et IN.

UNION :

SELECT P.NomProduit

FROM Client Cl

JOIN Commande Co ON Cl.ClientID=Co.ClientID

JOIN LignesCommande LC ON Co.CommandeID=LC.CommandeID

JOIN Produit P ON LC.ProduitID=P.ProduitID

WHERE Cl.Ville='London'

UNION

SELECT P.NomProduit

FROM Produit P

JOIN Fournisseur F ON P.FournisseurID=F.FournisseurID

WHERE F.Ville='London'

-- pas besoin de DISTINCT car UNION supprime les doublons

IN :

SELECT NomProduit

FROM Produit

WHERE ProduitID IN

(

SELECT LC.ProduitID

FROM LignesCommande LC

JOIN Commande Co ON LC.CommandeID=Co.CommandeID

JOIN Client Cl ON Co.ClientID=Cl.ClientID

WHERE Cl.Ville='London'

)

OR FournisseurID IN

(

SELECT FournisseurID

FROM Fournisseur

WHERE Ville='London'

)

1. Noms et prénoms des employés qui sont plus âgés que :
2. un employé quelconque qui habite Londres (ANY)

Any :

SELECT Nom, Prenom

FROM Employe

WHERE DateNaissance < ANY

(

SELECT DateNaissance

FROM Employe

WHERE Ville = 'London'

)

On peut également l’écrire avec une Auto-jointure :

SELECT DISTINCT emp1.Nom, emp1.Prenom

FROM Employe emp1

JOIN Employe emp2 ON emp1.DateNaissance<emp2. DateNaissance

WHERE emp2.Ville = 'London'

(B) tous les employés qui habitent Londres (vous utiliserez MIN puis ALL)

All :

SELECT Nom, Prenom

FROM Employe

WHERE DateNaissance < ALL

(

SELECT DateNaissance

FROM Employe

WHERE Ville = 'London'

)

Min :

SELECT Nom, Prenom

FROM Employe

WHERE DateNaissance <

(

SELECT MIN(DateNaissance)

FROM Employe

WHERE Ville ='London'

)

1. Afficher les employés (nom, prénom) qui ont la même fonction que ‘Dodsworth Anne’

SELECT nom, prenom

FROM Employe

WHERE fonction = (

SELECT fonction

FROM Employe

WHERE nom = 'Dodsworth'

AND prenom = 'Anne'

)

*Requêtes avec fonctions scalaires*

1. Afficher le jour de naissance de chaque employé (TO\_CHAR, CASE, CAST)

SELECT Nom, Prenom, datenaissance, CASE cast(to\_char(datenaissance,'D') as integer)

WHEN 1 THEN 'Dimanche'

WHEN 2 THEN 'Lundi'

WHEN 3 THEN 'Mardi'

WHEN 4 THEN 'Mercredi'

WHEN 5 THEN 'Jeudi'

WHEN 6 THEN 'Vendredi'

WHEN 7 THEN 'Samedi'

END AS "Jour date naissance"

FROM employe

Le CAST n’est pas obligatoire ici, sauf si on compare par rapport à un char, i.e. si l’on met WHEN '1' THEN… on n’a pas besoin de caster (=> utilisation des quotes).

SELECT Nom, Prenom, datenaissance, CASE to\_char(datenaissance,'D')

WHEN '1' THEN 'Dimanche'

WHEN '2' THEN 'Lundi'

WHEN '3' THEN 'Mardi'

WHEN '4' THEN 'Mercredi'

WHEN '5' THEN 'Jeudi'

WHEN '6' THEN 'Vendredi'

WHEN '7' THEN 'Samedi'

END AS "Jour date naissance"

FROM employe

1. Afficher les produits ayant un nom composé de plus de 20 caractères ou dont le nom commence par ‘ch’ (majuscule ou minuscule) (utilisez les fonctions LENGTH, UPPER/LOWER et SUBSTRING).

SELECT \*

FROM Produit

WHERE UPPER(SUBSTRING(nomproduit, 1,2))='CH'

OR LENGTH(nomproduit)>20

1. Afficher le CA arrondi par client (utiliser la fonction ROUND). Cf. Q12 pour le calcul.

SELECT Cl.nomsociete,

ROUND(SUM(LC.PrixUnitaire\*LC.Quantite\*(1-LC.Remise))) AS "CA arrondi"

FROM client Cl

JOIN Commande Co ON Cl.clientid=Co.clientid

JOIN LignesCommande LC ON Co.commandeid=LC.Commandeid

GROUP BY Cl.nomsociete;

1. Reprendre la question précédente et :

* Pour les CA < 10 000 $, afficher ‘Petit client’ ;
* Pour les CA compris entre 10 000 et 100 000 (non compris), afficher ‘Bon client’ ;
* Pour les budgets >= 100 000, afficher ‘Excellent client’ (utiliser la fonction CASE).

SELECT Cl.nomsociete,

ROUND(SUM(LC.PrixUnitaire\*LC.Quantite\*(1-LC.Remise))) AS "CA arrondi",

CASE

WHEN ROUND(SUM(LC.PrixUnitaire\*LC.Quantite\*(1-LC.Remise))) <10000 THEN 'Petit client'

WHEN ROUND(SUM(LC.PrixUnitaire\*LC.Quantite\*(1-LC.Remise))) >=10000 AND ROUND(SUM(LC.PrixUnitaire\*LC.Quantite\*(1-LC.Remise))) <100000 THEN 'Bon client'

ELSE 'Excellent client'

END AS "Type client"

FROM client Cl

JOIN Commande Co ON Cl.clientid=Co.clientid

JOIN LignesCommande LC ON Co.commandeid=LC.Commandeid

GROUP BY Cl.nomsociete;

La requête suivante ne marche pas car PostgreSQL attend qqchose avant < :

SELECT Cl.nomsociete,

ROUND(SUM(LC.PrixUnitaire\*LC.Quantite\*(1-LC.Remise))) AS "CA arrondi",

CASE ROUND(SUM(LC.PrixUnitaire\*LC.Quantite\*(1-LC.Remise)))

WHEN <10000 THEN 'Petit client'

WHEN >=10000 AND <100000 THEN 'Bon client'

ELSE 'Excellent client'

END AS "Type client"

FROM client Cl

JOIN Commande Co ON Cl.clientid=Co.clientid

JOIN LignesCommande LC ON Co.commandeid=LC.Commandeid

GROUP BY Cl.nomsociete;

*Requêtes complexes*

1. Sélectionner le nom des produits vendus par tous les employés.

⬄ Nom des produits pour lesquels il n’existe pas d’employés qui n’a pas commandé ces produits

⬄ Nom des produits pour lesquels il n’existe pas d’employés pour lesquels il n’existe pas de commandes de ces produits par ces employés

SELECT P.nomproduit

FROM produit P

WHERE P.produitid IN (

SELECT LC.produitid

FROM commande C

JOIN Lignescommande LC ON C.commandeid=LC.Commandeid

GROUP BY LC.produitid

HAVING COUNT(DISTINCT C.employeid)= (select COUNT(\*) FROM employe)

)

ORDER BY P.nomproduit;

OU

SELECT P.nomproduit

FROM produit P

WHERE NOT EXISTS (

SELECT 'X'

FROM employe E

WHERE NOT EXISTS (

SELECT 'X'

FROM commande C

JOIN Lignescommande LC ON C.commandeid=LC.Commandeid

WHERE C.employeID=E.employeID

AND LC.produitid=p.produitid

)

)

ORDER BY P.nomproduit;

1. Sélectionner le nom des clients et le nom des produits, pour les clients qui ont acheté de ce produit, lors d’une commande, 5 fois plus que la moyenne des ventes de ce produit.

SELECT DISTINCT Cl.Nomsociete, P.Nomproduit

FROM Client Cl

JOIN Commande Co ON Cl.clientid=Co.clientid

JOIN Lignescommande LC ON Co.commandeid=LC.commandeid

JOIN Produit P ON LC.produitid = P.produitID

WHERE LC.Quantite > (

SELECT 5\*AVG(Quantite)

FROM Lignescommande LC2

WHERE LC.ProduitID = LC2.ProduitID

)

ORDER BY Cl.Nomsociete, P.Nomproduit

1. Sélectionner le nom des clients qui ont acheté tous les produits achetés par le client dont l'identificateur est ‘LAZYK'

SELECT Cl.clientid, Cl.Nomsociete

FROM Client Cl

WHERE Cl.clientid <> 'LAZYK'

AND NOT EXISTS (

SELECT 'X'

FROM LignesCommande LC

WHERE LC.ProduitID IN (

SELECT LC2.ProduitID

FROM Commande C

JOIN LignesCommande LC2 ON C.commandeid = LC2.commandeid

WHERE C.clientid = 'LAZYK'

)

AND NOT EXISTS (

SELECT 'X'

FROM Commande C2

JOIN LignesCommande LC3 ON C2.commandeid = LC3.commandeid

WHERE Cl.clientid = C2.Clientid

AND LC.ProduitID = LC3.ProduitID

)

)

ORDER BY Cl.clientid;

OU

SELECT Cl.clientid, Cl.Nomsocietes

FROM Client Cl

WHERE ClientID<>'LAZYK'

AND NOT EXISTS (

SELECT 'X'

FROM Produit P

WHERE ProduitID IN (Select ProduitID

FROM LignesCommande LC

JOIN Commande Co ON LC.commandeid=Co.Commandeid

WHERE Co.ClientID='LAZYK')

AND NOT EXISTS (

SELECT 'X'

FROM LignesCommande LC

JOIN Commande C ON LC.commandeid=C.Commandeid

WHERE C.ClientID=Cl.ClientID

AND LC.ProduitID=P.ProduitID))

1. Afficher le nom des clients qui ont acheté **exactement** les mêmes produits que la société dont l'identificateur est 'LAZYK'.

SELECT Cl.ClientID, Cl.NomSociete

FROM Client Cl

WHERE Cl.ClientID <> 'LAZYK'

AND NOT EXISTS (

SELECT 'X'

FROM Commande Co1

JOIN LignesCommande LC1 ON Co1.commandeid=LC1.commandeid

WHERE Co1.ClientID = 'LAZYK'

AND NOT EXISTS (

SELECT 'X'

FROM Commande Co2

JOIN LignesCommande LC2 ON Co2.commandeid=LC2.commandeid

WHERE Co2.clientid=Cl.clientID

AND LC2.ProduitID = LC1.ProduitID

)

)

AND NOT EXISTS (

SELECT 'X'

FROM Commande Co3

JOIN LignesCommande LC3 ON Co3.commandeid=LC3.commandeid

WHERE Cl.clientid = Co3.clientid

AND NOT EXISTS (

SELECT 'X'

FROM Commande Co4

JOIN LignesCommande LC4 ON Co4.commandeid=LC4.commandeid

WHERE Co4.clientid = 'LAZYK'

AND LC4.ProduitID = LC3.ProduitID

)

)

ORDER BY Cl.clientid;

OU En se basant sur la Q24

SELECT clientid

FROM commande Co

JOIN lignescommande LC ON Co.commandeid=LC.commandeid

WHERE clientid <> 'LAZYK'

and clientid IN (

SELECT clientid

FROM commande Co

JOIN lignescommande LC ON Co.commandeid=LC.commandeid

WHERE produitid IN (

SELECT produitid

FROM commande Co

JOIN lignescommande LC ON Co.commandeid=LC.commandeid

WHERE clientid='LAZYK'

)

GROUP BY clientid

HAVING COUNT(DISTINCT produitid) =(

SELECT COUNT(DISTINCT produitid)

FROM commande Co

JOIN lignescommande LC ON Co.commandeid=LC.commandeid

WHERE clientid='LAZYK'

)

)

GROUP BY clientid

HAVING COUNT(DISTINCT produitid)= (

SELECT COUNT(DISTINCT produitid)

FROM commande Co

JOIN lignescommande LC ON Co.commandeid=LC.commandeid

WHERE clientid='LAZYK'

)

OU

SELECT Cl.ClientID, Cl.Nomsociete

FROM Client Cl

JOIN Commande Co ON Cl.ClientID=Co.ClientID

JOIN LignesCommande LC ON Co.CommandeID=LC.CommandeID

WHERE Cl.ClientID IN (SELECT Cl.clientid

FROM Client Cl

WHERE ClientID<>'LAZYK'

AND NOT EXISTS (

SELECT 'X'

FROM Produit P

WHERE ProduitID IN (Select ProduitID

FROM LignesCommande LC

JOIN Commande Co ON LC.commandeid=Co.Commandeid

WHERE Co.ClientID='LAZYK')

AND NOT EXISTS (

SELECT 'X'

FROM LignesCommande LC

JOIN Commande C ON LC.commandeid=C.Commandeid

WHERE C.ClientID=Cl.ClientID

AND LC.ProduitID=P.ProduitID)))

GROUP BY CL.ClientID, Cl.Nomsociete

HAVING count(distinct LC.produitid) = (SELECT COUNT(DISTINCT produitid)

FROM commande Co

JOIN lignescommande LC ON Co.commandeid=LC.commandeid

WHERE clientid='LAZYK')