TP SQL avancé

Pour chaque question, un aperçu du résultat désiré est représenté à la suite de la requête.

Quelques fonctions SQL pour Postgres:

Environnement

Les requêtes nécessitant des extensions du group by et la clause OVER() ne peuvent être formulées sur les versions trop ancienne de Postgres (ex: la version installée par défaut au département).Une solution est d'installer la dernière version de postgres, en utilisant Docker par exemple, comme indiqué dans les docs.

1 Premiers pas

Lab. Ex 2.1

Chargez les données en exécutant le script créant la base dans psql. Un schéma de la Base de donnée Northwind est fourni en Figure 1 ci-dessous. Vérifier depuis le client postgresql (en affichant la liste des tables et le schéma de quelques tables) que le schéma est bien correct. *Ce fichier est en fait un portage sous Oracle de la base "Northwind traders' conçue par Microsoft pour illustrer les possibilités d'Access.*

Lab. Ex 2.2

Quelle est la table de Faits? Y-a-t-il une dimension récursive (c'est-à-dire, qui se fait référence à elle-même)?

2 Extensions OLAP de la clause GROUP BY

Remarque: sauf mention contraire, "pays"... signifie "shipping country" (qui se trouve être identique à "customer country").

Lab. Ex 2.3

Rédiger des requêtes SQL pour calculer les expressions suivantes. Vous n'êtes pas autorisés à utiliser l'UNION de requêtes.

1. nombre de clients (customers) par pays.
2. nombres de commandes (orders) par pays, (pays et ville), et au total. Trier le résultat par ordre alphabétique sur pays puis ville.

SHIP_COUNTRY	SHIP_CITY	NBORDERS
Argentina	Buenos Aires	16
Argentina		16
Austria	Graz	30
Austria	Salzburg	10
Austria		40
Belgium	Bruxelles	7
...		

Ex1.3.2 (gauche)

C_COUNTRY	S_COUNTRY	QUANTITY	NBORDER
Argentina	Australia	13	2
Argentina	Canada	10	1
Argentina	Denmark	3	1
...			

Ex1.3.3 (dessus)

3. nombre de commandes et quantité d'éléments envoyés (selon la table `OrderDetails`) pour chaque paire (pays Client, pays Fournisseur (supplier)). Trier le résultat par pays du client d'abord, du fournisseur ensuite.
4. nombre de commandes à tous les niveaux de détail quand on s'intéresse seulement à l'origine géographique des clients et fournisseurs, au niveau des pays et du total seulement. (i.e., pareil que précédent mais avec en plus les totaux au niveau de chaque type de pays, et grand total).

C_COUNTRY	S_COUNTRY	QUANTITY	NBORDER
Argentina	Australia	13	2
...			
Argentina		339	16
...			
	USA	6828	244
		51317	830

5. prix total (Quantity* UnitPrice * (1-Discount)) des commandes avec un fournisseur francais, pour chaque pays, region, et ville. Le pays doit être affiché chaque fois que la région l'est, et pareillement la région chaque fois que la ville l'est. On ne veut pas afficher le total. Proposer 2 solutions; s'appuyant sur une fonction différente pour étendre le GROUP BY.

SHIP_COUNTRY	SHIP_REGION	SHIP_CITY	PRICE
UK		London	4452
UK			4452
UK	Essex	Colchester	510
...			
Switzerland			3458.4

Ex1.3.5 (dessus)

S_COUNTRY	S_CITY	NBORDERS
Argentina	Buenos Aires	16
Argentina	whole country	16
Austria	Graz	30
...		
Venezuela	whole country	46
		830

Ex1.3.6 (right)

6. modifier votre requête de question 2 afin d'afficher la chaine 'whole country' au lieu de NULL pour chaque ligne additionnant les nombres d'ordres de toutes les villes d'*un* pays.

3 Fenêtres de partitionnement

Lab. Ex 2.4

Ecrire des requêtes SQL calculant les expressions suivantes.

1. nombres de commandes par pays et ville sur une colonne, ainsi que sur d'autres colonnes nombre total de commandes sur le pays et nombres maximal de commandes réalisées sur une ville de ce pays.

SHIP_COUNTRY	SHIP_CITY	NBORDERS	NBORDCTY	NBORMAXCTY
Argentina	Buenos Aires	16	16	16
Austria	Graz	30	40	30
Austria	Salzburg	10	40	30
...				
70 rows selected.				

2. villes triées à l'intérieur d'un pays par ordre de commande, en affichant ce nombre de commandes et le rang. On ne veut pas sauter de valeurs pour le rang en cas d'ex-aequo.

SHIP_COUNTRY	SHIP_CITY	NBORDERS	RANK
Argentina	Buenos Aires	16	1
Austria	Salzburg	10	1
Austria	Graz	30	2
Belgium	Bruxelles	7	1
Belgium	Charleroi	12	2
...			
70 rows selected.			

3. ajouter à la requête précédente le pourcentage du nombre d'ordre réalisé par la ville à l'intérieur du pays.

SHIP_COUNTRY	SHIP_CITY	NBORDERS	RANK	PERCENTG
Argentina	Buenos Aires	16	1	1.00
Austria	Salzburg	10	1	.25
Austria	Graz	30	2	.75
Belgium	Bruxelles	7	1	.37
...				

Ex. 1.4.3(gauche)

ORDER_ID	PRICE
...	
11071	484.5
11073	300
11074	232.085

Ex. 1.4.4(droite)

4. prix total de chaque commande, en éliminant toutes les commandes dont le prix dépasse 110% du prix la précédente (OrderID) (vous avez le droit d'imbriquer des requêtes).

5. produits les plus vendus par année, avec la quantité. Proposer une réponse utilisant une clause de fenêtrage (partition by) et une autre sans (vous pouvez imbriquer des requêtes).

```
      YEAR PRODUCT_NAME                                QTITY
---------- ---------------------------------------- ----------
      1998 Konbu                                           659
      1997 Gnocchi di nonna Alice                          971
      1996 Gorgonzola Telino                               444
```

4 Requêtes récursives

Lab. Ex 2.5
Utiliser une requête récursive pour créer une table listant les entiers de 1 à 60.

Lab. Ex 2.6 (Requêtes récursives: hiérarchies)

Ecrire une requête récursive, basée sur un CTE, qui affiche pour chaque employé:

- pour chaque employé, indenter l'employé en fonction de son niveau hiérarchique (ex: le grand patron n'a aucune indentation, un employé à distance 2 du grand patron aura une indentation de 4 espaces, etc.)
- pour chaque employé, un attribut de type textuel affichera son chemin jusqu'au grand patron
- les employés sont affichés selon l'ordre préfixe (ordre d'apparition dans un parcours en profondeur).

Lab. Ex 2.7 (Question facultative, pour aller plus loin si vous avez terminé le TP)
Dans la requête précédente, étudier ce qu'il se passe en présence de cycles.

Lab. Ex 2.8 (Requêtes récursive (indépendemment de toute hiérarchie))

Ci-dessous deux exemples de requêtes récursives. Tous deux sont un peu artificiels, mais vous avez déjà vu des exemples d'usage réalistes en cours, mettant en jeu les hiérarchies. La deuxième question n'est pas facile car elle combine clause `OVER()` et CTEs.

1. (un peu trivial, juste un échauffement) Donner le terme u_{50} de la suite de Syracuse $(u_n)_{n\geq 0}$ définie ci-dessous: $u_0 = 127$, $\begin{cases} u_{n+1} = & u_n/2 \text{ si } u_n \text{ est pair} \\ u_{n+1} = & 3 * u_n/2 + 1 \text{ si } u_n \text{ est impair} \end{cases}$

   On pourra utiliser les fonctions modulo et la troncation à 0 décimales près: `MOD(5,2)=1, TRUNC(3.5,0)=2`

2. On se donne un grand plateau de jeu, sous forme de grille (ex: un échiquier). Une relation `D` que vous trouverez dans le fichier `deplacements.csv` vous donne les déplacements autorisés à partir de chaque case, sous la forme de 4 entiers: `x_origin, y_origin, x_destination,y_destination`. On commence par placer un objet sur la case (2,2). A chaque tour, l'objet est déplacé en choisissant un des déplacements autorisés (distribution de probabilité uniforme parmi les déplacements autorisés).

   (a) Est-il possible de rejoindre la case (0,9) en *exactement* deux étapes? en exactement 4? en au plus 4 étapes?

   Vous utiliserez une requête récursive calculant les cases accessibles et pouvant se généraliser à un nombre arbitraire d'étapes; pas une requête joignant 2 ou 4 fois `D`*. Pour s'y retrouver, il est utile de trier l'affichage.*

   (b) On souhaite connaitre la probabilité d'être arrivé à chaque case, au bout de (exactement) 4 étapes

   (c) Pour vérifier votre résultat vous pourrez éventuellement exporter la requête précédente en csv, et utiliser le script python `plotting-probas-heatmap.py` pour produire la Figure 2 (encore une fois, utiliser `ipython` car ce script utilise la librairie `pandas`).

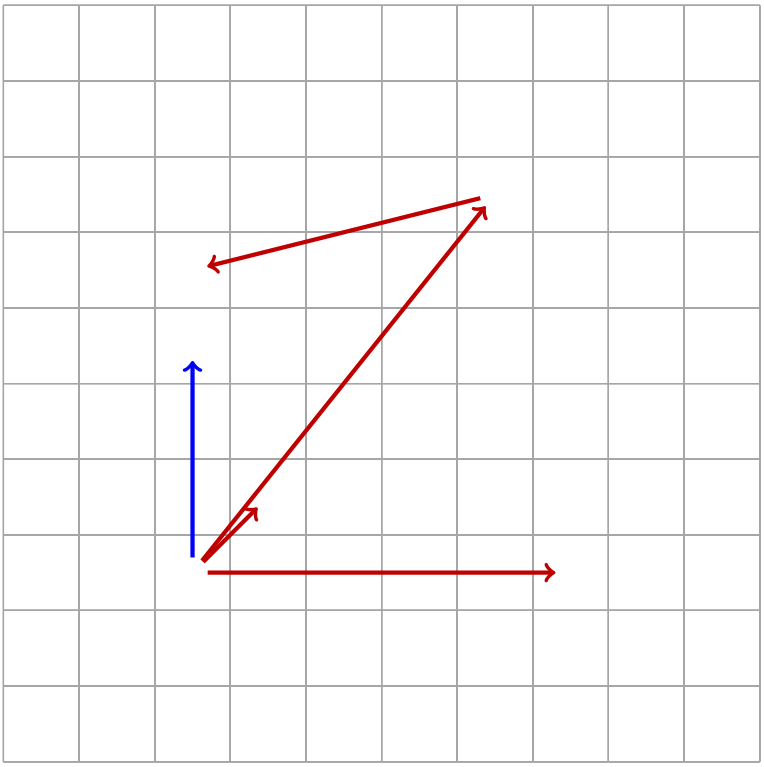

Figure 1: Une liste de déplacements possibles: (2,2,2,5)...

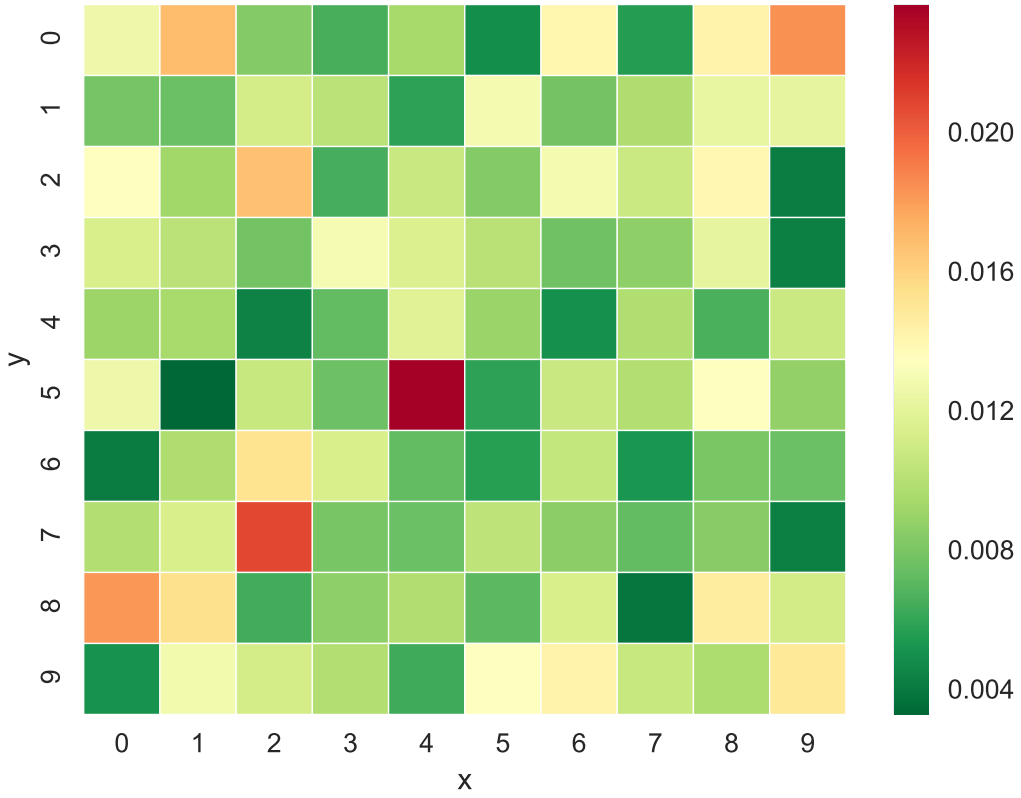

Figure 2: Probabilités au bout de 4 étapes

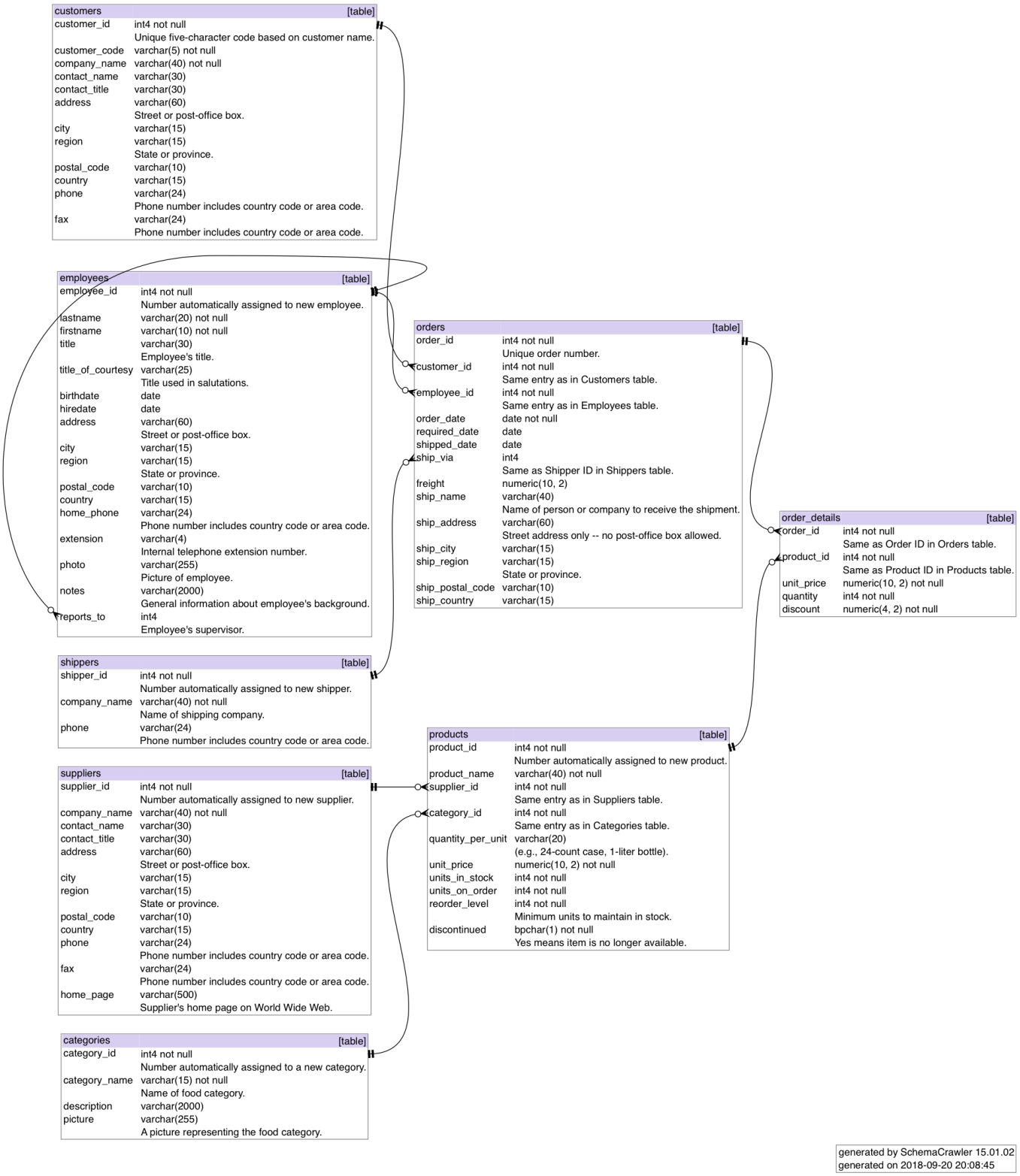

Figure 3: Schéma (approximatif) des relations de la base de donnée Northwind