**Laboratoire 8**

Utiliser le fichier BD\_sales.bak pour restaurer la base de données.

À l’aide de cette base de données, répondez aux questions suivantes:

1. Afficher le nom du produit le plus vendu avec deux méthodes différentes et comparer les deux méthodes en termes de performance en utilisant **SHOWPLAN\_ALL, STATISTICS PROFIL, etc.**
   1. TOP

-- La commande "SET STATISTICS PROFILE ON" permet d'afficher des informations sur les performances et pour aider à optimiser la requête.

-- La requête SQL permet de sélectionner le nom du produit ayant la quantité totale la plus élevée vendue dans les commandes.

-- On utilise les tables de production pour les produits et les stocks, ainsi que la table de ventes pour les commandes et les produits commandés.

SET STATISTICS PROFILE ON;

SELECT TOP 1 p.product\_name

FROM production.products p

JOIN production.stocks s ON p.product\_id = s.product\_id

JOIN sales.order\_items oi ON oi.product\_id = p.product\_id

GROUP BY p.product\_name

ORDER BY SUM(oi.quantity) DESC;

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

-- L'instruction "SET SHOWPLAN\_ALL ON" permet d'afficher le plan d'exécution de la requête.

-- La requête sélectionne le nom du produit le plus vendu en combinant les données des tables "production.products", "production.stocks" et "sales.order\_items".

-- On utilise une jointure pour lier les données entre les différentes tables, effectue un regroupement par nom de produit, puis trie les résultats en fonction

-- de la somme des quantités commandées pour chaque produit, pour ne retourner que le produit le plus vendu grâce à l'instruction "TOP 1".

SET SHOWPLAN\_ALL ON;

GO

SELECT TOP 1 p.product\_name

FROM production.products p

JOIN production.stocks s ON p.product\_id = s.product\_id

JOIN sales.order\_items oi ON oi.product\_id = p.product\_id

GROUP BY p.product\_name

ORDER BY SUM(oi.quantity) DESC;

Graphical user interface

Description automatically generated with low confidence

Table

Description automatically generated

* 1. MAX

-- L'instruction "SET SHOWPLAN\_ALL ON" permet d'afficher le plan d'exécution de la requête.

-- La requête sélectionne le nom du produit qui a réalisé le plus grand nombre de ventes en utilisant une sous-requête dans la clause HAVING.

-- La sous-requête calcule la somme des quantités vendues pour chaque produit, puis la requête principale sélectionne le produit ayant la plus grande somme.

SET SHOWPLAN\_ALL ON;

SELECT p.product\_name

FROM production.products p

JOIN production.stocks s ON p.product\_id = s.product\_id

JOIN sales.order\_items oi ON oi.product\_id = p.product\_id

GROUP BY p.product\_name

HAVING SUM(oi.quantity) = (

SELECT MAX(sales)

FROM (

SELECT SUM(oi.quantity) AS sales

FROM production.products p

JOIN production.stocks s ON p.product\_id = s.product\_id

JOIN sales.order\_items oi ON oi.product\_id = p.product\_id

GROUP BY p.product\_name

) AS sales\_by\_product

);

Graphical user interface

Description automatically generated

Table

Description automatically generated

-- b. MAX -SET STATISTICS PROFILE

-- La commande "SET STATISTICS PROFILE ON" permet d'afficher des informations sur les performances et pour aider à optimiser la requête.

-- La requête sélectionne le nom du produit ayant réalisé le plus grand nombre de ventes

-- en récupérant les données des tables "production.products", "production.stocks" et "sales.order\_items".

-- ON utilise une sous-requête pour calculer le nombre total de ventes pour chaque produit

-- et on récupère ensuite le nom du produit ayant réalisé le maximum de ventes.

SET STATISTICS PROFILE ON;

SELECT p.product\_name

FROM production.products p

JOIN production.stocks s ON p.product\_id = s.product\_id

JOIN sales.order\_items oi ON oi.product\_id = p.product\_id

GROUP BY p.product\_name

HAVING SUM(oi.quantity) = (

SELECT MAX(sales)

FROM (

SELECT SUM(oi.quantity) AS sales

FROM production.products p

JOIN production.stocks s ON p.product\_id = s.product\_id

JOIN sales.order\_items oi ON oi.product\_id = p.product\_id

GROUP BY p.product\_name

) AS sales\_by\_product

);

Graphical user interface, text, application

Description automatically generated

Table

Description automatically generated

1. En utilisant le Case, afficher les colonnes suivantes (category\_id, prix moyen, classement)

Le texte de la colonne classement généré en fonction de la valeur de **prix\_moyen**

|  |  |  |
| --- | --- | --- |
| category\_id | prix\_moyen (prix moyen des produits) | classement |
| 1 | prix\_moyen < 30 | Moins cher |
| 2 | 30 <= prix\_moyen < 50 | Cher |
| 3 | prix\_moyen >= 50 | Très cher |
| … |  | … |
| 7 |  | … |

-- Ceci calculera le prix moyen (AVG(list\_price)) pour chaque catégorie

-- (GROUP BY category\_id)

-- On utilise une instruction CASE

-- pour générer la colonne "classement" en fonction de la valeur du prix moyen.

-- Le résultat obtenu comprendra les colonnes category\_id, prix moyen et classement.

SELECT category\_id, AVG(list\_price) AS 'prix moyen',

CASE

WHEN AVG(list\_price) < 30 THEN 'Moins cher'

WHEN AVG(list\_price) >= 30 AND AVG(list\_price) < 50 THEN 'Cher'

ELSE 'Très cher'

END AS 'classement'

FROM production.products

GROUP BY category\_id;

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated