- 15.07.2021 DawSQL Sesion

2----- CROSS JOIN------- Soru1: Hangi markada hangi kategoride kaçar ürün olduğu bilgisine ihtiyaç duyuluyor  
-- Ürün sayısı hesaplamadan sadece marka \* kategori ihtimallerinin hepsini içeren bir tablo oluşturun  
-- Çıkan sonucu daha kolay yorumlamak için brand\_id ve category\_id alanlarına göre sıralayın.

SELECT \*  
FROM production.brands  
CROSS JOIN production.categories  
ORDER BY brand\_id

----- SELF JOIN-------- Soru2: Write a query that returns the staff with their managers.  
-- Expected columns: staff first name, staff last name, manager name

SELECT \*  
FROM sales.staffs AS A  
JOIN sales.staffs AS B  
ON A.manager\_id = B.staff\_id

SELECT A.first\_name AS Staff\_Name, A.last\_name AS Staff\_Last, B.first\_name AS Manager  
FROM sales.staffs A, sales.staffs B  
WHERE  A.manager\_id = B.staff\_id

---- GROUPBY / HAVING ----

--GROUPING OPERATION SORU1--  
--Write a query that checks if any product id is repeated in more than one row in the products table.

SELECT A.product\_name, COUNT(A.product\_name)  
FROM production.products AS A  
GROUP BY A.product\_name  
HAVING COUNT(A.product\_name) >1;

--HAVING’DE kullandığın sütun Aggregate te kullandığın sütun ismiyle aynı olmalı.-- hocanın çözümü:  
-- önce products ları görelim.

SELECT TOP 20\*  
FROM production.productsSELECT product\_id, COUNT(\*) AS CNT\_PRODUCT  
FROM production.products  
GROUP BY  
product\_id  -- bütün product\_id lerin product tablosunda birer kere geçtiğini gördüm.SELECT product\_id, COUNT(\*) AS CNT\_PRODUCT  
FROM production.products  
GROUP BY  
product\_id  
HAVING  
COUNT(\*) > 1

--HAVING’DE kullandığın sütun Aggregate te kullandığın sütun ismiyle aynı olmalı.-- product\_name e göre yapalım  
SELECT product\_name, COUNT(\*) AS CNT\_PRODUCT

-- count(\*) tüm rowları say demek. count(product\_id) de aynı işi görür.  
FROM production.products  
GROUP BY  
product\_name  
HAVING  
COUNT (\*) > 1  
-- aşağıdaki gibi de kullanabiliriz.

SELECT product\_name, COUNT(product\_id) AS CNT\_PRODUCT

-- count(\*) tüm rowları say demek. count(product\_id) de aynı işi görür.  
FROM production.products  
GROUP BY  
product\_name  
HAVING  
COUNT (product\_id) > 1

SELECT production\_id, production\_name, COUNT (\*) CNT\_PRODUCT  
FROM production.products  
GROUP BY  
product\_name  
HAVING  
COUNT (\*) > 1  
-- select te yazdığın sütunlar group by da olması gerekiyor. production\_id group by da olmadığı için hata verdi.

SELECT production\_id, production\_name, COUNT (\*) CNT\_PRODUCT  
FROM production.products  
GROUP BY  
product\_name, product\_id  
HAVING  
COUNT (\*) > 1

SELECT product\_id, COUNT (\*) AS CNT\_PRODUCT  
FROM production.products  
GROUP BY  
product\_id  
HAVING  
COUNT (\*) > 1

--GROUPING OPERATION SORU 2--  
-- Write a query that returns category ids with a maximum list price above 4000 or a minimum list price below 500  
SELECT category\_id, MIN(list\_price) AS min\_price, MAX(list\_price) AS max\_price -- grupladığımız şey category\_id olduğu için SELECT'te onu getiriyoruz  
FROM production.products  
-- ana tablo içinde herhangi bir kısıtlamam var mı yani where işlemi var mı? yok. devam ediyorum  
GROUP BY  
category\_id  
HAVING  
MIN(list\_price) < 500 OR MAX(list\_price) > 4000--GROUPING OPERATION SORU 3--  
-- Find the average product prices of the brands.  
-- As a result of the query, the average prices should be displayed in descending order.SELECT A.brand\_name, AVG(B.list\_price) AS AVG\_PRICE  
FROM production.brands A, production.products B  
-- buradaki virgül INNER JOIN ile aynı işi yapıyor! virgülle beraber WHERE kullanıyoruz.  
WHERE A.brand\_id = B.brand\_id  
GROUP BY  
A.brand\_name  
ORDER BY  
AVG\_PRICE DESC-- (virgül + WHERE yerine--> INNER JOIN ile çözüm)  
SELECT A.brand\_name, AVG(B.list\_price) AS AVG\_PRICE  
FROM production.brands AS A  
INNER JOIN production.products AS B  
ON A.brand\_id = B.brand\_id  
GROUP BY  
A.brand\_name  
ORDER BY  
AVG\_PRICE DESC  
-- ORDER BY 2 DESC olarak da yazabilirdik. Burada 2 --> SELECT'teki ikinci belirtilen veriyi temsil ediyor.--GROUPING OPERATION SORU 4--  
-- Write a query that returns BRANDS with an average product price more than 1000  
SELECT A.brand\_name, AVG(B.list\_price) AS AVG\_PRICE  
FROM production.brands A, production.products B  
WHERE A.brand\_id = B.brand\_id  
GROUP BY  
A.brand\_name  
HAVING AVG(B.list\_price) > 1000  
ORDER BY  
2 DESC--GROUPING OPERATION SORU 5--  
--  Write a query that returns the net price paid by the customer for each order. (Don't neglect discounts and quantities)  
SELECT \*, (quantity \* list\_price \* (1-discount)) as net\_price --list\_price-list\_price\*discount olarak da yazılabilir  
FROM sales.order\_items  
-- bu query ile önce her bire order\_id için list\_price ile indirim uygulanmış net\_price ları görüyoruz.-- order'larda birden fazla ürün sipariş verilmiş olduğunu görmüştüm.  
-- O yüzden ürünleri order\_id olarak gruplandırıp her grup için toplama (SUM) yaparak  
-- her order için toplam net\_price'ı görmüş olacağım  
SELECT order\_id, SUM(quantity \* list\_price \* (1-discount)) as net\_price  
FROM sales.order\_items  
GROUP BY  
order\_id--- SUMMARY TABLE---SELECT \*  
INTO NEW\_TABLE -- INTO SATIRINDAKİ TABLO İSEMİ İLE YENİ BİR TABLO OLUŞTURUYORUZ.  
FROM SOURCE\_TABLE  -- FROM'DAN SONRASI KAYNAK TABLOMUZ  
WHERE ...SELECT C.brand\_name as Brand, D.category\_name as Category, B.model\_year as Model\_Year,  
ROUND (SUM (A.quantity \* A.list\_price \* (1 - A.discount)), 0) total\_sales\_price  
INTO sales.sales\_summary  
FROM sales.order\_items A, production.products B, production.brands C, production.categories D  
WHERE A.product\_id = B.product\_id  
AND B.brand\_id = C.brand\_id  
AND B.category\_id = D.category\_id  
GROUP BY  
C.brand\_name, D.category\_name, B.model\_yearSELECT \*  
FROM sales.sales\_summary  
ORDER BY 1,2,3-- Bundan sonra bu tabloyu kullanacağım!--- GROUPING SETS------ 1. Toplam sales miktarını hesaplayınız.  
SELECT SUM(total\_sales\_price)  
FROM sales.sales\_summary-- 2. Markaların toplam sales miktarını hesaplayınız.  
SELECT Brand, SUM(total\_sales\_price)  
FROM sales.sales\_summary  
GROUP BY  
Brand-- 3. Kategori bazında toplam sales miktarını hesaplayınız  
SELECT Category, SUM(total\_sales\_price)  
FROM sales.sales\_summary  
GROUP BY  
Category-- 4. Marka ve kategori kırılımlarındaki toplam sales miktarlarını hesaplayınız  
SELECT Brand, Category, SUM(total\_sales\_price)  
FROM sales.sales\_summary  
GROUP BY  
Brand, Category-- BU İŞLERMLERİ GROUPING SETS YÖNTEMİ İLE YAPALIM :---  
SELECT brand, category, SUM(total\_sales\_price)  
FROM sales.sales\_summary  
GROUP BY  
GROUPING SETS(  
(Brand),  
(category),  
(brand, category),  
()      -- boş parantez ile  
)  
ORDER BY  
1,2----- ROLLUP GRUPLAMA-----  
SELECT  
d1,  
d2,  
d3,  
aggregate\_function  
FROM  
table\_name  
GROUP BY  
ROLLUP (d1,d2,d3); -- önce tüm sütuınları alıyor sonra sağdan başlayarak teker teker silerek her defasında yeniden bir gruplama yapıyor;  
-- önce üç sütuna göre grupluyor, sonra sondakini atıp ilk 2 sütuna göre grupluyor  
-- sonra sondakini yine atıp ilk sütuna göre grupluyor  
-- sonra hiç gruplamıyor.--SELECT brand, category, SUM(total\_sales\_price)  
FROM sales.sales\_summary  
GROUP BY  
ROLLUP (Brand, Category)  
ORDER BY  
1,2  
;--- CUBE GRUPLAMA------- önce önce üç sütunu birden grupluyor  
-- sonra kalanları 2'şer 2'şer 3 defa gruplama yapıyor  
-- sonra kalanları teker teker grupluyor  
-- en son gruplamıyor.SELECT brand, category, SUM(total\_sales\_price)  
FROM sales.sales\_summary  
GROUP BY  
CUBE (Brand, Category)  
ORDER BY  
1,2  
;