**SQL CRASH COURSE PART 2**

**Query the database joining tables**

**-- ----------------------------------------------------**

**-- Query 1 ​ Our first query should return the "sku", "product\_quantity", "date" and "unit\_price" from the line\_item table together with the "name" and the "price" of each product from the "products" table. We want only products present in both tables.**

SELECT

itm.sku,

itm.product\_quantity,

itm.date,

itm.unit\_price AS IPrice,

prd.name\_en,

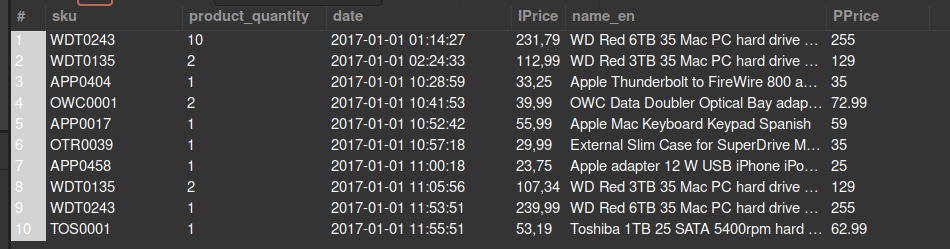
prd.price AS PPrice

FROM

line\_item itm

INNER JOIN

products prd ON itm.sku = prd.sku;



**-- ----------------------------------------------------**

**-- Query 2 You might notice that the ​ unit\_price ​ from the ​ line\_item ​ table and the price from the ​ product ​ table is not the same. Let's investigate that! Extend your previous query by adding a column with the difference in price. Name that column ​ price\_difference ​ .**

SELECT

itm.sku,

itm.product\_quantity,

itm.date,

itm.unit\_price AS IPrice,

prd.name\_en,

prd.price AS PPrice,

ROUND(prd.price - itm.unit\_price, 1) AS price\_diff

FROM

line\_item itm

INNER JOIN

products prd ON itm.sku = prd.sku;



**-- ----------------------------------------------------**

**-- Query 3 Build a query that outputs the price difference that you just calculated, grouping products by category. Round the result.**

SELECT

inn.manual\_categories,

ROUND(AVG(inn.price\_dif),2) as AVG\_Price\_diff

FROM

(SELECT

line\_item.sku,

line\_item.product\_quantity,

line\_item.date,

line\_item.unit\_price,

products.name\_en,

products.price,

ROUND(ABS(line\_item.unit\_price - products.price), 1) AS price\_dif,

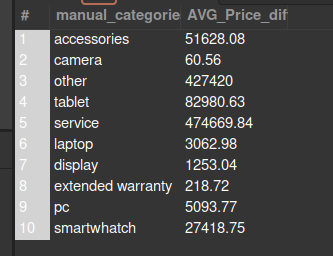
products.manual\_categories

FROM line\_item

INNER JOIN products

ON line\_item.sku = products.sku) as inn

GROUP BY inn.manual\_categories;



**-- ----------------------------------------------------**

**-- Query 4. ​ Create the same query as before (calculating the price difference between the ​line\_item and the ​products tables, but now grouping by brands instead of categories.**

SELECT

inn.brand,

ROUND(AVG(inn.price\_dif),1) as AVG\_price\_dif

FROM

(SELECT

itm.sku,

itm.product\_quantity,

itm.date,

prd.name\_en,

itm.unit\_price AS IPrice,

prd.brand,

prd.price AS PPrice,

ROUND(prd.price - itm.unit\_price, 1) AS price\_dif

FROM

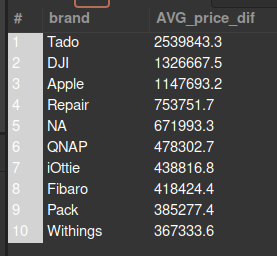
line\_item itm

INNER JOIN

products prd ON itm.sku = prd.sku ) AS inn

GROUP BY inn.brand

ORDER BY AVG\_price\_dif DESC;



**-- ----------------------------------------------------**

**-- Query 5. ​ Let's focus on the brands with a big price difference: run the same query as before, but now limiting the results to only brands with an avg\_price\_dif of more than 50000. Order the results by ​ avg\_price\_dif ​ (bigger to smaller).**

SELECT

inn.brand,

ROUND(AVG(inn.price\_dif),1) as AVG\_price\_dif

FROM

(SELECT

itm.sku,

itm.product\_quantity,

itm.date,

prd.name\_en,

itm.unit\_price AS IPrice,

prd.brand,

prd.price AS PPrice,

ROUND(prd.price - itm.unit\_price, 1) AS price\_dif

FROM

line\_item itm

INNER JOIN

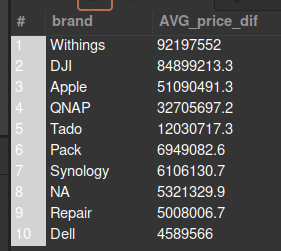
products prd ON itm.sku = prd.sku ) AS inn

WHERE

inn.price\_dif > 50000

GROUP BY inn.brand

ORDER BY AVG\_price\_dif DESC;



**-- ----------------------------------------------------**

**-- Query 6. ​ Query 6. We want to know the sku, product\_quantity and date of all the orders, ordered by SKU. If it takes too long and/or the connection gets lost try first selecting only the first 50 results and then ordering by sku.**

SELECT

ord.created\_date,

ord.state,

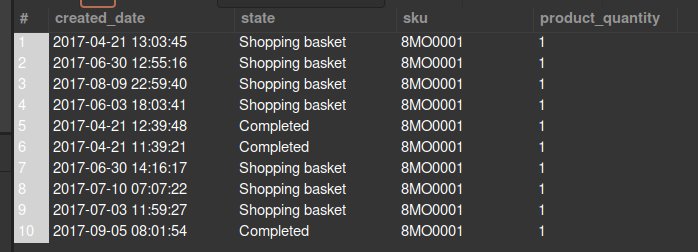
itm.sku,

itm.product\_quantity

FROM line\_item AS itm

INNER JOIN orders AS ord ON itm.id\_order = ord.id\_order

ORDER BY itm.sku;



**-- ----------------------------------------------------**

**-- Query 7. ​ Add to the previous information about "​brand" ​and "​manual\_categories" fields from the products table.**

SELECT

ord.created\_date, ord.state,

itm.sku, itm.product\_quantity,

prd.brand, prd.manual\_categories

FROM

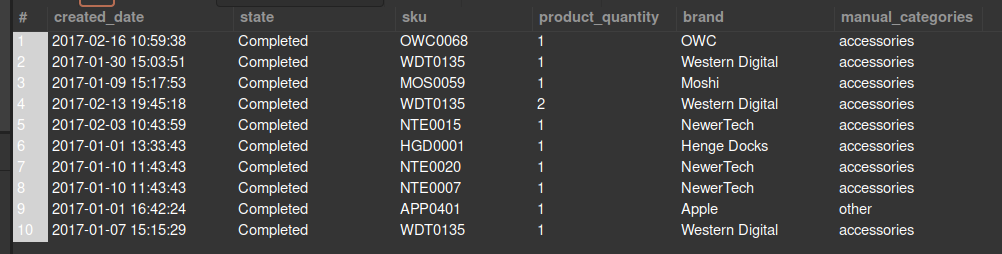
line\_item AS itm

INNER JOIN

orders AS ord ON itm.id\_order = ord.id\_order

INNER JOIN

products AS prd ON itm.sku = prd.sku;



**-- -----------------------------------------------------------------------------**

**-- Query 8.​ We want to know which brand and which categories are most frequent in Cancelled orders.**

**-- Let's keep working on the same query: now we want to keep only Cancelled orders. Modify this query to group the results from the previous query, first by category and then by brand, adding in both cases a count and ordering by descending count. If it takes too long and/or the connection gets lost try putting a LIMIT command.**

SELECT

inn.brand, COUNT(\*) AS 'Cancelled Orders by Brand'

FROM (

SELECT

-- ord.created\_date,

ord.state,

-- itm.sku,

-- itm.product\_quantity,

prd.brand -- , prd.manual\_categories

FROM

line\_item AS itm

INNER JOIN

orders AS ord ON itm.id\_order = ord.id\_order

INNER JOIN

products AS prd ON itm.sku = prd.sku

LIMIT 1000

) as inn

WHERE inn.state = 'Cancelled'

GROUP BY inn.brand ;

