DBMS Assignment 1

Alli Khadga Jyoth 19024

February 2022

1 Schema:

products (p_id : int , cat_id : int, p_name : string, p_price : real, p_rating : real, seller_id : int)
categories (cat_id : int, cat_name : string)
seller_rating (seller_id : int, seller_rating : real)

p_id	cat_id	p_name	p_price	p_rating	seller_id
1	1	jeans	30	3.1	1
2	1	jeans	30	4	2
3	1	jeans	40	4.5	1
4	1	hoodie	30	4	1
5	1	hoodie	35	4.5	1
6	1	shirt	25	4	2
7	1	shirt	30	4.5	3
8	2	dal	10	3.8	4
9	2	dal	20	4.5	5
10	2	cookie	5	4	4
11	2	cookie	8	4.6	6
12	2	cookie	5	4.6	5
13	3	phone	400	4.6	7
14	3	phone	600	4.7	8
15	3	headphone	100	3.8	8
16	3	headphone	150	4.2	9

Table 1: Products table

1.1 Question:

select all the different products from the products relation having the cheapest price of the category and if two products have the same price select the one having the higher rating.

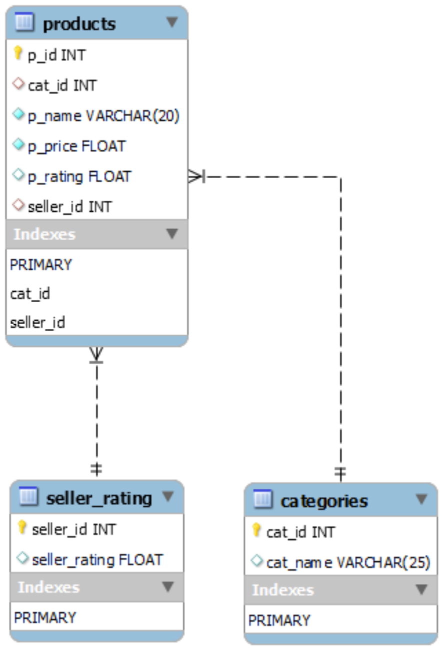

Figure 1: ER model

1.2 Solution:

$$products - (products \bowtie \rho_{p\_id}(\pi_{m.p\_id}(\rho_p products \bowtie_{(p.p\_name=m.p\_name)\wedge((p.p\_price<m.p\_price)\vee((p.p\_price=m.p\_price)\wedge p.p\_rating>m.p\_rating))} \rho_m products)))$$

p_id	cat_id	p_name	p_price	p_rating	seller_id
2	1	jeans	30	4	2
4	1	hoodie	30	4	1
6	1	shirt	25	4	2
8	2	dal	10	3.8	4
12	2	cookie	5	4.6	5
13	3	phone	400	4.6	7
15	3	headphone	100	3.8	8

Table 2: Solution Table

1.2.1 MySQL Command:

```
select * from products where p_id not in (select m.p_id from products p
        join products m on (p.p_name = m.p_name) and
            (p.p_price < m.p_price or
                (p.p_price = m.p_price and p.p_rating > m.p_rating)))
```

1.2.2 Explanation:

We need to select different products from each category so that the product selected has the least price and if two or more products have the same price, select the one with the higher rating. So we first do a self join with products (rename as p and m) on p_name. So this gives all the combinations of the same product names. From here we select the products with p.p_price < m.p_price or (p.p_price = m.p_price and p.p_rating > m.p_rating), giving a set of products that satisfy our conditions. This gives m.p_id, which are exactly the ones which we do not want. So, we do another join on products by renaming m.p_id as p_id so we can get all the rows we do not want and then subtract from the relation products.

p.p_id	p.p_name	p.p_price	p.p_rating	m.p_id	m.p_name	m.p_price	m.p_rating
2	jeans	30	4	1	jeans	30	3.1
2	jeans	30	4	3	jeans	40	4.5
1	jeans	30	3.1	3	jeans	40	4.5
4	hoodie	30	4	5	hoodie	35	4.5
6	shirt	25	4	7	shirt	30	4.5
8	dal	10	3.8	9	dal	20	4.5
12	cookie	5	4.6	10	cookie	5	4
12	cookie	5	4.6	11	cookie	8	4.6
10	cookie	5	4	11	cookie	8	4.6
13	phone	400	4.6	14	phone	600	4.7
15	headphone	100	3.8	16	headphone	150	4.2

Table 3: Table obtained from doing join of products p, m with constraints, we can see m.p_id contains exactly all the products which we do not want.