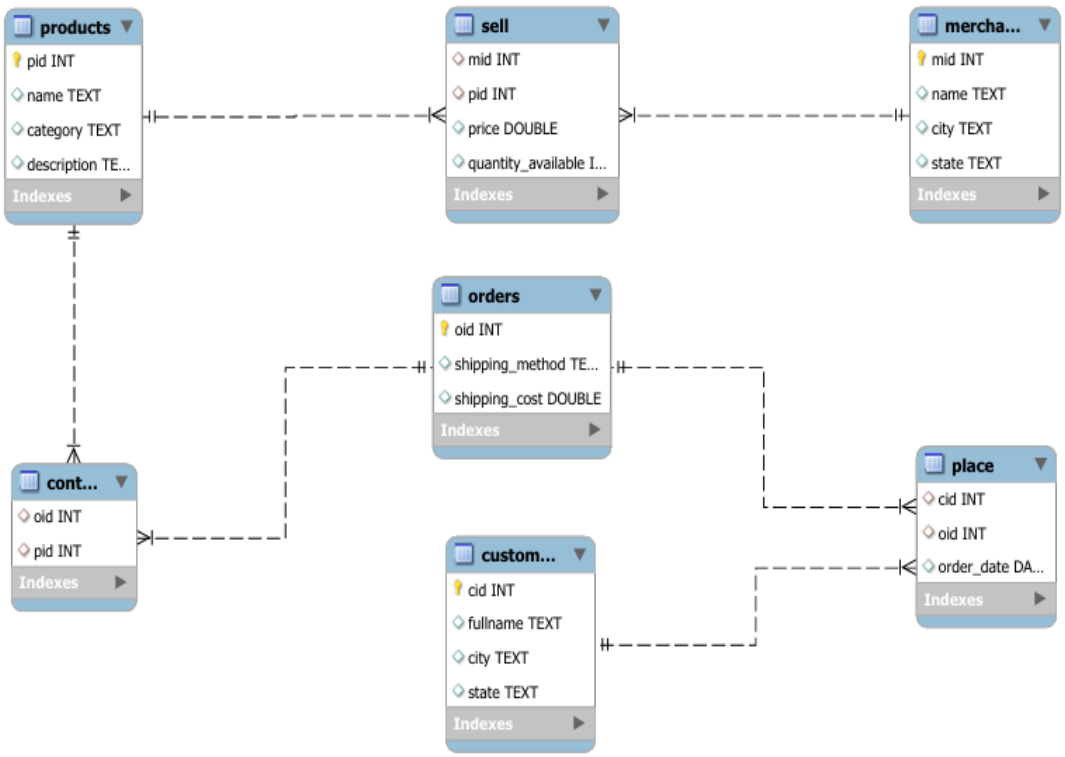
DB Assignment 3
Katrina Cwiertniewicz
10/11/2024

Primary key and check constraints syntax found from GeeksforGeeks.
Round():MySQL :: MySQL 8.4 Reference Manual :: 14.6.2 Mathematical Functions
Rank()/over/Parition by found here:MySQL :: MySQL 8.4 Reference Manual :: 5.6.4 The Rows Holding the Group-wise Maximum of a Certain Column

ERD Diagram



1. List names and sellers of products that are no longer available (quantity=0)

The query selects merchant names and product names. It joins merchants to products through the foreign keys in sell. It finds when the quantity available is 0 and groups by merchant names and product names.

```
-- 1. List names and sellers of products that are no longer available (quantity=0)
select merchants.name as "Sellers of Products", products.name as "Product Names"
from merchants inner join sell on merchants.mid = sell.mid
               inner join products on sell.pid = products.pid
where sell.quantity_available = 0
group by merchants.name, products.name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Sellers of Products	Product Names
Acer	Router
Acer	Network Card
Apple	Printer
Apple	Router
HP	Router
HP	Super Drive
HP	Laptop
Dell	Router
Lenovo	Ethernet Adapter

2. List names and descriptions of products that are not sold.

The query selects product names and product descriptions. It determines which products do not exist by finding where there is no match between the pid of products and sell. It is grouped by product name and product description.

```
-- 2. List names and descriptions of products that are not sold.
select products.name as "Products Names", products.description as "Descriptions of Products"
from products
where not exists (select * from sell where products.pid = sell.pid)
group by products.name, products.description;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Products Names	Descriptions of Products
Super Drive	External CD/DVD/RW
Super Drive	UInternal CD/DVD/RW

3. How many customers bought SATA drives but not any routers?

The query selects count all and creates a subquery to select distinct customers' full names from customers. It joins customers to order through the foreign keys in place and order to product through the foreign keys in contain. It finds all customers who bought a product that included the description ‘SATA’ and then creates an additional subquery to find those who did not buy any routers using not exists to select distinct customer full names who bought a product that included the description ‘Router’.

```
-- 3. How many customers bought SATA drives but not any routers?
select count(*) as "Number of Distinct Customers that bought SATA drives but not routers"
from (select distinct customers.fullname
from customers inner join place on customers.cid = place.cid
               inner join orders on place.oid = orders.oid
               inner join contain on orders.oid = contain.oid
               inner join products on contain.pid = products.pid
where products.description like '%SATA%'
and not exists(select distinct customers.fullname
from customers inner join place on customers.cid = place.cid
               inner join orders on place.oid = orders.oid
               inner join contain on orders.oid = contain.oid
               inner join products on contain.pid = products.pid
where products.description = 'Router')) as table1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Number of Distinct Customers that bought SATA drives but not routers
20

4. HP has a 20% sale on all its Networking products.

The query selects the merchant name, product name, and sell price. It joins merchants to products through the foreign keys in sell. It filters names by ‘HP’ and categories by ‘Networking’. This displays the original price of HP products before the discounts.

```
-- 4. HP has a 20% sale on all its Networking products.
select merchants.name as "Company Name", products.name "Product Name", sell.price as "Original Price of Networking Products"
from merchants join sell on merchants.mid = sell.mid
               join products on sell.pid = products.pid
where merchants.name = 'HP' and products.category = 'Networking';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Company Name	Product Name	Original Price of Networking Products
HP	Router	1034.46
HP	Network Card	1154.68
HP	Network Card	345.01
HP	Network Card	262.2
HP	Ethernet Adapter	1260.45
HP	Router	205.56
HP	Router	1474.87
HP	Router	552.02
HP	Router	100.95
HP	Network Card	1179.01

This query updates sell and joins merchants to sell. It sets the sell price to 20% discount when the merchant's name is HP and the category is Networking.

```
update sell
join merchants on merchants.mid = sell.mid
set sell.price = sell.price - (.20 * sell.price)
where merchants.name = 'HP';

select merchants.name as "Company Name", products.name "Product Name", sell.price as "20% off HP Products"
from merchants join sell on merchants.mid = sell.mid
               join products on sell.pid = products.pid
where merchants.name = 'HP' and products.category = 'Networking';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Company Name	Product Name	20% off HP Products
HP	Router	827.568
HP	Network Card	923.744
HP	Network Card	276.008
HP	Network Card	209.76
HP	Ethernet Adapter	1008.36
HP	Router	164.448
HP	Router	1179.896
HP	Router	441.616
HP	Router	80.76
HP	Network Card	943.208

5. What did Uriel Whitney order from Acer? (make sure to at least retrieve product names and prices).

This query selects customer's full name, product name, and sell price. It joins customers and order through the foreign keys in place, orders and products through the foreign keys in contain and products and merchants through the foreign keys in sell. It uses where to filter the products and their price of what Uriel Whitney ordered from Acer.

```
-- 5. What did Uriel Whitney order from Acer? (make sure to at least retrieve product names and prices).
select customers.fullname, products.name, sell.price
from customers join place on customers.cid = place.cid
               join orders on place.oid = orders.oid
               join contain on orders.oid = contain.oid
               join products on contain.pid = products.pid
               join sell on products.pid = sell.pid
               join merchants on sell.mid = merchants.mid
where customers.fullname = 'Uriel Whitney' and merchants.name = 'Acer';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

fullname	name	price
Uriel Whitney	Monitor	1435.38
Uriel Whitney	Router	521.07
Uriel Whitney	Router	1256.57
Uriel Whitney	Monitor	1103.47
Uriel Whitney	Super Drive	356.13
Uriel Whitney	Printer	1345.37
Uriel Whitney	Super Drive	671.75
Uriel Whitney	Super Drive	1135.3
Uriel Whitney	Super Drive	356.13
Uriel Whitney	Super Drive	1015.95
Uriel Whitney	Network C...	405.4
Uriel Whitney	Hard Drive	836.99
Uriel Whitney	Super Drive	1124.26
Uriel Whitney	Network C...	609.2
Uriel Whitney	Printer	1345.37
Uriel Whitney	Network C...	405.4
Uriel Whitney	Super Drive	671.75
Uriel Whitney	Super Drive	1135.3

6. List the annual total sales for each company (sort the results along the company and the year attributes).

The query selects merchant names, sum of total sales (sell.price • quantity_available) rounded to two decimal places and order date year. Merchants is joined with products through the foreign keys in sell and contain is joined with place through the foreign keys in orders. The results are grouped by merchant names and order date year and ordered by order date year.

```
-- 6. List the annual total sales for each company (sort the results along the company and the year attributes).
select merchants.name as "Company", round(sum(sell.price * quantity_available),2) as "Total Sales", year(place.order_date) as "Year"
from merchants join sell on merchants.mid = sell.mid
                join products on sell.pid = products.pid
                join contain on products.pid = contain.pid
                join orders on contain.oid = orders.oid
                join place on orders.oid = place.cid
group by merchants.name, year(place.order_date)
order by year(place.order_date);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Company	Total Sales	Year
Acer	659028.95	2011
Apple	846538.38	2011
Dell	1197141.19	2011
Lenovo	1236193.67	2011
HP	489187.51	2011
Acer	219823.18	2016
Apple	253761.59	2016
Dell	353022.7	2016
Lenovo	369650.76	2016
HP	168809.65	2016
Acer	799267.32	2017
Apple	905690.84	2017
Dell	1222680.45	2017
Lenovo	1293855.3	2017
HP	583297.16	2017
Acer	881113.32	2018
Apple	1151043.65	2018
Dell	1845081.22	2018
Lenovo	1699529.84	2018
HP	865136.74	2018
Acer	969181.23	2019
Apple	1057628.06	2019
Dell	1530274.41	2019
Lenovo	1412380.43	2019
HP	618308.11	2019
Acer	788998.68	2020
Apple	797955.09	2020
Dell	1235477.59	2020
Lenovo	1189000.72	2020
HP	496491.17	2020

7. Which company had the highest annual revenue and in what year?

The query selects merchant names, sum of sell price rounded to two decimal places and order date year. Merchants is joined with products through the foreign keys in sell and contain is joined with place through the foreign keys in orders. The results are grouped by merchant name, sell price and order date year and ordered by sell price sum descending. It only displays the top one using limit 1, displaying the company with the highest annual revenue.

```
-- 7. Which company had the highest annual revenue and in what year?
select merchants.name as "Company", round(sum(sell.price),2) as "Highest Annual Revenue", year(place.order_date) as "Year"
from merchants join sell on merchants.mid = sell.mid
                join products on sell.pid = products.pid
                join contain on products.pid = contain.pid
                join orders on contain.oid = orders.oid
                join place on orders.oid = place.cid
group by merchants.name, sell.price, year(place.order_date)
order by sum(sell.price) desc
limit 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Company	Highest Annual Revenue	Year
Acer	32670.82	2018

8. On average, what was the cheapest shipping method used ever?

The query selects shipping method and shipping cost average rounded to 2 decimal places from order. The results are grouped by cheapest shipping method ordering by descending order. It only displays the top one using limit 1, displaying the cheapest shipping method.

```
-- 8. On average, what was the cheapest shipping method used ever?
select orders.shipping_method as "Cheapest Shipping Method", round(avg(orders.shipping_cost),2) as "Cost"
from orders
group by shipping_method
order by shipping_method desc
limit 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Cheapest Shipping Method	Cost
USPS	7.46

9. What is the best sold ($) category for each company?

The query is aliased as c and selects merchant names, the average sell price rounded to 2 decimal places and the product category. It partitions to group each merchant name and is ordered by average sell price. Merchants is joined with products through the foreign keys in sell and contain is joined with place through the foreign keys in orders. The results are grouped by merchant name and product category. All are selected from c and best sold category for each company is displayed.

```
-- 9. What is the best sold ($) category for each company?
with c as (select merchants.name as "Company Name", round(avg(sell.price),2) as "Revenue",
products.category as "Best Sold Category", row_number() over (partition by merchants.name order by avg(sell.price) desc) as order_rank
from merchants join sell on merchants.mid = sell.mid
               join products on sell.pid = products.pid
               join contain on products.pid = contain.pid
               join orders on contain.oid = orders.oid
               join place on orders.oid = place.oid
group by merchants.name, products.category)
select *
from c
where order_rank < 2;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Company Name	Revenue	Best Sold Category	order_rank
Acer	899.76	Peripheral	1
Apple	874.66	Networking	1
Dell	1092.28	Computer	1
HP	771.43	Computer	1
Lenovo	1216.03	Computer	1

10. For each company find out which customers have spent the most and the least amounts.

The query selects all from a subquery aliased as c. It selects merchants name, customers full name and what customers spent (sum(sell.price)) rounded to 2 decimal places. It partitions to group each merchant name and is ordered by max sell price. Merchants is joined with products through the foreign keys in sell, contain is joined with place through the foreign keys in orders and orders is joined with customers through the foreign keys in place. The results are grouped by merchant name, customer full name and sell price and filters by order_rank 1 where each company's top result (max sell price) will display. This is unioned with subquery aliased as c2 with the only difference between the subqueries being the min (sell.price). This displays the customers that spent the most and the least with each company.

```
-- 10. For each company find out which customers have spent the most and the least amounts.
select * from(select distinct merchants.name, customers.fullname,
round(sum(sell.price),2) as "Customers that Spent the Most and Least Amount", row_number() over (partition by merchants.name order by max(sell.price) desc) as order_rank
from merchants join sell on merchants.mid = sell.mid
                join products on sell.pid = products.pid
                join contain on products.pid = contain.pid
                join orders on contain.oid = orders.oid
                join place on orders.oid = place.cid
                join customers on place.cid = customers.cid
group by merchants.name, customers.fullname, sell.price) c
where order_rank = 1

union

select * from(select distinct merchants.name, customers.fullname,
round(sum(sell.price),2) as "Customers that Spent the Most and Least Amount", row_number() over (partition by merchants.name order by min(sell.price)) as order_rank
from merchants join sell on merchants.mid = sell.mid
                join products on sell.pid = products.pid
                join contain on products.pid = contain.pid
                join orders on contain.oid = orders.oid
                join place on orders.oid = place.cid
                join customers on place.cid = customers.cid
group by merchants.name, customers.fullname, sell.price)c2
where order_rank = 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	fullname	Customers that Spent the Most and Least Amount	order_rank
Acer	Jerry Roberts	25836.84	1
Apple	Inez Long	21097.8	1
Dell	Demetrius Garcia	31916.28	1
HP	Jerry Roberts	21772.62	1
Lenovo	Dean Heath	42834.56	1
Acer	Nissim Rosa	938	1
Apple	Dean Heath	3357.3	1
Dell	Justin Mccray	2107.95	1
HP	Wynne Mckinney	1292.16	1
Lenovo	Uriel Whitney	1484.16	1