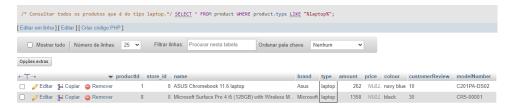
Repositório: https://www.github.com/Navomeister/bcd-umbrela/tree/main

## **Queries:**

1. Consultar o endereço, hora de início (start time) e hora final (end time) dos pontos de serviço da mesma cidade que o usuário cujo ID é 5.



2. Consultar todos os produtos que é do tipo laptop.



3. Consultar a quantidade total de produtos que foram colocados no carrinho (shopping cart), considerando a loja com storeid (sid) igual a 8.



4. Consultar os campos name, streetaddr e city (tabela adrress) de todos os pedidos que foram entregues em 17-02-2017.



## 5. Consulte os comentários do produto 123456789;



## **Tipos dos Campos:**

'addrld', 'cardld', 'buyerld', 'orderld', 'itemld', 'creditld', 'debitld', 'addrld', 'sellerld', 'storeld', 'productld', 'buyerld', 'idComm', 'serviceld' e suas respectivas chaves estrangeiras: int(10); são chaves primárias, portanto precisam sem exatas e permanentes. As chaves estrangeiras precisam ser idênticas às primárias, portanto são do mesmo tipo.

address => name, city, postalCode, province: varchar(50); são textos pequenos e variáveis, então não precisam de muito espaço.

streetAddr: varchar(250); é um texto que pode ser grande, precisa de mais espaço disponível.

contactPhoneNumber: char(11); é um número de tamanho fixo, mas não são feitas contas com ele, podendo ser deixado como char.

bank\_card => cardNumber: char(19); também um campo com tamanho fixo que não será feito contas.

bank: varchar(50); pequeno texto de tamanho variável.

expiryDate: date; data que não precisa de horário.

contain => quantity: int(5); pequena quantidade de tamanho variável.

credit\_card => organization: varchar(50); texto pequeno.

deliver\_to => timeDelivered: datetime; data com horário, já que precisa de especificidade.

manage => setupTime: datetime; data com horário, também precisa ser específico

order => creationTime: datetime; apenas o horário deixaria muito aberto paymentStatus: varchar(50); texto com poucas palavras, não precisa de muito.

totalAmount: int(5); um número com poucos dígitos, dificilmente haverá uma compra grande os uficiente para usá-los.

order\_item => price: float(6,2); um valor decimal com 2 digitos após a vírgula, para preço em dinheiro.

creationTime: datetime; novamente data com hora, pelos mesmos motivos.

payment => paytime: datetime; mesmo motivo.

product => name: varchar(100); texto para o nome do produto, não sendo nem tão grande nem pequeno demais.

brand: varchar(50); marcas não costumam ser grandes.

type: varchar(50); o tipo do produto normalmente é usado em tags, então não são grandes.

amount: int(5); o estoque não deve ser maior que 99999 unidades.

price: float(6,2); preço decimal para uso em moeda.

colour: varchar(50); apenas o nome da cor.

customerReview: int(3); avaliação de 0 a 100.

save\_to\_shopping\_cart => addTime: datetime; data com especificidade da hora.

quantity: int(3); não precisa ser tão grande.

store => name: varchar(50); nomes de lojas não costumam ser tão grandes.

startTime: datetime; necessário pro input, mesmo não tendo horário.

customerGrade: int(1); número de estrelas, só vai até 5.

streetAddr: varchar(100); rua e número, valor alto por precaução.

city: varchar(50); nome da cidade normalmente é pequeno.

province: varchar(50); nome de estado também é pequeno.

user => name: varchar(100); valor grande para comportar nomes exóticos.

phoneNum: char(13); char com valor fixo, já que número de telefone tem um padrão.

service\_point => streetAddr: varchar(250); endereço completo.

province: varchar(50); apenas nome do estado.

city: varchar(50); apenas o nome da cidade.

startTime: varchar(50); texto para aceitar o modo do input (ex. '5:00 am').

endTime: varchar(50); mesmo motivo acima.

comments => creationTime: datetime; para aceitar tanto data quanto hora.

grade: float(2,1); para aceitar notas de 0,1 até 5,0.

content: varchar(500); tamanho grande para aceitar comentários detalhados.