**INF 280**

**Final Project Submission**

**Technology Store**

**Team Members:**

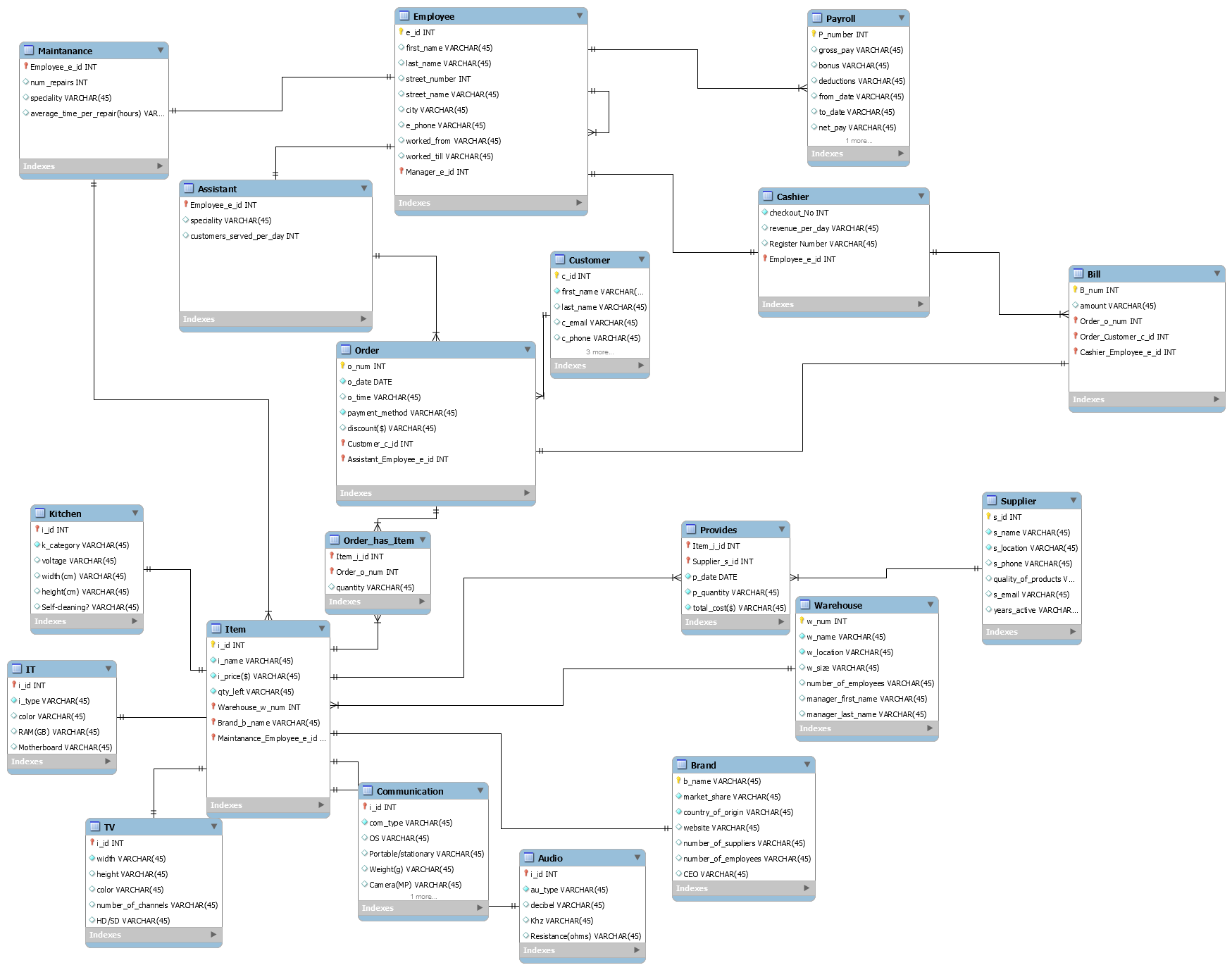
Babev, Nikolay, Leader  
Amgalantuul, Ulsbold, Designer  
Pavlova, Ivelina, Designer  
Robo, Xhoni, Developer  
Zahariev, Nikolay, Developer

**Description**:

Our project represent a database for a large-scale technology Store. The technology store has a lot of analytical data and the database will facilitate this process. First and foremost – the store has employees.. First when implementing we had a manager entity, but we decided to change it with payroll, as it was better implementable. Employees can be either assistant or a part of the maintenance team. Assistants are those who help customers to place orders. Each order contains items and is documented in the data base.

Items are distributed into subclasses such as TV, AUDIO, ACCESSORIES, IT, KITCHEN, COMMUNICATIONS. The subclasses consist of different types of technology such as chargers and batteries belong to ACCESSORIES subclass. Each item belongs to a brand that also has its own attributes in the system. Items are delivered to the store by different suppliers so the data base contains their distractive attributes. The characteristics of the customers are also recorded in the data base as each customer places an order of items. Items are not all stored in one place. Some of them are on storage and other are available in the store, therefore the attributes of the warehouses are also listed in the data base. We also have cashiers and bills as entities. During the process of implementation our database was subject to change few times.

**EER Relational Model**



**Normalization:**

All the tables have been normalized to be of the form - 3NF.

**Tools used to develop the project:**

* My SQL workben
* PhpMyAdmin

**Process of development:**

Once we had the Relational model in MySQL it was relatively easy to start developing the database. We used the help of mysql creation to accumulate the queries for create tables. We had to modify them a little bit according to our needs. Once we had the database ready, we had to insert the data. The insertion was a little bit bothersome, but we learned a lot about the Insert statement.

We had a disjoint set for the attribute of TV,communication, it, audio, Kitchen, but we couldn’t implement it completely in mysql so we decided to hard code it: every itemId as a prefix, depending on the group it is in.

**Queries:**

**1.First Query**

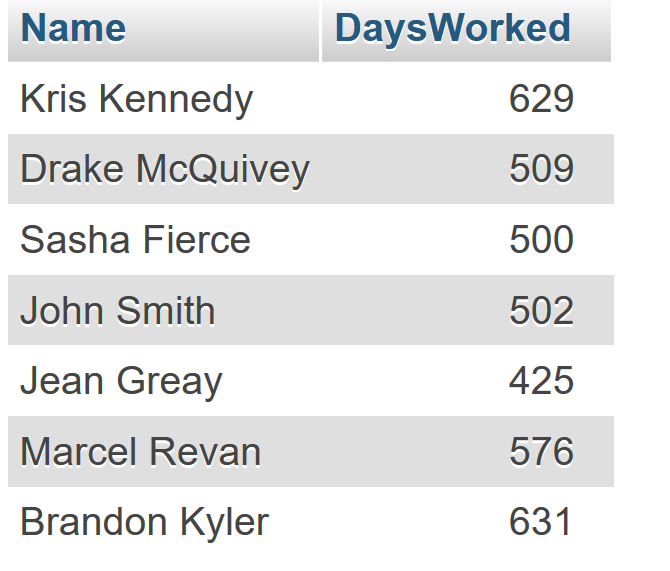
SELECT CONCAT(`first\_name`,' ',`last\_name`) As Name,(DATEDIFF (`worked\_till`,`worked\_from`)) As DaysWorked

FROM Employee

WHERE (DATEDIFF (`worked\_till`,`worked\_from`)) > 365

**Displays the employees who worked for more than a year.**

**Output:**

****

**2.Second Query**

SELECT CONCAT(`first\_name`,' ',`last\_name`) as Name, Employee.e\_id as ID , payroll.net\_pay as Wage

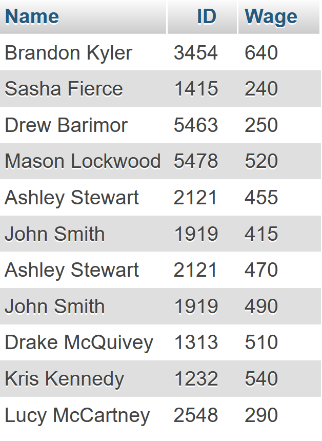
FROM Employee

INNER JOIN Payroll

ON Employee.e\_id = payroll.Employee\_e\_id

**Displays employees and respectively their wage.**

**Output:**

****

**3. Third Query**

SELECT first\_name as 'First Name', maintanance.num\_repairs as 'Number of Repairs'

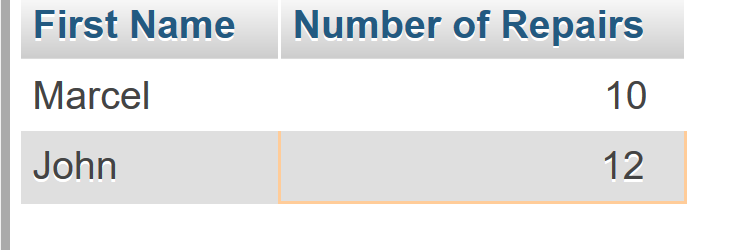
FROM employee

INNER JOIN maintanance

ON maintanance.Employee\_e\_id = employee.e\_id

WHERE maintanance.num\_repairs >= 10

**Output:**



**Selects the the first name of all maintenance employees with more than 10 repairs.**

4. **Fourth Query**

SELECT customer.first\_name As 'First Name', CONCAT(c\_street\_num,c\_street\_name,c\_city) As Adress

FROM customer

WHERE customer.C\_id IN

(SELECT `order`.Customer\_c\_id

FROM `order`

WHERE `order`.payment\_method = 'cash'

AND `order`.o\_num IN

(SELECT Order\_has\_item.Order\_o\_num

FROM Order\_has\_item

WHERE quantity > 4))

UNION

SELECT customer.first\_name As 'First Name', customer.c\_email As 'E-mail'

FROM customer

WHERE customer.C\_id IN

(SELECT `order`.Customer\_c\_id

FROM `order`

WHERE `order`.payment\_method = 'credit'

AND `order`.o\_num IN

(SELECT Order\_has\_item.Order\_o\_num

FROM Order\_has\_item

WHERE quantity > 4))

**The company has decided they want to send Christmas cards to their customers – specially preferred customers, who have 4 items in their orders. That being said, people who pay with cash are more likely to check their regular mail, while people who pay with credit cards are more likely to use electronic mail. This query outputs the first name and the address or respectively the e-mail needed for the card sender to make send the cards.**

**Output:**



**5.Fifth Query**

SELECT Brand.website

FROM Brand

INNER JOIN Item

ON Item.Brand\_b\_name = Brand.b\_name

WHERE Item.i\_id IN (

SELECT `order\_has\_item`.`Item\_i\_id`

FROM `order\_has\_item` INNER JOIN

(SELECT `provides`.`Item\_i\_id`

FROM provides

WHERE p\_quantity > 50) as TempProvidesTable

ON `order\_has\_item`.`Item\_i\_id` = TempProvidesTable.`Item\_i\_id`

WHERE `order\_has\_item`.quantity > 2)

**This query has the purpose to show popular brands. A popular brand is one, whose items are being ordered more 2 at a time and are being supplied more than 50 at a time by the supplier. The query shows company websiteds.**

**Output:**

****

**6.Sixth Query**

SELECT Employee.city as 'City',customer.last\_name AS 'Customer First Name', employee.first\_name AS 'Assistant Name', employee.e\_phone as 'Assistant number'

FROM Employee

INNER JOIN Customer

ON Employee.city = Customer.c\_city

WHERE e\_id IN

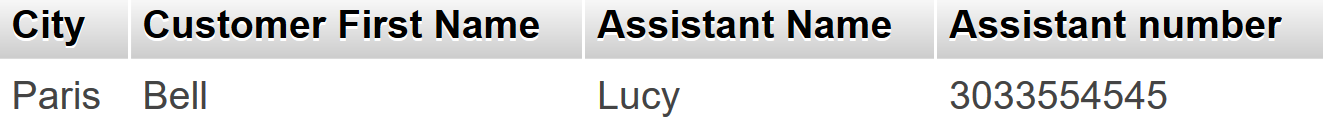
(SELECT Assistant.Employee\_e\_id

FROM Assistant

WHERE employee.worked\_till < '2016-01-01' AND assistant.speciality IN ('TV','Kitchen'))

**This query’s purpose is to show customers the assistants who live in their city and are specialists in home-related products(Tv/Kitchen). It displays Customer last name(customers usually are referred by first name) and assistant’ first name (employees use usually first name) who are from the same city . Also that assistant is still working in 2015.**

**Output:**

****

**7. Seventh Query**

SELECT

\*

FROM

Item

WHERE

i\_id IN (SELECT

i\_id

FROM

Kitchen)

AND Warehouse\_w\_num IN (SELECT

w\_num

FROM

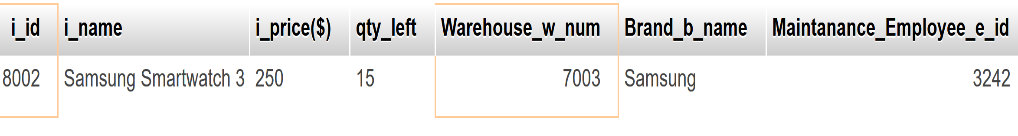
warehouse

WHERE

number\_of\_employees > 10 AND w\_size > 70);

**Find Kitchen items stored in a warehouse that has more than 10 employees and that has a size of more than 70.**

**Output:**

****

**8.Eight Query**

SELECT

employee.e\_id, employee.first\_name, employee.last\_name, employee.Manager\_e\_id

FROM

employee

INNER JOIN

assistant ON (employee.e\_id = assistant.Employee\_e\_id)

INNER JOIN

(SELECT

\*

FROM

payroll

WHERE

net\_pay > (SELECT

AVG(net\_pay)

FROM

payroll)) AS A ON (e\_id = A.Employee\_e\_id)

WHERE

e\_id NOT IN (SELECT

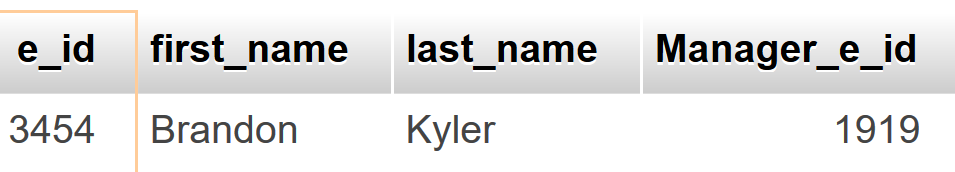
manager\_e\_id

FROM

employee)

**Find all employees who are assistants, who received a net pay greater than the average net pay for all employees, and who are not managers.**

**Output:**

****

**9**. **Ninth Query**

SELECT

\*

FROM

customer

WHERE

EXISTS( SELECT

\*

FROM

`order`

WHERE

customer.c\_id = `order`.Customer\_c\_id

AND o\_num IN (SELECT

order\_o\_num

FROM

order\_has\_item

WHERE

order\_has\_item.Item\_i\_id IN (SELECT

i\_id

FROM

item

WHERE

i\_id IN (SELECT

i\_id

FROM

kitchen)

AND Brand\_b\_name IN ('Apple' , 'Acer'))

AND quantity > 2)

AND EXISTS( SELECT

\*

FROM

bill

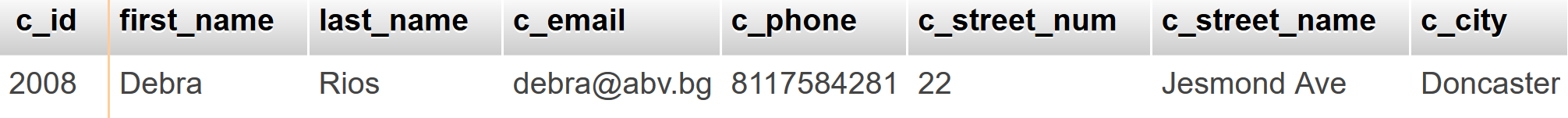
WHERE

amount > 3000

AND bill.Order\_o\_num = `order`.o\_num));

**List all clients who ordered more than 2 kitchen items from either Apple or Acer and who paid more than $3000 for their order.**

**Output:**

****

**10. Tenth Query**

SELECT

\*

FROM

bill

WHERE

Order\_o\_num IN (SELECT

order\_o\_num

FROM

order\_has\_item

WHERE

Item\_i\_id IN (SELECT

i\_id

FROM

item

WHERE

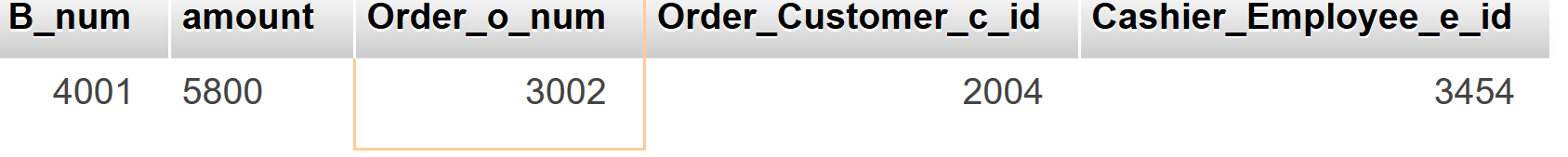
`i\_price($)` >= 500)

GROUP BY Order\_o\_num

HAVING (COUNT(Order\_o\_num) > 1));

**Sum up the amount paid for all orders which include at least two different items with each item having a price over $500.**

**Output:**

****

**11.Eleventh Query**

SELECT customer.last\_name as 'Customer Name' , SUM(`order`.`discount($)`) as 'Accumulated discount'

FROM customer

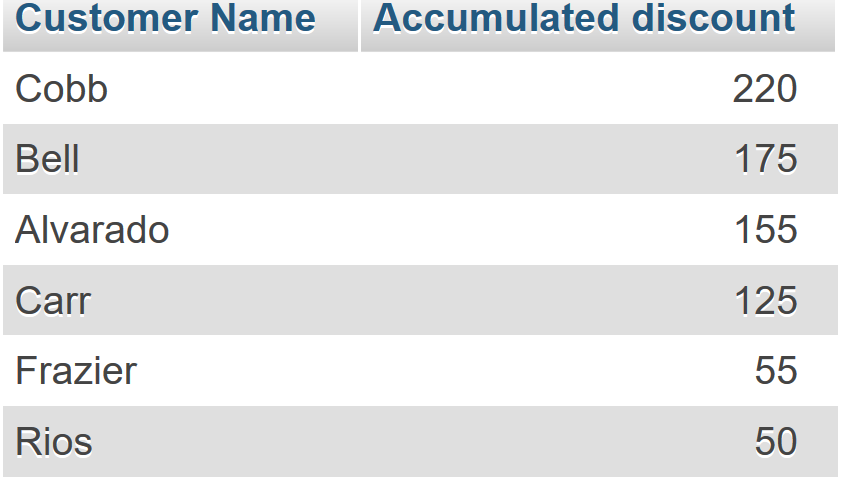
INNER JOIN `order` ON customer.c\_id = `order`.Customer\_c\_id

WHERE `order`.`discount($)` > 0

Group BY customer.last\_name

Order BY SUM(`order`.`discount($)`) DESC

**Output:**



**Shows the accumulated discount by each customer in descenig order.**

**12. Twelvth Query**

SELECT `item`.`i\_name`, `warehouse`.`w\_location`

FROM `warehouse`

INNER JOIN item ON `warehouse`.`w\_num` = item.`Warehouse\_w\_num`

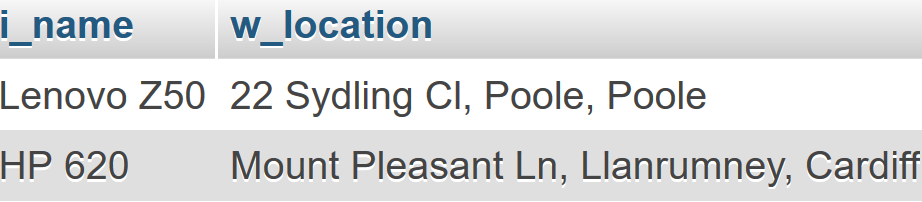
WHERE `Brand\_b\_name` IN (

SELECT b\_name

FROM `brand`

WHERE `brand`.`country\_of\_origin` = 'China')

**Output**

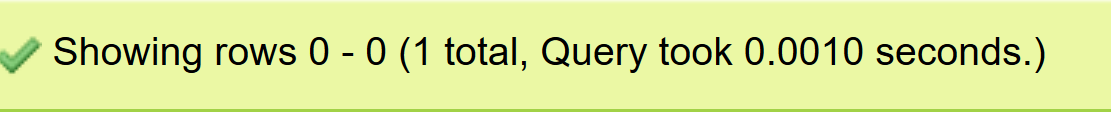


**Where are the items stored now, which are made by a brand from China**

**Query Optimization**

**All queries are optimized due to the fact of the small amount of data used. All queries execute in really high speed. The only way that would reduce the speed is calculating which rows are present, but still negligibly small.**

**For example:**

****