ISO 9001:2015
SOCOTEC Certificate No. SCP000722Q

REPUBLIC OF THE PHILIPPINES
BICOL UNIVERSITY
POLANGUI
Email: bu-cpro@bicol-u.edu.ph | bupc.bu@gmail.com
jULY 13, 2024

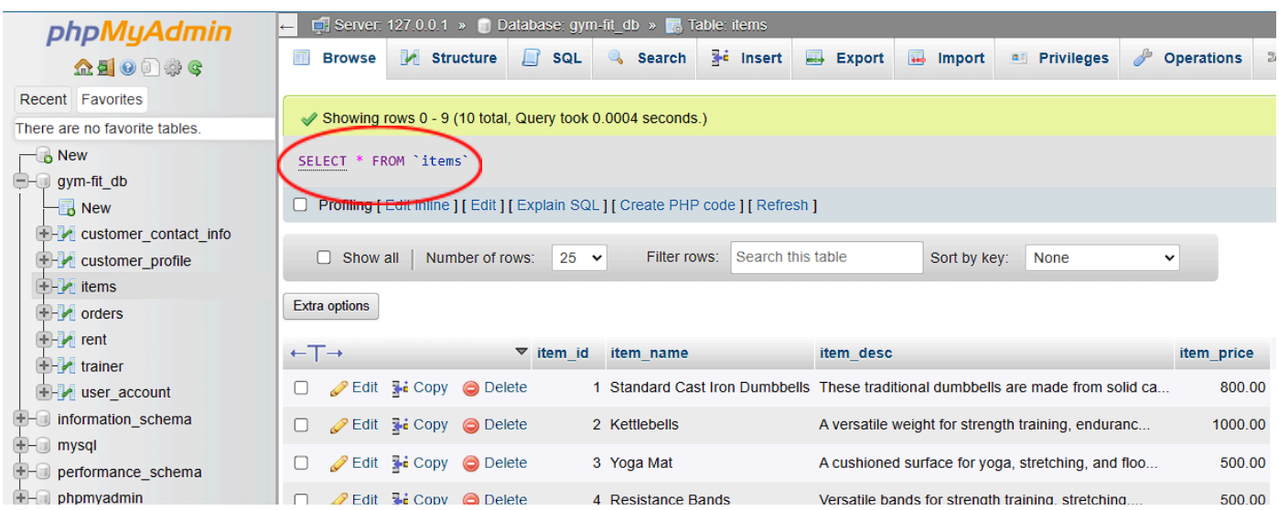
SUBJECT: WEB SYSTEMS

TEAM: 404 NOT FOUND

MEMBERS:

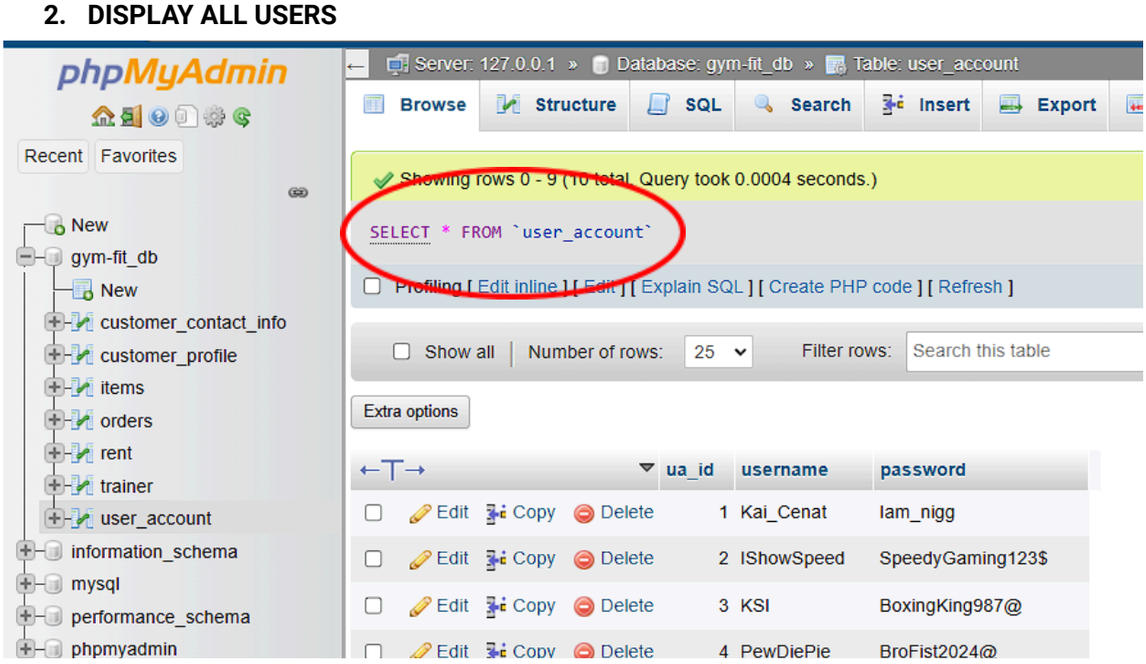
- **JOHN ELMER BLANQUISCO BOBIER JR.**
- **ROD B. RAÑOLA**
- **CHARLS EMIL C. BARQUIN**
- **CHRISTIAN JERIC NON**
- **Vincent Macauyam**

1. **DISPLAY ALL ITEMS:**

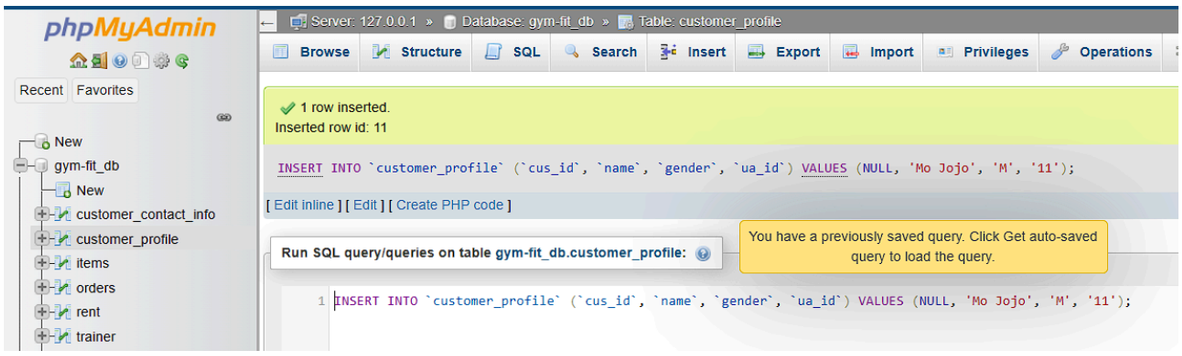


In this step we inserted the name of the items, the item prices, and their own descriptions. After that we displayed the information for the item table.

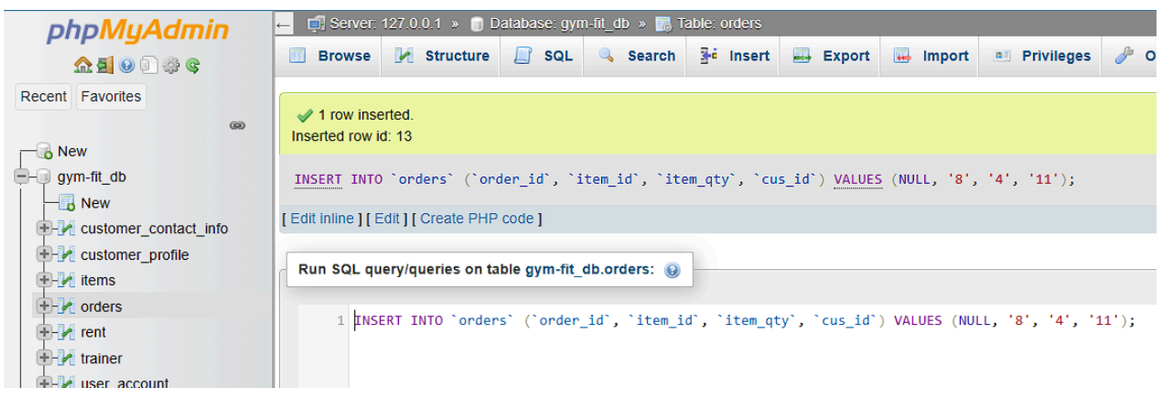
2. **DISPLAY ALL USERS**



We then created a new table for the users containing user-names and passwords. We then displayed the data for the user table.

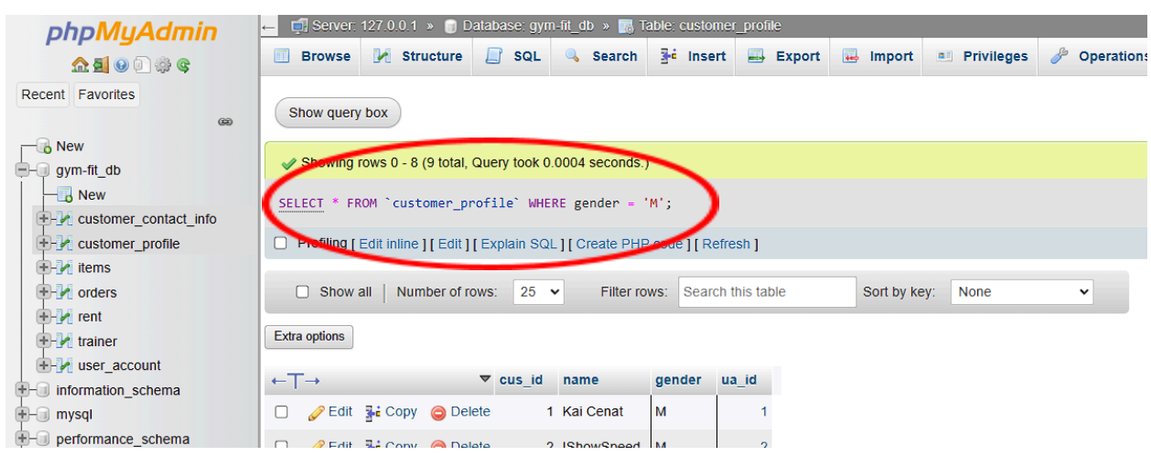


We add some records to the Customer_profile table, having ensured that the ua_id should be in the users_account table. That was done for the purpose of verifying that there is no mismatch between customer profiles and user accounts in order to provide better control in terms of interaction with customers' data and acting upon it.



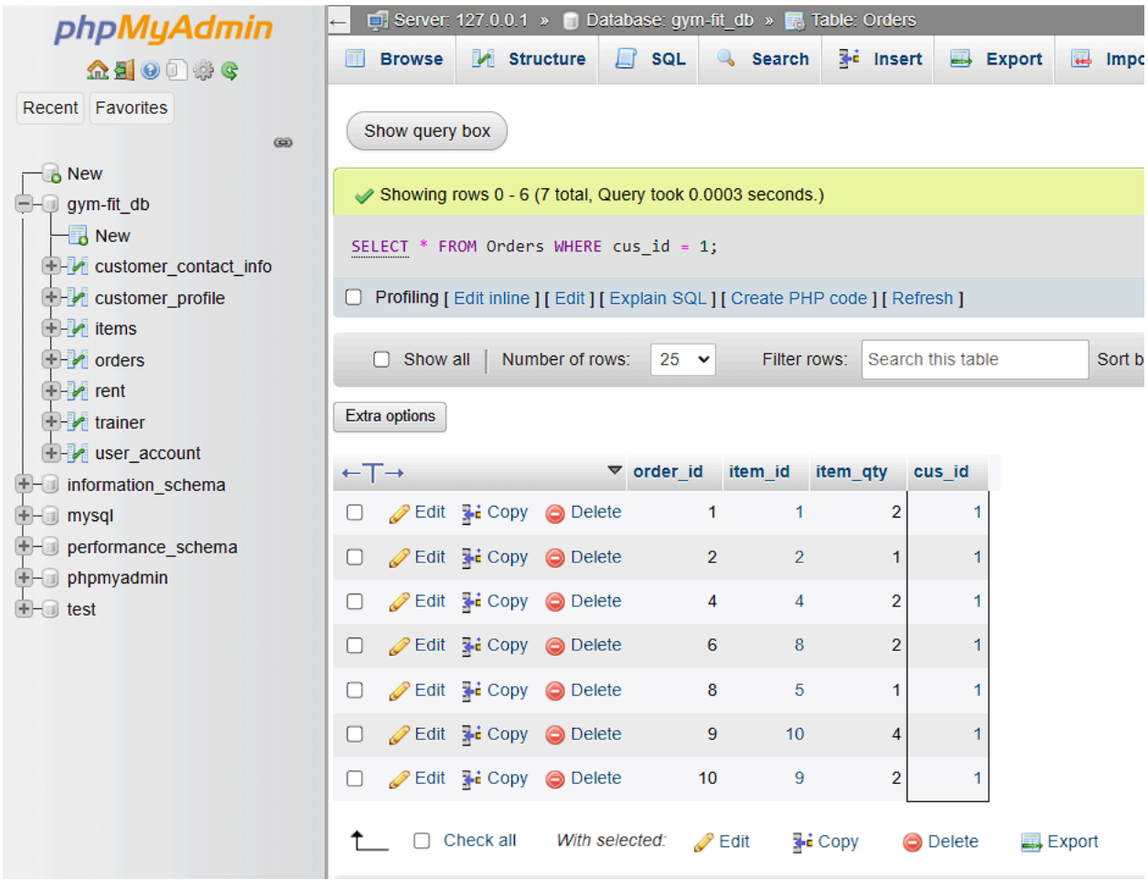
We inserted orders in the Orders table, with the item_id you insert matching those listed within the item_ids and the cus_id already in the customer_profile table. This holds data integrity and relationships between orders, items, and customers are accurate. By enforcing these kinds of constraints, we can make the process more efficient about order processing and overall reliability of the database.

3. USING WHERE CLAUSE:

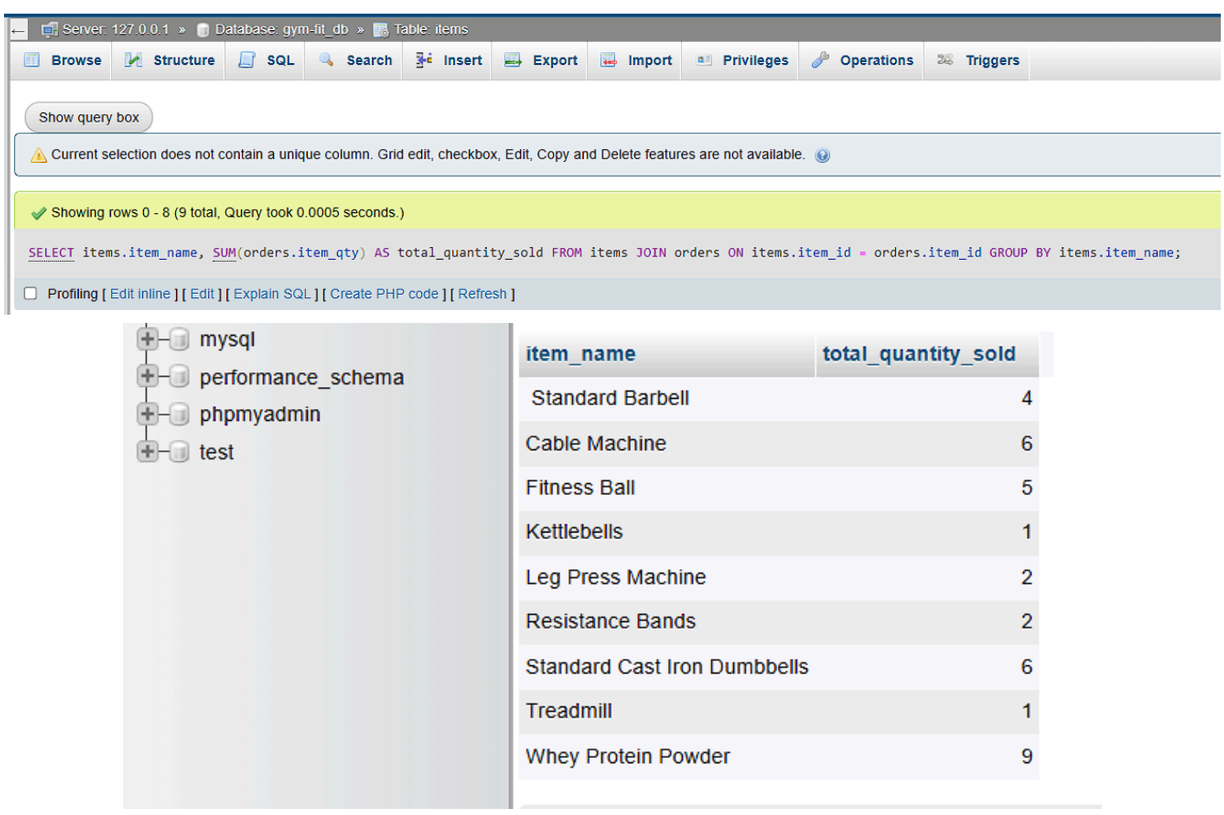


This query would list all the customers in the customer profile table who are males or identified by an "M" in the gender category. You would find this helpful to know about your demographics of customer and would allow targeted marketing strategies to really have more resonance with the male audience.

4. DISPLAYING ALL ORDERS OF CUSTOMER ID 1 OR CUS_ID 1:



5. DISPLAY SALES:



We presented the total quantity sold for each product while making sure to only send descriptive columns and total quantity sold and nothing else in output. It's one way we can easily present our information in a clear and usable manner. Putting emphasis on the

quantities, we can be better assured of which products are doing well. In the end, it helps us have well-informed decisions about our inventory and sales strategies.

6. DISPLAYING ITEMS HAVING MORE THAT 0 SALES

Showing rows 0 - 8 (9 total, Query took 0.0005 seconds.)

```
SELECT items.item_name, SUM(orders.item_qty) AS total_quantity_sold FROM items JOIN orders ON items.item_id = orders.item_id GROUP BY items.item_name;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Extra options

item_name	total_quantity_sold
Standard Barbell	4
Cable Machine	6
Fitness Ball	5
Kettlebells	1
Leg Press Machine	2
Resistance Bands	2
Standard Cast Iron Dumbbells	6
Treadmill	1
Whey Protein Powder	9

Show all | Number of rows: 25 Filter ro

7. DISPLAY SALES:

```
SELECT i.item_name, SUM(o.item_qty * i.item_price) AS sales_amount FROM items AS i JOIN orders AS o ON i.item_id = o.item_id GROUP BY i.item_name;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Extra options

item_name	sales_amount
Standard Barbell	6000.00
Cable Machine	57000.00
Fitness Ball	1500.00
Kettlebells	1000.00
Leg Press Machine	17000.00
Resistance Bands	1000.00
Standard Cast Iron Dumbbells	4800.00
Treadmill	9000.00
Whey Protein Powder	13500.00

Show all | Number of rows: 25 Filter rows: Search this table

Query results operations

We printed the sales per item. In that sheet, we only needed to print the item name and the sales amount. That way, we got to see clearly how every product is doing on a revenue basis. Being able to narrow it down to just those two columns helped us limit the information so we could analyze better. Meanwhile, we were able to quickly pin down which were our best-selling items. And overall, this exercise really gave us a good insight into the trends of sales, as it enabled us to strategize for future inventory. So I am just pleased how it all came out.

8. DISPLAYING THE TOTAL SALES PER DAY

Server: 127.0.0.1 » Database: gym-fit_db » Table: orders

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Triggers

Show query box

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 1 (2 total, Query took 0.0004 seconds.)

```
SELECT DATE(orders.date) AS order_day, SUM(orders.item_qty * items.item_price) AS sales_per_day FROM orders JOIN items ON items.item_id = orders.item_id GROUP BY DATE(orders.date);
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

order_day	sales_per_day
2024-10-09	800.00
2024-10-18	111600.00

Show all | Number of rows: 25 Filter rows: Search this table

9. DISPLAYING THE TOTAL SALES AND QUANTITY PER ITEM AND DAY

Showing rows 0 - 8 (9 total, Query took 0.0005 seconds.)

```
SELECT items.item_name, SUM(orders.item_qty * items.item_price) as Totals_sales, SUM(orders.item_qty) as Quantity_per_item FROM orders JOIN items ON items.item_id = orders.item_id GROUP BY items.item_name;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

item_name	Totals_sales	Quantity_per_item
Standard Barbell	6000.00	4
Cable Machine	57000.00	6
Fitness Ball	1500.00	5
Kettlebells	1000.00	1
Leg Press Machine	17000.00	2
Resistance Bands	1000.00	2
Standard Cast Iron Dumbbells	6400.00	8
Treadmill	9000.00	1
Whey Protein Powder	13500.00	9

Show all | Number of rows: 25 Filter rows: Search this table

10. DISPLAYING THE TOTAL SALES TOTAL QUANTITY SOLD PER ITEM AND DAY

Showing rows 0 - 9 (10 total, Query took 0.0004 seconds.)

```
SELECT items.item_name, SUM(items.item_price * orders.item_qty) AS Total_Sales, DATE(orders.date) AS Order_day, SUM(orders.item_qty) AS Total_Quantity FROM orders JOIN items ON items.item_id = orders.item_id GROUP BY items.item_name, DATE(orders.date) ORDER BY `Order_day` DESC
```

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

item_name	Total_Sales	Order_day 1	Total_Quantity
Standard Cast Iron Dumbbells	5600.00	2024-10-18	7
Kettlebells	1000.00	2024-10-18	1
Fitness Ball	1500.00	2024-10-18	5
Resistance Bands	1000.00	2024-10-18	2
Cable Machine	57000.00	2024-10-18	6
Whey Protein Powder	13500.00	2024-10-18	9
Treadmill	9000.00	2024-10-18	1
Standard Barbell	6000.00	2024-10-18	4
Leg Press Machine	17000.00	2024-10-18	2
Standard Cast Iron Dumbbells	800.00	2024-10-09	1

11. DISPLAYING THE TOTAL SALES AND TOTAL QUANTITY SOLD PER FULLNAME

Showing rows 0 - 6 (7 total, Query took 0.0006 seconds.)

```
SELECT customer_profile.name AS fullname, SUM(orders.item_qty * items.item_price) AS total_sales, SUM(orders.item_qty) AS total_quantity_sold FROM orders JOIN items ON items.item_id = orders.item_id JOIN customer_profile ON customer_profile.cus_id = orders.cus_id GROUP BY customer_profile.name;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

fullname	total_sales	total_quantity_sold
Batman	52000.00	8
CongTV	11000.00	6
IShowSpeed	800.00	1
Kai Cenat	38600.00	14
KSI	800.00	1
Mo Jojo	6000.00	4
MrBeast	3200.00	4

Show all | Number of rows: 25 Filter rows: Search this table

Query results operations

12. DISPLAYING THE NUMBER OF ITEMS BROUGHT BY CUSTOMERS USING COUNT(*):

Showing rows 0 - 6 (7 total, Query took 0.0004 seconds.)

```
SELECT user_account.username, COUNT(*) AS items_bought FROM orders JOIN customer_profile ON customer_profile.ua_id = orders.cus_id JOIN user_account ON user_account.ua_id = customer_profile.ua_id GROUP BY user_account.username;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

username	items_bought
CongTV	2
ImBatman	2
IShowSpeed	1
Kai_Cenat	7
KSI	1
Mo JOJO	1
MrBeast	1

Show all | Number of rows: 25 Filter rows: Search this table

Query results operations

In this step, we displayed the number of items bought by each username. To achieve this, we used the COUNT(*) function in our SQL query, which allowed us to count the total number of purchases associated with each user. By grouping the results by username, we could easily see how many items each individual had bought, providing valuable insights into user activity and engagement in our application. This analysis helps us understand purchasing patterns and can inform future marketing strategies.

13. USING WILDCARDS:

Showing rows 0 - 7 (8 total, Query took 0.0004 seconds.)

```
SELECT items.item_name FROM items WHERE items.item_name LIKE '%A%';
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Extra options

←T→			item_name
Edit	Copy	Delete	Standard Cast Iron Dumbbells
Edit	Copy	Delete	Yoga Mat
Edit	Copy	Delete	Resistance Bands
Edit	Copy	Delete	Treadmill
Edit	Copy	Delete	Fitness Ball
Edit	Copy	Delete	Cable Machine
Edit	Copy	Delete	Leg Press Machine
Edit	Copy	Delete	Standard Barbell

Check all *With selected:* Edit Copy

Show all | Number of rows: 25 Filter rows: Sea

Query results operations

Showing rows 0 - 0 (1 total, Query took 0.0003 seconds.)

```
SELECT items.item_name FROM items WHERE items.item_name LIKE 'S%';
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

←T→			item_name
Edit	Copy	Delete	Standard Cast Iron Dumbbells

Check all *With selected:* Edit Copy Delete Export

Here, we used the wildcards to display the items from a particular string starting. We used the SQL LIKE operator along with a wildcard character to filter out our result and make sure only those items were displayed which matched our search criteria. Using this method made it easier and faster for us to pick the relevant items. This is a very useful technique, especially in cases where we would like to highlight specific categories or brands in our inventory.