```sql
-- Create the database
CREATE DATABASE global_store_db;
USE global_store_db;

-- Create the products table
CREATE TABLE PRODUCTS (
    product_id INT PRIMARY KEY,
    name VARCHAR(100),
    price DECIMAL(10,2),
    quantity INT
);
```

Output

Action Output

#	Time	Action	Message
1	23:23:12	CREATE DATABASE global_store_db	1 row(s) affected
2	23:23:15	USE global_store_db	0 row(s) affected
3	23:23:24	CREATE TABLE PRODUCTS (product_id INT PRIMARY KEY, name VARCHA...	0 row(s) affected

```sql
-- Create the orders table
CREATE TABLE orders (
    order_id INT PRIMARY KEY,
    product_id INT,
    quantity_ordered INT,
    order_date DATE
);
```

Output

Action Output

#	Time	Action	Message
1	23:23:12	CREATE DATABASE global_store_db	1 row(s) affected
2	23:23:15	USE global_store_db	0 row(s) affected
3	23:23:24	CREATE TABLE PRODUCTS (product_id INT PRIMARY KEY, name VARCHA...	0 row(s) affected
4	23:24:25	CREATE TABLE orders (order_id INT PRIMARY KEY, product_id INT, quantit...	0 row(s) affected

```
23      -- Alter the products table to add the category column
24 •    ALTER TABLE PRODUCTS ADD COLUMN category VARCHAR(50) AFTER price;
25
26
```

Output

Action Output

#	Time	Action	Message
1	23:23:12	CREATE DATABASE global_store_db	1 row(s) affected
2	23:23:15	USE global_store_db	0 row(s) affected
3	23:23:24	CREATE TABLE PRODUCTS (product_id INT PRIMARY KEY, name VARCHA...	0 row(s) affected
4	23:24:25	CREATE TABLE orders (order_id INT PRIMARY KEY, product_id INT, quantit...	0 row(s) affected
5	23:25:02	ALTER TABLE PRODUCTS ADD COLUMN category VARCHAR(50) AFTER price	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

```
25
26      -- Rename products table to inventory
27 •    ALTER TABLE products RENAME TO inventory;
28
29
30
31
```

Output

Action Output

#	Time	Action	Message
2	23:23:15	USE global_store_db	0 row(s) affected
3	23:23:24	CREATE TABLE PRODUCTS (product_id INT PRIMARY KEY, name VARCH...	0 row(s) affected
4	23:24:25	CREATE TABLE orders (order_id INT PRIMARY KEY, product_id INT, quan...	0 row(s) affected
5	23:25:02	ALTER TABLE PRODUCTS ADD COLUMN category VARCHAR(50) AFTER price	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
6	23:25:35	ALTER TABLE products RENAME TO inventory	0 row(s) affected

```sql
-- Insert records into the inventory table
INSERT INTO inventory (product_id, name, price, quantity, category) VALUES
(101, 'Product 1', 10.00, 20, 'Category A'),
(102, 'Product 2', 15.50, 15, 'Category B'),
(103, 'Product 3', 20.25, 30, 'Category A'),
(104, 'Product 4', 8.75, 10, 'Category C'),
(105, 'Product 5', 12.00, 25, 'Category B'),
(106, 'Product 6', 18.50, 5, 'Category A'),
(107, 'Product 7', 22.75, 0, 'Category C'),
(108, 'Product 8', 30.00, 8, 'Category B'),
(109, 'Product 9', 11.50, 12, 'Category A'),
(110, 'Product 10', 25.00, 3, 'Category C');

```

Output

Action Output

#	Time	Action	Message
3	23:23:24	CREATE TABLE PRODUCTS (product_id INT PRIMARY KEY, name VARCH...	0 row(s) affected
4	23:24:25	CREATE TABLE orders (order_id INT PRIMARY KEY, product_id INT, quan...	0 row(s) affected
5	23:25:02	ALTER TABLE PRODUCTS ADD COLUMN category VARCHAR(50) AFTER price	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
6	23:25:35	ALTER TABLE products RENAME TO inventory	0 row(s) affected
7	23:26:14	INSERT INTO inventory (product_id, name, price, quantity, category) VALUES (101, ...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0

```sql
SELECT * FROM INVENTORY;

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

product_id	name	price	category	quantity
101	Product 1	10.00	Category A	20
102	Product 2	15.50	Category B	15
103	Product 3	20.25	Category A	30
104	Product 4	8.75	Category C	10
105	Product 5	12.00	Category B	25
106	Product 6	18.50	Category A	5
107	Product 7	22.75	Category C	0
108	Product 8	30.00	Category B	8
109	Product 9	11.50	Category A	12
110	Product 10	25.00	Category C	3
NULL	NULL	NULL	NULL	NULL

Result Grid

Form Editor

Field Types

```sql
-- Insert records into the orders table
INSERT INTO orders (order_id, product_id, quantity_ordered, order_date) VALUES
(1, 102, 5, '2024-05-20'),
(2, 110, 2, '2024-05-21'),
(3, 105, 15, '2024-05-22'),
(4, 108, 3, '2024-05-23'),
(5, 102, 10, '2024-05-24');

SELECT * FROM ORDERS;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

order_id	product_id	quantity_ordered	order_date
1	102	5	2024-05-20
2	110	2	2024-05-21
3	105	15	2024-05-22
4	108	3	2024-05-23
5	102	10	2024-05-24
NULL	NULL	NULL	NULL

Result Grid | Form Editor

```sql
-- Update inventory table
UPDATE INVENTORY INNER JOIN ORDERS ON INVENTORY.PRODUCT_ID = ORDERS.PRODUCT_ID
SET INVENTORY.QUANTITY = INVENTORY.QUANTITY - ORDERS.QUANTITY_ORDERED;

SELECT * FROM INVENTORY;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

product_id	name	price	category	quantity
101	Product 1	10.00	Category A	20
102	Product 2	15.50	Category B	10
103	Product 3	20.25	Category A	30
104	Product 4	8.75	Category C	10
105	Product 5	12.00	Category B	10

Result Grid

```sql
-- Display distinct categories from the inventory table
SELECT DISTINCT category FROM inventory;
```

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

category
Category A
Category B
Category C

Result Grid

```sql
-- Select the top 5 products by their prices in descending order from the inventory table
SELECT * FROM inventory ORDER BY price DESC LIMIT 5;
```

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

product_id	name	price	category	quantity
108	Product 8	30.00	Category B	5
110	Product 10	25.00	Category C	1
107	Product 7	22.75	Category C	0
103	Product 3	20.25	Category A	30
106	Product 6	18.50	Category A	5
NULL	NULL	NULL	NULL	NULL

Result Grid | Form Editor

```sql
-- Display the names of products with a quantity greater than 10 from the inventory table
SELECT name FROM inventory WHERE quantity > 10;
```

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

name
Product 1
Product 3
Product 9

Result Grid | Form Editor

```sql
-- Use the SUM() function to calculate the total price of all products in the inventory table
SELECT SUM(price * quantity) AS total_price FROM inventory;
```

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

total_price
1575.50

Result Grid | Form Editor

```sql
-- Group products by their categories and display the count of products in each category
SELECT category, COUNT(*) AS product_count FROM inventory GROUP BY category;
```

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

category	product_count
Category A	4
Category B	3
Category C	3

Result Grid | Form Editor

```
76    -- Write a query to identify products that are currently out of stock (i.e., quantity is zero). Display th
77 •  SELECT name, price FROM inventory WHERE quantity = 0;
78
```

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

name	price
Product 7	22.75

```
79    -- Create a view named expensive_products that displays the details of products with a price above the ave
80 •  CREATE VIEW expensive_products AS
81    SELECT * FROM inventory WHERE price > (SELECT AVG(price) FROM inventory);
82 •  SELECT * FROM EXPENSIVE_PRODUCTS;
```

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

product_id	name	price	category	quantity
103	Product 3	20.25	Category A	30
106	Product 6	18.50	Category A	5
107	Product 7	22.75	Category C	0
108	Product 8	30.00	Category B	5
110	Product 10	25.00	Category C	1

```
84    -- Write a join query to display the names of products along with the corresponding order quantities from
85 •  SELECT INVENTORY.NAME AS product_name, ORDERS.quantity_ordered AS order_quantity
86    FROM inventory INNER JOIN orders  ON INVENTORY.product_id = ORDERS.product_id;
87
```

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

product_name	order_quantity
Product 2	5
Product 10	2
Product 5	15
Product 8	3
Product 2	10