

## SQL Drill Item 34

*NOTE: These exercises are based on the dataset defined at [https://en.wikibooks.org/wiki/SQL\\_Exercises/The\\_computer\\_store#Exercises](https://en.wikibooks.org/wiki/SQL_Exercises/The_computer_store#Exercises)*


1. Select the names of all the products in the store.

Query:

```
SELECT Name  
FROM Products
```

Result:

Results		Messages	
	Name		
1	Hard drive		
2	Memory		
3	ZIP drive		
4	Floppy disk		
5	Monitor		
6	DVD drive		
7	CD drive		
8	Printer		
9	Toner cartridge		

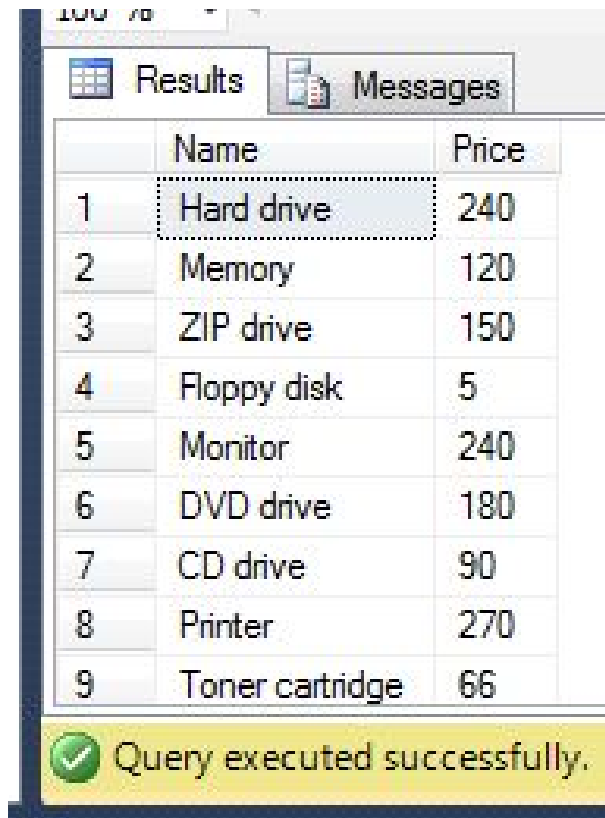
 Query executed successfully.

2. Select the names and the prices of all the products in the store.

Query:

```
SELECT Name, Price  
FROM Products
```

Result:



The screenshot shows a database application window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with the following data:

	Name	Price
1	Hard drive	240
2	Memory	120
3	ZIP drive	150
4	Floppy disk	5
5	Monitor	240
6	DVD drive	180
7	CD drive	90
8	Printer	270
9	Toner cartridge	66

Below the table, a yellow status bar with a green checkmark icon contains the text: 'Query executed successfully.'

3. Select the name of the products with a price less than or equal to \$200.

Query:

```
SELECT Name  
FROM Products  
WHERE Price <= 200
```

Result:

Results		Messages
	Name	
1	Memory	
2	ZIP drive	
3	Floppy disk	
4	DVD drive	
5	CD drive	
6	Toner cartridge	
7	DVD burner	

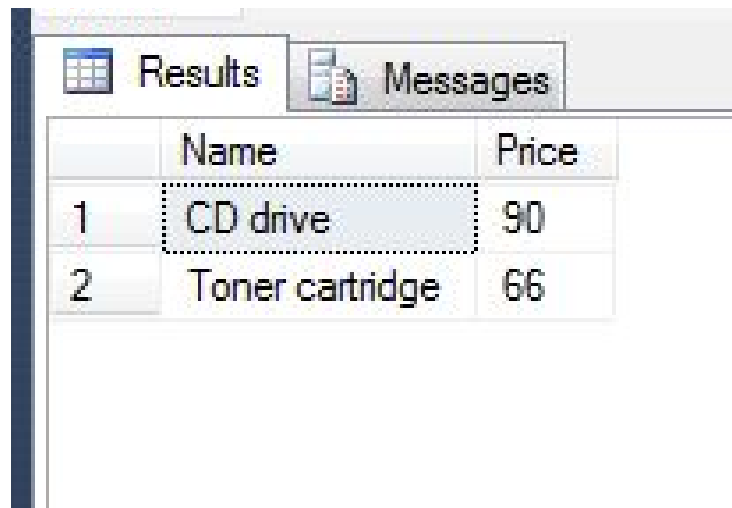
✓ Query executed successfully.

4. Select all the products with a price between \$60 and \$120. (NOTE: I assumed this range did not include the endpoints.)

Query:

```
SELECT Name, Price
FROM Products
WHERE Price > 60
AND Price < 120
```

Result:



	Name	Price
1	CD drive	90
2	Toner cartridge	66

5. Select the name and price in cents (i.e., the price must be multiplied by 100).

Query:

```
SELECT Name, Price*100 AS PriceInCents
FROM Products
```

Result:

	Name	PriceInCents
1	Hard drive	24000
2	Memory	12000
3	ZIP drive	15000
4	Floppy disk	500
5	Monitor	24000
6	DVD drive	18000
7	CD drive	9000
8	Printer	27000
9	Toner cartridge	6600

✓ Query executed successfully. | STU

6. Compute the average price of all the products.

Query:

```
SELECT AVG(Price) AS AveragePrice
FROM Products
```

Result:

	AveragePrice
1	154.1

7. Compute the average price of all products with manufacturer code equal to 2.

Query:

```
SELECT AVG(Price) AS AveragePrice2
FROM Products
WHERE Manufacturer = 2
```

Result:

Results		Messages	
AveragePrice2			
1	150		

8. Compute the number of products with a price larger than or equal to \$180.

Query:

```
SELECT COUNT(Price) AS PriceGREQ180
FROM Products
WHERE Price >= 180
```

Result:

Results		Messages	
PriceGREQ180			
1	5		

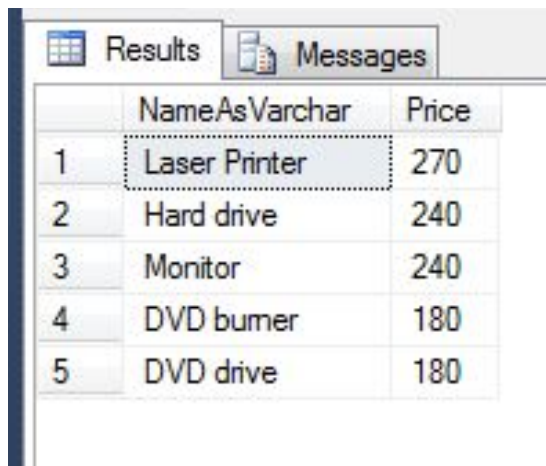
9. Select the name and price of all products with a price larger than or equal to \$180, and sort first by price (in descending order), and then by name (in ascending order).

Query:

```
SELECT CAST(Name AS VARCHAR(50)) AS NameAsVarchar, Price
FROM Products
WHERE Price >= 180
ORDER BY Price DESC, CAST(Name AS VARCHAR(50)) ASC;
```

*NOTE: The solution in the exercise produces the following error related to the column Name having a data type 'text': **The text, ntext, and image data types cannot be compared or sorted, except when using IS NULL or LIKE operator.** I found a similar workaround to this error on stackoverflow and implemented it as shown above. This solution worked in subsequent exercises. In hindsight, I could have changed the data type for this column but decided not to alter the data for this exercise.*

Result:



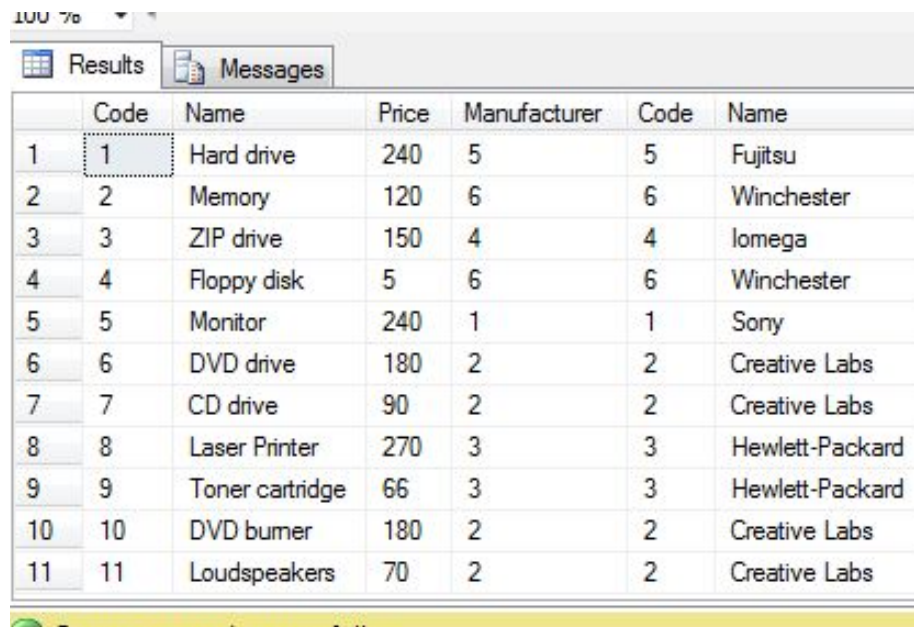
	NameAsVarchar	Price
1	Laser Printer	270
2	Hard drive	240
3	Monitor	240
4	DVD burner	180
5	DVD drive	180

10. Select all the data from the products, including all the data for each product's manufacturer.

Query:

```
SELECT *  
FROM Products  
INNER JOIN Manufacturers  
ON Products.Manufacturer = Manufacturers.Code
```

Result:



	Code	Name	Price	Manufacturer	Code	Name
1	1	Hard drive	240	5	5	Fujitsu
2	2	Memory	120	6	6	Winchester
3	3	ZIP drive	150	4	4	Iomega
4	4	Floppy disk	5	6	6	Winchester
5	5	Monitor	240	1	1	Sony
6	6	DVD drive	180	2	2	Creative Labs
7	7	CD drive	90	2	2	Creative Labs
8	8	Laser Printer	270	3	3	Hewlett-Packard
9	9	Toner cartridge	66	3	3	Hewlett-Packard
10	10	DVD burner	180	2	2	Creative Labs
11	11	Loudspeakers	70	2	2	Creative Labs



11. Select the product name, price, and manufacturer name of all the products.

Query:

```
SELECT P.Name, P.Price, M.Name
FROM Products AS P
INNER JOIN Manufacturers as M
ON P.Manufacturer = M.Code
```

Result:

	Name	Price	Name
1	Hard drive	240	Fujitsu
2	Memory	120	Winchester
3	ZIP drive	150	Iomega
4	Floppy disk	5	Winchester
5	Monitor	240	Sony
6	DVD drive	180	Creative Labs
7	CD drive	90	Creative Labs
8	Laser Printer	270	Hewlett-Packard
9	Toner cartridge	66	Hewlett-Packard
10	DVD burner	180	Creative Labs
11	Loudspeakers	70	Creative Labs

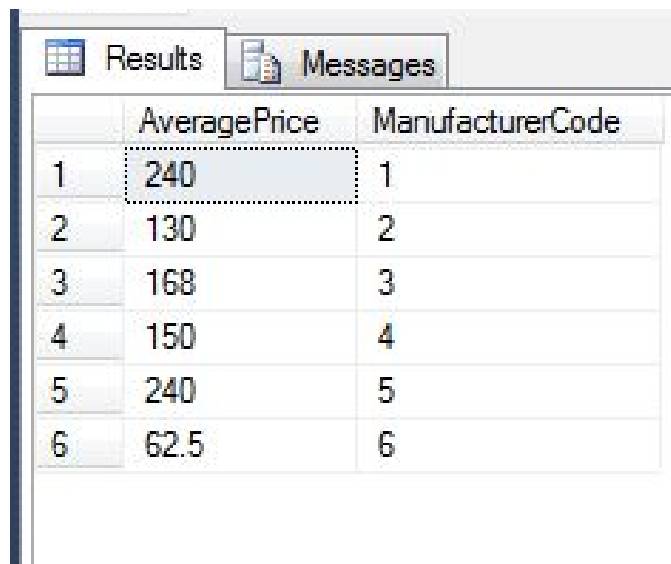
12. Select the average price of each manufacturer's products, showing only the manufacturer's code.

Query:

```
SELECT AVG(Price) AS AveragePrice, Manufacturer AS ManufacturerCode
FROM Products
GROUP BY Manufacturer
```

*NOTE: Lesson learned here - I originally did this by specifying "WHERE Manufacturer < 10" thinking I needed a WHERE clause to capture all the data for some reason. After checking the solution, I realized that I could have specified a specific manufacturer (or group of them) this way, but the problem asked for all the manufacturers and this specification was not needed.*

Result:



The screenshot shows a SQL Server interface with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with two columns: 'AveragePrice' and 'ManufacturerCode'. The table contains six rows of data, indexed 1 through 6. The first row shows an average price of 240 for manufacturer 1. The second row shows 130 for manufacturer 2. The third row shows 168 for manufacturer 3. The fourth row shows 150 for manufacturer 4. The fifth row shows 240 for manufacturer 5. The sixth row shows 62.5 for manufacturer 6.

	AveragePrice	ManufacturerCode
1	240	1
2	130	2
3	168	3
4	150	4
5	240	5
6	62.5	6

\*Note: Due to a problem with the screenshots I captured at various steps in this exercise, I re-ran most of the queries to produce new screenshots after adding the Loudspeakers item and updating the name of the Laser Printer. Thus, results may have been affected by these changes.

13. Select the average price of each manufacturer's products, showing the manufacturer's name.

Query:

```
SELECT AVG(P.Price) AS AveragePrice, CAST(M.Name AS VARCHAR(50)) AS  
ManufacturerName  
FROM Products AS P  
      INNER JOIN Manufacturers AS M  
      ON P.Manufacturer = M.Code  
GROUP BY CAST(M.Name AS VARCHAR(50))
```

Result:

	AveragePrice	ManufacturerName
1	130	Creative Labs
2	240	Fujitsu
3	168	Hewlett-Packard
4	150	lomega
5	240	Sony
6	62.5	Winchester

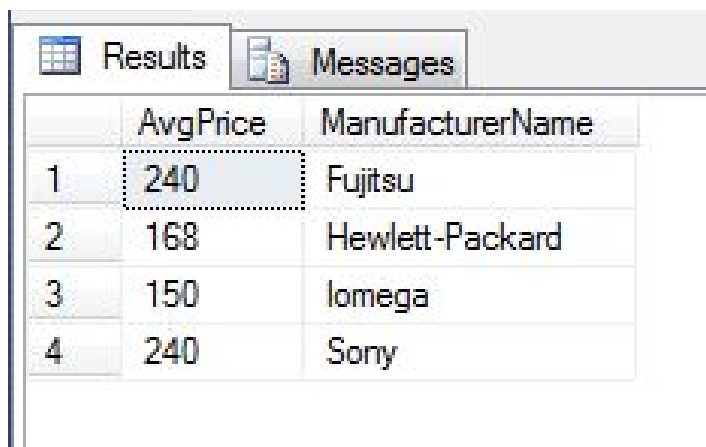
\*Note: Due to a problem with the screenshots I captured at various steps in this exercise, I re-ran most of the queries to produce new screenshots after adding the Loudspeakers item and updating the name of the Laser Printer. Thus, results may have been affected by these changes.

14. Select the names of manufacturer whose products have an average price larger than or equal to \$150.

Query:

```
SELECT AVG(P.Price) AS AvgPrice, CAST(M.Name AS VARCHAR(50)) AS  
ManufacturerName  
FROM Products AS P  
      INNER JOIN Manufacturers AS M  
      ON P.Manufacturer = M.Code  
GROUP BY CAST(M.Name AS VARCHAR(50))  
HAVING AVG(P.Price) >= 150
```

Result:



	AvgPrice	ManufacturerName
1	240	Fujitsu
2	168	Hewlett-Packard
3	150	lomega
4	240	Sony

\*Note: Due to a problem with the screenshots I captured at various steps in this exercise, I re-ran most of the queries to produce new screenshots after adding the Loudspeakers item and updating the name of the Laser Printer. Thus, results may have been affected by these changes.

15. Select the name and price of the cheapest product.

Query:

```
SELECT Name, Price AS Cheapest  
FROM Products  
WHERE Price = (  
      SELECT MIN(price)  
      FROM Products)
```

Result:

Results Messages		
	Name	Cheapest
1	Floppy disk	5

16. Select the name of each manufacturer along with the name and price of its most expensive product.

Query:

```
SELECT Manufacturers.Name, CAST(Products.Name AS VARCHAR(50)) AS ProductName,
Products.Price
FROM Products
INNER JOIN Manufacturers
ON Products.Manufacturer = Manufacturers.Code,
(SELECT MAX(Price) AS MostExpensiveProduct, Products.Manufacturer
FROM Products
GROUP BY Products.Manufacturer) Pricey
WHERE Products.Manufacturer = Pricey.Manufacturer
AND Products.Price = Pricey.MostExpensiveProduct;
```

Result:

Results Messages			
	Name	ProductName	Price
1	Winchester	Memory	120
2	Fujitsu	Hard drive	240
3	Iomega	ZIP drive	150
4	Hewlett-Packard	Laser Printer	270
5	Creative Labs	DVD burner	180
6	Creative Labs	DVD drive	180
7	Sony	Monitor	240

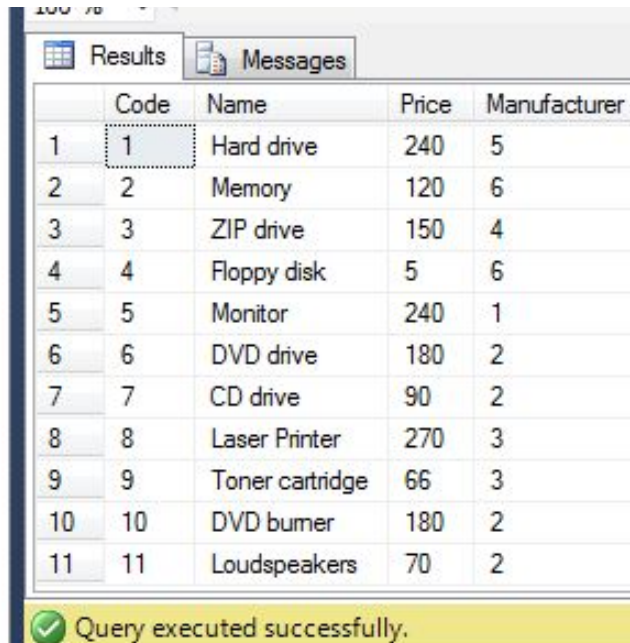
*NOTE: Creative Labs had a tie for most expensive product.*

17. Add a new product: Loudspeakers, \$70, manufacturer 2.

Query:

```
INSERT INTO Products  
VALUES(11,'Loudspeakers',70,2);
```

Result: (1 row(s) affected) - I have included the SELECT \* FROM Products result below:



	Code	Name	Price	Manufacturer
1	1	Hard drive	240	5
2	2	Memory	120	6
3	3	ZIP drive	150	4
4	4	Floppy disk	5	6
5	5	Monitor	240	1
6	6	DVD drive	180	2
7	7	CD drive	90	2
8	8	Laser Printer	270	3
9	9	Toner cartridge	66	3
10	10	DVD burner	180	2
11	11	Loudspeakers	70	2

Query executed successfully.

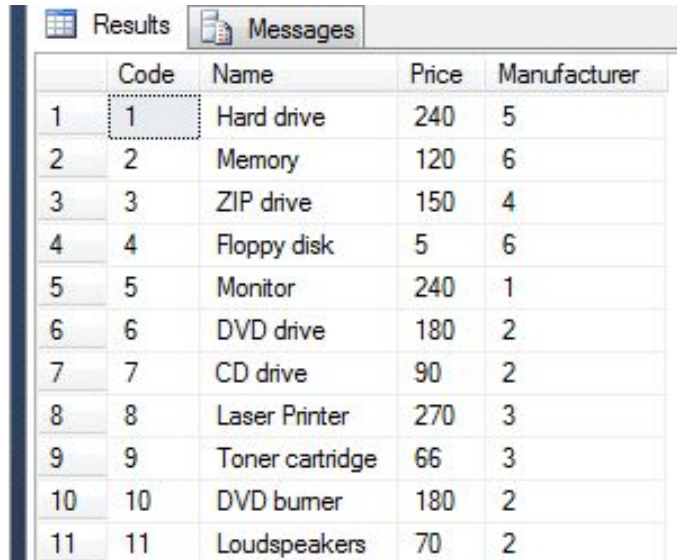
*\*Additionally, I re-ran most of the queries for screenshots in this document after I completed this step. Thus, the additional item (Loudspeakers) may appear in screenshots of queries prior to this step.*

18. Update the name of product 8 to "Laser Printer".

Query:

```
UPDATE Products  
SET Name = 'Laser Printer'  
WHERE Code = 8;
```

Result: (1 row(s) affected) - I have included the SELECT \* FROM Products result below.



The screenshot shows a database interface with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with the following data:

	Code	Name	Price	Manufacturer
1	1	Hard drive	240	5
2	2	Memory	120	6
3	3	ZIP drive	150	4
4	4	Floppy disk	5	6
5	5	Monitor	240	1
6	6	DVD drive	180	2
7	7	CD drive	90	2
8	8	Laser Printer	270	3
9	9	Toner cartridge	66	3
10	10	DVD burner	180	2
11	11	Loudspeakers	70	2

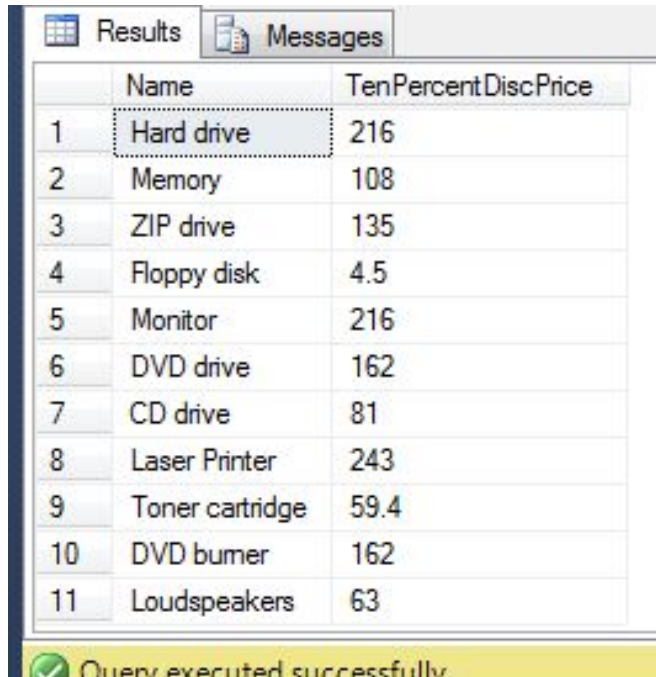
*\*Additionally, I re-ran most of the queries for screenshots in this document after I completed this step. Thus, the updated name may show in screenshots of queries prior to this step.*

19. Apply a 10% discount to all products.

Query:

```
SELECT Products.Name, Products.Price*0.9 AS TenPercentDiscPrice
FROM Products
```

Result:



The screenshot shows the 'Results' tab of a SQL query window. It displays a table with two columns: 'Name' and 'TenPercentDiscPrice'. The table contains 11 rows of data, representing various computer components and their discounted prices. A status bar at the bottom indicates 'Query executed successfully.'.

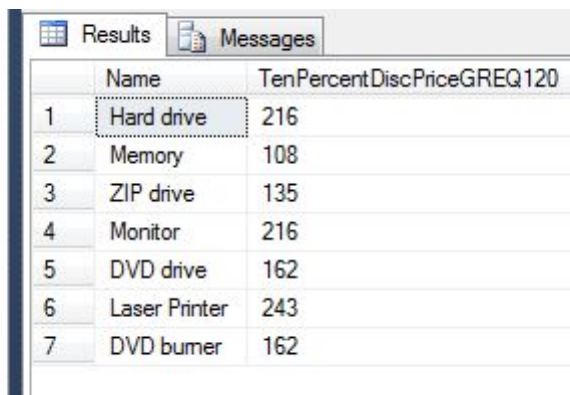
	Name	TenPercentDiscPrice
1	Hard drive	216
2	Memory	108
3	ZIP drive	135
4	Floppy disk	4.5
5	Monitor	216
6	DVD drive	162
7	CD drive	81
8	Laser Printer	243
9	Toner cartridge	59.4
10	DVD burner	162
11	Loudspeakers	63

20. Apply a 10% discount to all products with a price larger than or equal to \$120.

Query:

```
SELECT Products.Name, Products.Price*0.9 AS TenPercentDiscPriceGREQ120
FROM Products
WHERE Products.Price >= 120
```

Result:



The screenshot shows the 'Results' tab of a SQL query window. It displays a table with two columns: 'Name' and 'TenPercentDiscPriceGREQ120'. The table contains 7 rows of data, representing products whose original price was \$120 or more. The 'Name' column is highlighted with a dotted border.

	Name	TenPercentDiscPriceGREQ120
1	Hard drive	216
2	Memory	108
3	ZIP drive	135
4	Monitor	216
5	DVD drive	162
6	Laser Printer	243
7	DVD burner	162