

Comp 1630 - Relational Database & SQL

Term Project

INSTRUCTIONS

Follow the instructions given in part A to first create and then build the data base by running the SQL script provided. Using Microsoft SQL Server, create the SQL scripts for the tasks in each part of the project (B, C, and D) that are necessary to generate the required result sets, clearly identifying your answers.

Submit each part of the project in separate PDF files using your Student Id (A00123456_Part B.pdf) to the appropriate LearningHub drop box.

MARKING GUIDE

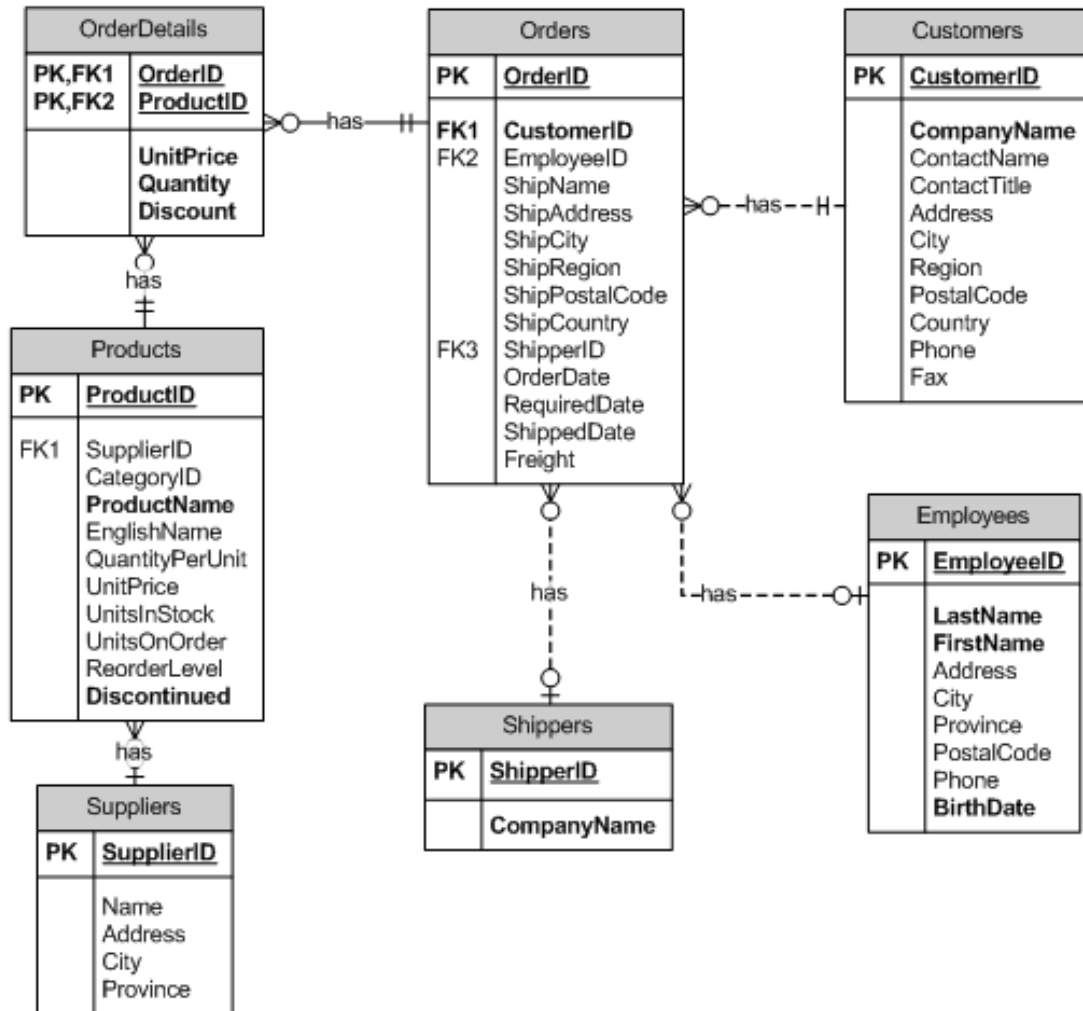
Although a result set may match the one provided, marks will be deducted for the following:

- | | |
|-------------------------------|---|
| 1. Missing code | (e.g. missing ORDER BY clause) |
| 2. Extra code | (e.g. unnecessary JOIN clause) |
| 3. Incorrect code | (e.g. CREATE PROCEDURE instead of CREATE TRIGGER) |
| 4. Inefficient code | (e.g. LEFT JOIN instead of INNER JOIN) |
| 5. Incorrect data type | (e.g. using INT instead of MONEY) |
| 6. Not following instructions | |
| 7. Non-working code | |

Comp 1630 - Relational Database & SQL

Term Project

ENTITY RELATIONSHIP DIAGRAM



PART A - Database and Tables

A1. Create a database called **A00123456_Project**.

A2. Run the script **ProjectData.sql** to create the tables listed below, and to populate the tables with data.

| | |
|--------------|-----------|
| Customers | 91 rows |
| Employees | 9 rows |
| Shippers | 3 rows |
| Suppliers | 15 rows |
| Products | 77 rows |
| Orders | 1078 rows |
| OrderDetails | 2820 rows |

Comp 1630 - Relational Database & SQL

Term Project

PART B - SQL Statements

- B1.** List the order details where the quantity is between 65 and 70. Display the order id and quantity from the OrderDetails table, the product id and reorder level from the Products table, and the supplier id from the Suppliers table. Order the result set by the order id. The query should produce the result set listed below.

| OrderID | Quantity | ProductID | ReorderLevel | SupplierID |
|---------|----------|-----------|--------------|------------|
| ----- | ----- | ----- | ----- | ----- |
| 10009 | 70 | 23 | 25 | 9 |
| 10018 | 70 | 11 | 30 | 5 |
| 10058 | 70 | 41 | 10 | 9 |
| ... | | | | |
| 11008 | 70 | 28 | 0 | 12 |
| 11030 | 70 | 5 | 0 | 2 |
| 11033 | 70 | 53 | 0 | 14 |

(49 row(s) affected)

- B2.** List the product id, product name, English name, and unit price from the Products table where the unit price is less than \$8.00. Order the result set by the product id. The query should produce the result set listed below.

| ProductID | ProductName | EnglishName | UnitPrice |
|-----------|----------------------|-------------------------------|-----------|
| ----- | ----- | ----- | ----- |
| 13 | Konbu | Kelp Seaweed | 6.00 |
| 24 | Guaraná Fantástica | Guaraná Fantástica Soft Drink | 4.50 |
| 33 | Geitost | Goat Cheese | 2.50 |
| 52 | Filo Mix | Mix for Greek Filo Dough | 7.00 |
| 54 | Tourtière | Pork Pie | 7.45 |
| 75 | Rhönbräu Klosterbier | Rhönbräu Beer | 7.75 |

(6 row(s) affected)

- B3.** List the customer id, company name, country, and phone from the Customers table where the country is equal to Canada or USA. Order the result set by the customer id. The query should produce the result set listed below.

| CustomerID | CompanyName | Country | Phone |
|------------|-----------------------------------|---------|----------------|
| ----- | ----- | ----- | ----- |
| BOTTM | Bottom-Dollar Markets | Canada | (604) 555-4729 |
| GREAL | Great Lakes Food Market | USA | (503) 555-7555 |
| HUNGC | Hungry Coyote Import Store | USA | (503) 555-6874 |
| | | | |
| THECR | The Cracker Box | USA | (406) 555-5834 |
| TRAIH | Trail's Head Gourmet Provisioners | USA | (206) 555-8257 |
| WHITC | White Clover Markets | USA | (206) 555-4112 |

(16 row(s) affected)

Comp 1630 - Relational Database & SQL

Term Project

- B4.** List the products where the **reorder level is equal to the units in stock**. Display the supplier id and supplier name from the Suppliers table, the product name, reorder level, and units in stock from the Products table. Order the result set by the supplier id. The query should produce the result set listed below.

| SupplierID | Name | ProductName | ReorderLevel | UnitsInStock |
|------------|------------|-------------------------|--------------|--------------|
| 2 | Supplier B | Chef Anton's Gumbo Mix | 0 | 0 |
| 7 | Supplier G | Alice Mutton | 0 | 0 |
| 12 | Supplier L | Thüringer Rostbratwurst | 0 | 0 |
| 14 | Supplier N | Perth Pasties | 0 | 0 |

(4 row(s) affected)

- B5.** List the orders where the **shipped date is greater than or equal to 1st of Jan 1994**, and **calculate the length in years from the shipped date to 1st of Jan 2009**. Display the order id, and the shipped date from the Orders table, the company name, and the contact name from the Customers table, and the calculated length in years for each order. Display the shipped date in the format MMM DD YYYY. Order the result set by order id and the calculated years. The query should produce the result set listed below.

| OrderID | CompanyName | ContactName | ShippedDate | ElapsedYears |
|---------|--------------------------|----------------------|-------------|--------------|
| 10840 | LINO-Delicateses | Felipe Izquierdo | Jan 10 1994 | 15 |
| 10847 | Save-a-lot Markets | Jose Pavarotti | Jan 4 1994 | 15 |
| 10856 | Antonio Moreno Taquería | Antonio Moreno | Jan 4 1994 | 15 |
| ... | | | | |
| 11066 | White Clover Markets | Karl Jablonski | Mar 28 1994 | 15 |
| 11067 | Drachenblut Delikatessen | Sven Ottlieb | Mar 30 1994 | 15 |
| 11069 | Tortuga Restaurante | Miguel Angel Paolino | Mar 30 1994 | 15 |

(198 row(s) affected)

- B6.** List all the orders where the **order date is between 1st of Jan and 30th of Mar 1992**, and the **cost of the order is greater than or equal to \$1500.00**. Display the order id, order date, and a new shipped date calculated by adding 10 days to the shipped date from the Orders table, the product name from the Products table, the company name from the Customer table, and the cost of the order. Format the date order date and the shipped date as MON DD YYYY. Use the formula **(OrderDetails.Quantity * Products.UnitPrice)** to calculate the cost of the order. Order the result set by order id. The query should produce the result set listed below.

| OrderID | ProductName | CompanyName | OrderDate | NewShippedDate | OrderCost |
|---------|--------------------|------------------------|-------------|----------------|-----------|
| 10141 | Schoggi Schokolade | Frankenversand | Jan 2 1992 | Jan 19 1992 | 2195.00 |
| 10141 | Camembert Pierrot | Frankenversand | Jan 2 1992 | Jan 19 1992 | 1700.00 |
| 10163 | Camembert Pierrot | Frankenversand | Feb 7 1992 | Feb 21 1992 | 1632.00 |
| | | | | | |
| 10195 | Côte de Blaye | Que Delícia | Mar 23 1992 | Apr 6 1992 | 5533.50 |
| 10196 | Côte de Blaye | LILA-Supermercado | Mar 24 1992 | May 2 1992 | 7905.00 |
| 10198 | Côte de Blaye | Océano Atlántico Ltda. | Mar 26 1992 | Apr 9 1992 | 1581.00 |

(11 row(s) affected)

Comp 1630 - Relational Database & SQL

Term Project

- B7.** List all the orders with a **shipping city of Vancouver**. Display the order from the Orders table, and the unit price and quantity from the OrderDetails table. Order the result set by the order id. The query should produce the result set listed below.

| OrderID | UnitPrice | Quantity |
|---------|-----------|----------|
| 10495 | 7.20 | 10 |
| 10495 | 7.70 | 20 |
| 10495 | 10.40 | 5 |
| 10620 | 4.50 | 5 |
| 10620 | 7.00 | 5 |
| 10810 | 6.00 | 7 |
| 10810 | 14.00 | 5 |
| 10810 | 15.00 | 5 |

(8 row(s) affected)

- B8.** List all the orders that have not been shipped (**shipped date is null**). Display the customer id, company name and fax number from the Customers table, and the order id and order date from the Orders table. Order the result set by the customer id and order date. The query should produce the result set listed below.

| CustomerID | CompanyName | Fax | OrderID | OrderDate |
|------------|-------------------------|----------------|---------|---------------------|
| BLAUS | Blauer See Delikatessen | 0621-08924 | 11058 | 1994-03-23 00:00:00 |
| BONAP | Bon app' | 91.24.45.41 | 11076 | 1994-03-30 00:00:00 |
| BOTTM | Bottom-Dollar Markets | (604) 555-3745 | 11045 | 1994-03-17 00:00:00 |
| | | | | |
| RICAR | Ricardo Adocicados | NULL | 11059 | 1994-03-23 00:00:00 |
| RICSU | Richter Supermarkt | NULL | 11075 | 1994-03-30 00:00:00 |
| SIMOB | Simons bistro | 31 13 35 57 | 11074 | 1994-03-30 00:00:00 |

(21 row(s) affected)

- B9.** List the products which contain **choc or tofu in their name**. Display the product id, product name, quantity per unit and unit price from the Products table. Order the result set by product id. The query should produce the result set listed below.

| ProductID | ProductName | QuantityPerUnit | UnitPrice |
|-----------|----------------------------|----------------------|-----------|
| 14 | Tofu | 40 - 100 g pkgs. | 23.25 |
| 19 | Teatime Chocolate Biscuits | 10 boxes x 12 pieces | 9.20 |
| 48 | Chocolade | 10 pkgs. | 12.75 |
| 74 | Longlife Tofu | 5 kg pkg. | 10.00 |

(4 row(s) affected)

Comp 1630 - Relational Database & SQL

Term Project

B10. List the number of products and their **names beginning with each letter of the alphabet**. Only display the letter and count if there are at least three product names begin with the letter. The query should produce the result set listed below.

| ProductName | Total |
|-------------|-------|
| ----- | ----- |
| C | 9 |
| G | 11 |
| I | 3 |
| L | 5 |
| M | 5 |
| N | 3 |
| P | 3 |
| R | 6 |
| S | 9 |
| T | 6 |

(10 row(s) affected)

Comp 1630 - Relational Database & SQL

Term Project

PART C - INSERT, UPDATE, DELETE and VIEWS

- C1.** Create a view called **vw_supplier_items** listing the distinct suppliers and the items they have shipped. Display the supplier id and name from the Suppliers table, and the product id and product name from the Products table. Use the following query to test your view to produce the result set listed below.

```
SELECT *
FROM vw_supplier_items
ORDER BY Name, ProductID
```

| SupplierID | Name | ProductID | ProductName |
|------------|------------|-----------|---------------------------|
| 5 | Supplier | 11 | Queso Cabrales |
| 5 | Supplier | 12 | Queso Manchego La Pastora |
| 1 | Supplier A | 1 | Chai |
| ... | | | |
| 15 | Supplier O | 56 | Gnocchi di nonna Alice |
| 15 | Supplier O | 69 | Gudbrandsdalsost |
| 15 | Supplier O | 71 | Fløtemysost |

(77 row(s) affected)

- C2.** Create a view called **vw_employee_info** to list all the employees in the Employee table. Display the employee id, last name, first name, and birth date. Format the name as first name followed by a space followed by the last name. Use the following query to test your view to produce the result set listed below.

```
SELECT *
FROM vw_employee_info
WHERE EmployeeID IN ( 3, 6, 9 )
```

| EmployeeID | Name | BirthDate |
|------------|-----------------|-------------------------|
| 3 | Janet Leverling | 1963-08-30 00:00:00.000 |
| 6 | Michael Suyama | 1963-07-02 00:00:00.000 |
| 9 | Anne Dodsworth | 1966-01-27 00:00:00.000 |

(3 row(s) affected)

- C3.** Using the UPDATE statement, change the **fax value to Unknown** for all rows in the Customers table where the current fax value is null (22 rows affected).

Comp 1630 - Relational Database & SQL

Term Project

- C4.** Create a view called **vw_order_cost** to list the cost of orders. Display the order id and order_date from the Orders table, the product id from the Products table, the company name from the Customers table, and the order cost. To calculate the cost of the orders, use the formula: (OrderDetails.Quantity * OrderDetails.UnitPrice).

Use the following query to test your view to produce the result set listed below.

```
SELECT *
FROM vw_order_cost
WHERE orderID BETWEEN 10100 AND 10200
ORDER BY ProductID
```

| OrderID | OrderDate | ProductID | CompanyName | OrderCost |
|---------|---------------------|-----------|-------------------------|-----------|
| 10101 | 1991-10-28 00:00:00 | 1 | Antonio Moreno Taquería | 96.00 |
| 10156 | 1992-01-28 00:00:00 | 1 | Richter Supermarkt | 300.00 |
| 10159 | 1992-01-31 00:00:00 | 1 | Maison Dewey | 480.00 |
| ... | | | | |
| 10170 | 1992-02-20 00:00:00 | 77 | Reggiani Caseifici | 216.00 |
| 10115 | 1991-11-20 00:00:00 | 77 | Océano Atlántico Ltda. | 45.00 |
| 10117 | 1991-11-22 00:00:00 | 77 | Tradição Hipermercados | 254.80 |

(257 row(s) affected)

- C5.** Using the INSERT statement, add a row to the Suppliers table with a **supplier id of 16 and a name of 'Supplier P'**.

- C6.** Using the UPDATE statement, increase the unit price in the Products table by 15% for rows with a current **unit price less than \$5.00** (2 rows affected).

- C7.** Create a view called **vw_orders** to list orders. Display the order id and shipped date from the Orders table, and the customer id, company name, city, and country from the Customers table. Use the following query to test your view to produce the result set listed below.

```
SELECT *
FROM vw_orders
WHERE ShippedDate BETWEEN '1993-01-01' AND '1993-01-31'
ORDER BY CompanyName, Country
```

| OrderID | CustomerID | CompanyName | City | Country | ShippedDate |
|---------|------------|----------------------|--------|---------|---------------------|
| 10453 | AROUT | Around the Horn | London | UK | 1993-01-20 00:00:00 |
| 10444 | BERGS | Berglunds snabbköp | Luleå | Sweden | 1993-01-15 00:00:00 |
| 10445 | BERGS | Berglunds snabbköp | Luleå | Sweden | 1993-01-14 00:00:00 |
| ... | | | | | |
| 10459 | VICTE | Victuailles en stock | Lyon | France | 1993-01-22 00:00:00 |
| 10455 | WARTH | Wartian Herkku | Oulu | Finland | 1993-01-25 00:00:00 |
| 10437 | WARTH | Wartian Herkku | Oulu | Finland | 1993-01-06 00:00:00 |

(33 row(s) affected)

Comp 1630 - Relational Database & SQL

Term Project

PART D - Stored Procedures and Triggers

- D1.** Create a stored procedure called **sp_emp_info** to display the employee id, last name, first name, and phone number from the Employees table for a particular employee. The employee id will be an input parameter for the stored procedure. Use the following query to test your stored procedure to produce the result set listed below.

EXEC sp_emp_info 7

| EmployeeID | LastName | FirstName | Phone |
|------------|----------|-----------|------------|
| 7 | King | Robert | 6045555598 |

(1 row(s) affected)

- D2.** Create a stored procedure called **sp_orders_by_dates** displaying the orders shipped between particular dates. The start and end date will be input parameters for the stored procedure. Display the order id, customer id, and shipped date from the Orders table, the company name from the Customer table, and the shipper name from the Shippers table. Use the following query to test your stored procedure to produce the result set listed below.

EXEC sp_orders_by_dates '1991-01-01', '1991-12-31'

| OrderID | CustomerID | CompanyName | ShipperName | ShippedDate |
|---------|------------|------------------------------|------------------|---------------------|
| 10000 | FRANS | Franchi S.p.A. | Federal Shipping | 1991-05-15 00:00:00 |
| 10001 | MEREP | Mère Paillarde | Federal Shipping | 1991-05-23 00:00:00 |
| 10002 | FOLKO | Folk och fä HB | Federal Shipping | 1991-05-17 00:00:00 |
| ... | | | | |
| 10133 | REGGC | Reggiani Caseifici | United Package | 1991-12-23 00:00:00 |
| 10134 | WARTH | Wartian Herkku | Speedy Express | 1991-12-27 00:00:00 |
| 10136 | HUNGO | Hungry Owl All-Night Grocers | Speedy Express | 1991-12-30 00:00:00 |

(134 row(s) affected)

- D3.** Create a stored procedure called **sp_products** listing a specified product ordered during a specified month and year. The product name, month, and year will be input parameters for the stored procedure. Display the product name, unit price, and units in stock from the Products table, and the supplier name from the Suppliers table. Use the following query to test your stored procedure to produce the result set listed below.

EXEC sp_products '%tofu%', 'December', 1992

| ProductName | UnitPrice | UnitsInStock | Name |
|---------------|-----------|--------------|------------|
| Tofu | 23.25 | 35 | Supplier F |
| Tofu | 23.25 | 35 | Supplier F |
| Longlife Tofu | 10.00 | 4 | Supplier D |
| Tofu | 23.25 | 35 | Supplier F |

(4 row(s) affected)

Comp 1630 - Relational Database & SQL

Term Project

- D4.** Create a stored procedure called **sp_unit_prices** listing the products where the unit price is between particular values. The two unit prices will be input parameters for the stored procedure. Display the product id, product name, English name, and unit price from the Products table. Use the following query to test your stored procedure to produce the result set listed below.

EXEC sp_unit_prices 5.50, 8.00

| ProductID | ProductName | EnglishName | UnitPrice |
|-----------|----------------------|--------------------------|-----------|
| 13 | Konbu | Kelp Seaweed | 6.00 |
| 52 | Filo Mix | Mix for Greek Filo Dough | 7.00 |
| 54 | Tourtière | Pork Pie | 7.45 |
| 75 | Rhönbräu Klosterbier | Rhönbräu Beer | 7.75 |

(4 row(s) affected)

- D5.** Create a stored procedure called **sp_customer_city** displaying the customers living in a particular city. The city will be an input parameter for the stored procedure. Display the customer id, company name, address, city and phone from the Customers table. Use the following query to test your stored procedure to produce the result set listed below.

EXEC sp_customer_city 'Paris'

| CustomerID | CompanyName | Address | City | Phone |
|------------|----------------------|-------------------------|-------|-----------------|
| PARIS | Paris spécialités | 265, boulevard Charonne | Paris | (1) 42.34.22.66 |
| SPECD | Spécialités du monde | 25, rue Lauriston | Paris | (1) 47.55.60.10 |

(2 row(s) affected)

- D6.** Create a stored procedure called **sp_reorder_qty** to show when the reorder level subtracted from the units in stock is less than a specified value. The unit value will be an input parameter for the stored procedure. Display the product id, product name, units in stock, and reorder level from the Products table, and the supplier name from the Suppliers table. Use the following query to test your stored procedure to produce the result set listed below.

EXEC sp_reorder_qty 9

| ProductID | ProductName | Name | UnitsInStock | ReorderLevel |
|-----------|------------------------|------------|--------------|--------------|
| 2 | Chang | Supplier A | 17 | 25 |
| 3 | Aniseed Syrup | Supplier A | 13 | 25 |
| 5 | Chef Anton's Gumbo Mix | Supplier B | 0 | 0 |
| ... | | | | |
| 68 | Scottish Longbreads | Supplier H | 6 | 15 |
| 70 | Outback Lager | Supplier G | 15 | 30 |
| 74 | Longlife Tofu | Supplier D | 4 | 5 |

(26 row(s) affected)

Comp 1630 - Relational Database & SQL

Term Project

- D7.** Create a stored procedure called `sp_shipping_date` where the shipped date is equal to the order date plus 10 days. The shipped date will be an input parameter for the stored procedure. Display the order id, order date and shipped date from the Orders table, the company name from the Customers table, and the company name from the Shippers table. Use the following query to test your stored procedure to produce the result set listed below.

EXEC sp_shipping_date '1993-11-29'

| OrderID | CustomerName | ShipperName | OrderDate | ShippedDate |
|---------|--------------------|------------------|---------------------|---------------------|
| 10798 | Island Trading | Speedy Express | 1993-11-19 00:00:00 | 1993-11-29 00:00:00 |
| 10799 | Königlich Essen | Federal Shipping | 1993-11-19 00:00:00 | 1993-11-29 00:00:00 |
| 10800 | Seven Seas Imports | Federal Shipping | 1993-11-19 00:00:00 | 1993-11-29 00:00:00 |

(3 row(s) affected)

- D8.** Create a stored procedure called `sp_del_inactive_cust` to delete customers that have no orders. Use the following query to test your procedure. The stored procedure should delete 1 row.

EXEC sp_del_inactive_cust

- D9.** Create an UPDATE trigger called `tr_check_qty` on the OrderDetails table to prevent the updating of orders of products that have units-in-stock less than the quantity ordered. Use the following query to test your trigger.

```
UPDATE OrderDetails
SET Quantity = 40
WHERE OrderID = 10044
AND ProductID = 77
```

- D10.** Create an INSTEAD OF INSERT trigger called `tr_insert_shippers` on the Shippers table preventing inserting a row with a company name which already exists. Use the following query to test your trigger.

```
INSERT Shippers
VALUES ( 4, 'Federal Shipping' )
```