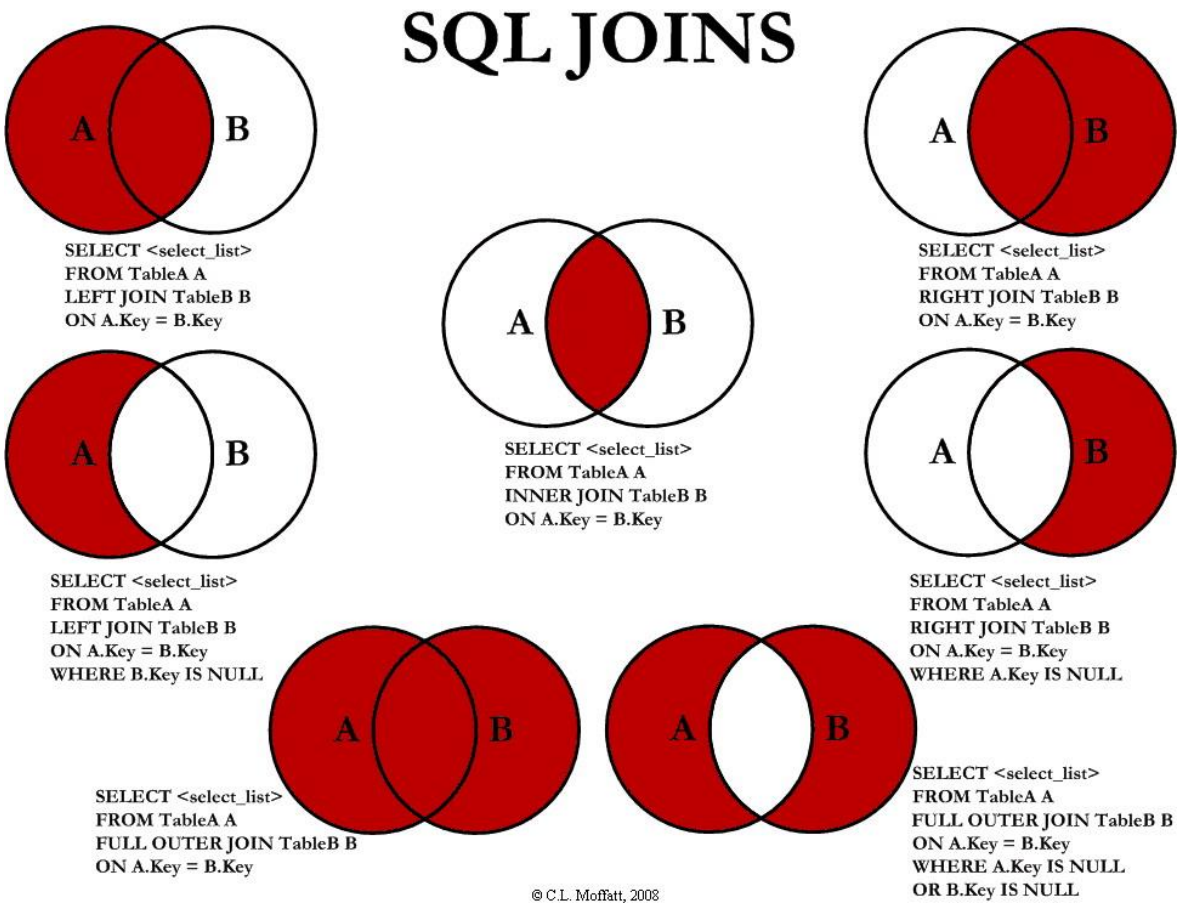
LAB 07

MYSQL JOIN

❖ **Main contents:** *In previous exercises, queries were executed on a data table. Not surprisingly, many queries require information from many different data tables. For example, if we want to show customer information of orders, we need to combine information from two data tables, customers and orders. Combining data tables to create an inference table is called a join. In this article, we will become familiar with the join operation to query data from multiple tables: INNER JOIN, LEFT JOIN, SELF JOIN*



1. INNER JOIN

`INNER JOIN`, also known as internal join, is an optional part of the `SELECT` statement. It appears immediately after the `FROM` clause. Before using `INNER JOIN`, the following criteria must be clearly defined:

- First, we need to identify the tables that you want to link to the main table. The main table appears in the `FROM` clause. The table we want to join the main table must appear after the keyword `INNER JOIN`. Theoretically, it is possible to join a table with an unlimited number of other tables, however, for better performance, it is advisable to limit

the number of tables joining joins based on join conditions and volume of data. Data in the tables.

- Second, the connection conditions need to be determined. The join condition appears after the keyword ON. The join condition is the rule for finding matching records in tables and joining them together.

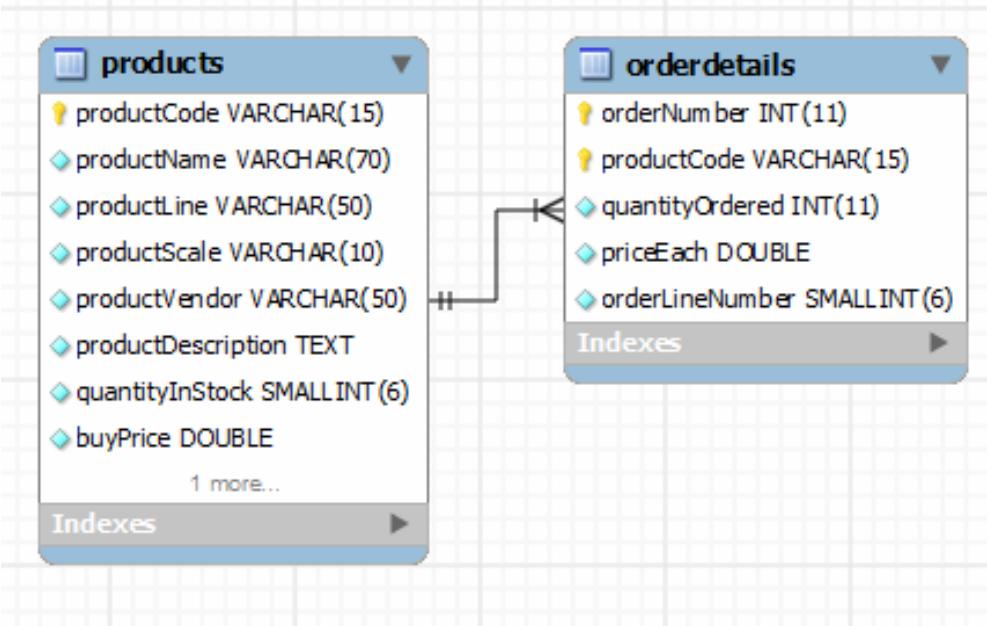
The `INNER JOIN` syntax:

```
SELECT column_list
FROM table1
INNER JOIN table2 ON join_condition1
INNER JOIN table3 ON join_condition2
...
WHERE conditions;
```

For example: if connecting tables A and B, `INNER JOIN` compares each record of table A with each record of table B to find all pairs of records that meet the join condition. When the join condition is met, the column values for each matching record pair of table A and table B are combined into one record in the return result.

Limit names of columns when using `INNER JOIN`: If connecting multiple tables with columns with similar names, must specify the name of the table that contains the data column to retrieve to avoid error column unclear. Suppose if the tables *tbl_A* and *tbl_B* have similar columns M. In the SELECT statement with `INNER JOIN`, the column M must be referenced using the syntax like *tbl_A.M* .

Example: Consider the *Products* and *OrderDetails* tables. The products table is the overall data sheet that stores all products. Whenever a product is sold, it is stored in the *OrderDetails* table along with other information. The link between these tables is the *productCode* column.



Example: If we want to know which products have been sold, we can use `INNER JOIN` as follows:

```
SELECT products.productCode, products.productName,
orderDetails.orderNumber
FROM products
INNER JOIN orderDetails on products.productCode =
orderDetails.productCode;
```

productCode	productName	orderNumber
S18_1749	1917 Grand Touring Sedan	10100
S18_2248	1911 Ford Town Car	10100
S18_4409	1932 Alfa Romeo 8C2300 Spider Sport	10100
S24_3969	1936 Mercedes Benz 500k Roadster	10100
S18_2325	1932 Model A Ford J-Coupe	10101
S18_2795	1928 Mercedes-Benz SSK	10101
S24_1937	1939 Chevrolet Deluxe Coupe	10101
S24_2022	1938 Cadillac V-16 Presidential Limousine	10101
S18_1342	1937 Lincoln Berline	10102
S18_1367	1936 Mercedes-Benz 500K Special Roadster	10102
S10_1949	1952 Alpine Renault 1300	10103
S10_4962	1962 LanciaA Delta 16V	10103
S12_1666	1958 Setra Bus	10103
S18_1097	1940 Ford Pickup Truck	10103

`INNER JOIN` compares each row in the products and *OrderDetails* tables to find a pair of records that have the same *ProductCode*. If a pair of records has the same product code, then the product name and order number will also be combined into a row to return the result.

Alias: you can create the alias of the *tbl_A* table as A and refer to column M as *A.M*, so there is no need to retype the table name again. The example above can be rewritten as follows:

```
SELECT p.productCode, p.productName, o.orderNumber
FROM products p
INNER JOIN orderDetails o on p.productCode = o.productCode;
```

Note: In addition, to concatenate using the `INNER JOIN ... ON` clause, it is possible to join in two tables by including the join condition to the `WHERE` clause. The example above can be rewritten as follows:

```
SELECT p.productCode, p.productName, o.orderNumber
FROM products p, orderDetails o
WHERE p.productCode = o.productCode;
```

We will consider some other examples using the following join:
Example: *Employees* table is a table that holds information about employees of the company; The *Customers* table is a table that stores customer information, including information relating to customer service personnel codes. So the link between these two tables is done through the *employeeNumber* column of the *Employees* table and the *salesRepemployeeNumber* column of the *Customers* table.
For information about the customer and the name of the customer service officer, we can execute the query using `INNER JOIN` as follows:

```
SELECT customerName, firstname as EmployeeName
FROM customers c join employees e
on c.salesrepemployeenumber = e.employeenumber;
```

Result:

customerName	EmployeeName
Atelier graphique	Gerard
Signal Gift Stores	Leslie
Australian Collectors, Co.	Andy
La Rochelle Gifts	Gerard
Baane Mini Imports	Barry
Mini Gifts Distributors Ltd.	Leslie
Blauer See Auto, Co.	Barry
Mini Wheels Co.	Leslie
Land of Toys Inc.	George
Euro+ Shopping Channel	Gerard
Volvo Model Replicas, Co	Barry
Danish Wholesale Imports	Pamela
Saveley & Henriot, Co.	Loui
Dragon Souveniers, Ltd.	Mami
Muscle Machine Inc	Foon Yue
Diecast Classics Inc.	Steve
Technics Stores Inc.	Leslie

Example: Get the information about product lines and the total number of products in that product line.

```
SELECT pl.productLine, pl.textDescription, sum(quantityInStock)
FROM productlines pl JOIN products p ON pl.productLine
=p.productLine
GROUP by pl.productLine;
```

Result:

productLine	textDescription	sum(quantityInStock)
Classic Cars	Attention car enthusiasts: Ma...	219183
Motorcycles	Our motorcycles are state of t...	69401
Planes	Unique, diecast airplane and ...	62287
Ships	The perfect holiday or anniver...	26833
Trains	Model trains are a rewarding ...	16696
Trucks and Buses	The Truck and Bus models ar...	35851
Vintage Cars	Our Vintage Car models realist...	124880

Example: Get the information about the products and the total value ordered for the product

```
SELECT p.productCode,
            p.productName,
            SUM(priceEach * quantityOrdered) total
FROM orderdetails o
INNER JOIN products p ON o.productCode = p.productCode
GROUP by productCode
ORDER BY total;
```

Result:

productCode	productName	total
S24_1937	1939 Chevrolet Deluxe Coupe	28052.94
S24_3969	1936 Mercedes Benz 500k Roadster	29763.39
S24_2972	1982 Lamborghini Diablo	30972.869999999995
S24_2840	1958 Chevy Corvette Limited Edition	31627.960000000003
S32_2206	1982 Ducati 996 R	33268.76
S24_2022	1938 Cadillac V-16 Presidential Limousine	38449.090000000004
S50_1341	1930 Buick Marquette Phaeton	41599.24
S24_1628	1966 Shelby Cobra 427 S/C	42015.53999999999
S72_1253	Boeing X-32A JSF	42692.53
S18_4668	1939 Cadillac Limousine	44037.83999999999
S18_2248	1911 Ford Town Car	45306.770000000004
S18_1367	1936 Mercedes-Benz 500K Special Roadster	46078.29
S32_2509	1954 Greyhound Scenicruiser	46519.04999999998
S72_3212	Pont Yacht	47550.399999999994

Besides joining two data tables, we can join multiple data tables in the same `SELECT` statement.

Example: Get the names of customers and the total value of orders of those customers.

```
SELECT c.customerName, SUM(od.priceEach*od.quantityOrdered) as
total
FROM customers c
INNER JOIN orders o ON c.customerNumber = o.customerNumber
INNER JOIN orderdetails od ON o.orderNumber = od.orderNumber
GROUP BY c.customerName;
```

As in the example above, the information that needs to be combined from the three data tables is *customers*, *orders* and *orderdetails*.

customerName	total
Alpha Cognac	60483.359999999986
Amica Models & Co.	82223.23
Anna's Decorations, Ltd	137034.21999999994
Atelier graphique	22314.36
Australian Collectables, Ltd	55866.02
Australian Collectors, Co.	180585.07
Australian Gift Network, Co	55190.16
Auto Associés & Cie.	58876.409999999996
Auto Canal+ Petit	86436.96999999999
Auto-Moto Classics Inc.	21554.260000000002
AV Stores, Co.	148410.09000000003
Baane Mini Imports	104224.78999999996
Bavarian Collectables Imports, Co.	31310.089999999997
Blauer See Auto, Co.	75937.76
Boards & Toys Co.	7918.6
CAF Imports	46751.14000000001
Cambridge Collectables Co.	32198.69

Example: Get orders, customer names and total value of that order.

```
SELECT o.orderNumber,c.customerName,
SUM(od.priceEach*od.quantityOrdered) as total
FROM customers c
INNER JOIN orders o ON c.customerNumber = o.customerNumber
INNER JOIN orderdetails od ON O.orderNumber = od.orderNumber
GROUP BY o.orderNumber;
```

orderNumber	customerName	total
10100	Online Diecast Creations Co.	10223.829999999998
10101	Blauer See Auto, Co.	10549.01
10102	Vitachrome Inc.	5494.78
10103	Baane Mini Imports	50218.950000000004
10104	Euro+ Shopping Channel	40206.2
10105	Danish Wholesale Imports	53959.21
10106	Rovelli Gifts	52151.810000000005
10107	Land of Toys Inc.	22292.620000000003
10108	Cruz & Sons Co.	51001.219999999994
10109	Motor Mint Distributors Inc.	25833.14
10110	AV Stores, Co.	48425.69
10111	Mini Wheels Co.	16537.850000000002
10112	Volvo Model Replicas, Co	7674.9400000000005
10113	Mini Gifts Distributors Ltd.	11044.300000000001
10114	La Corne D'abondance, Co.	33383.14000000001
10115	Classic Legends Inc.	21665.980000000003
10116	Royale Belge	1627.56

2. LEFT JOIN

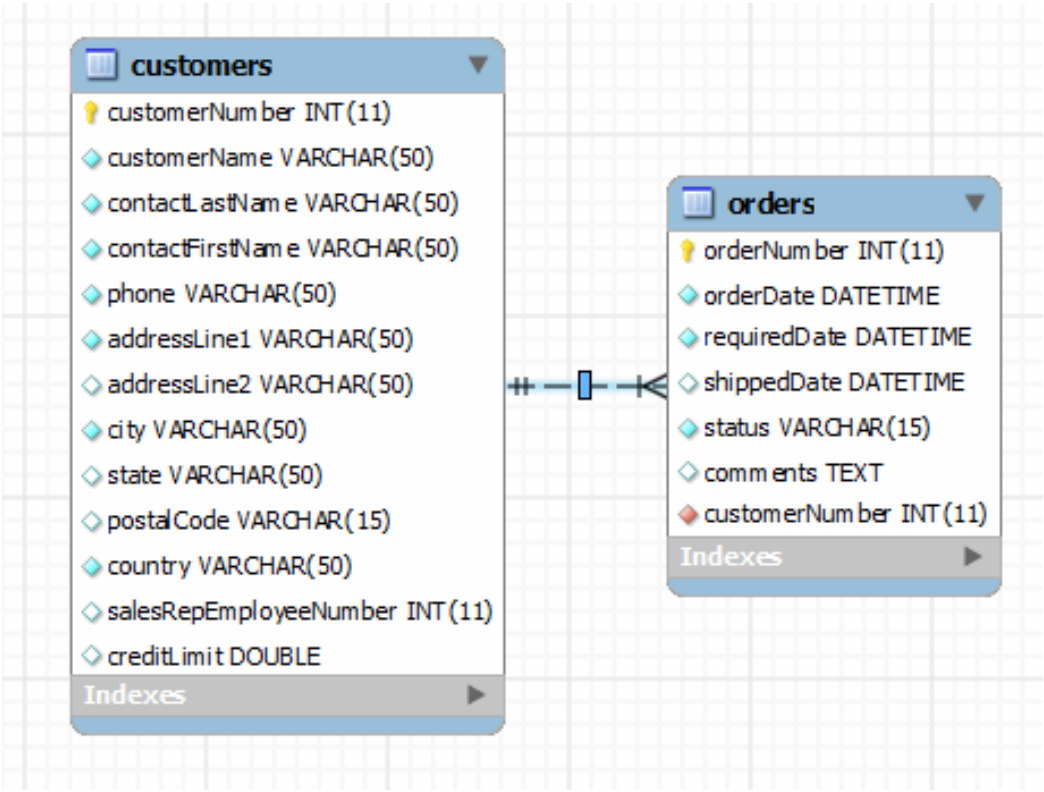
`LEFT JOIN` is also an option of the `SELECT` statement that allows additional data to be retrieved from other tables. `LEFT JOIN` includes the keywords `LEFT JOIN`, followed by the second table that wants to join. The next element is the keyword ON and followed by join conditions.

`LEFT JOIN` clause will be executed as follows: when a row from the left table matches a row from the right table based on join conditions, the contents of that row will be selected as a row in the output. When a row in the left table finds no matching rows in the join table, it still appears in the output, but combined with a "dummy" row from the right table with NULL values for all column.

In summary, `LEFT JOIN` allows selecting all rows from the left table even if there are no records that match it in the right table.

Example: Using `LEFT JOIN`

Let's look at the customers and orders tables. If you want to know a customer with their specific invoice and invoice status, you can use MySQL `LEFT JOIN` as follows:




```
SELECT c.customerNumber, customerName,orderNUmber, o.status
FROM customers c
LEFT JOIN orders o ON c.customerNumber = o.customerNumber;
```

customerNumber	customerName	orderNUmber	status
124	Mini Gifts Distributors L...	10371	Shipped
124	Mini Gifts Distributors L...	10382	Shipped
124	Mini Gifts Distributors L...	10385	Shipped
124	Mini Gifts Distributors L...	10390	Shipped
124	Mini Gifts Distributors L...	10396	Shipped
124	Mini Gifts Distributors L...	10421	In Process
125	Havel & Zbyszek Co	NULL	NULL
128	Blauer See Auto, Co.	10101	Shipped
128	Blauer See Auto, Co.	10230	Shipped
128	Blauer See Auto, Co.	10300	Shipped
128	Blauer See Auto, Co.	10323	Shipped

In the above result table, it can be seen that all the listed customers. However, there are records with customer information but all order information is NULL. This means that these customers do not have any orders stored in our database.

`LEFT JOIN` is especially useful when looking for records in the left table that do not match any records in the right table. This can be done by adding a `WHERE` clause to select rows with only

NULL values in a column in the right table. So, to find all customers who don't have any orders in our database, we can use `LEFT JOIN` as follows:

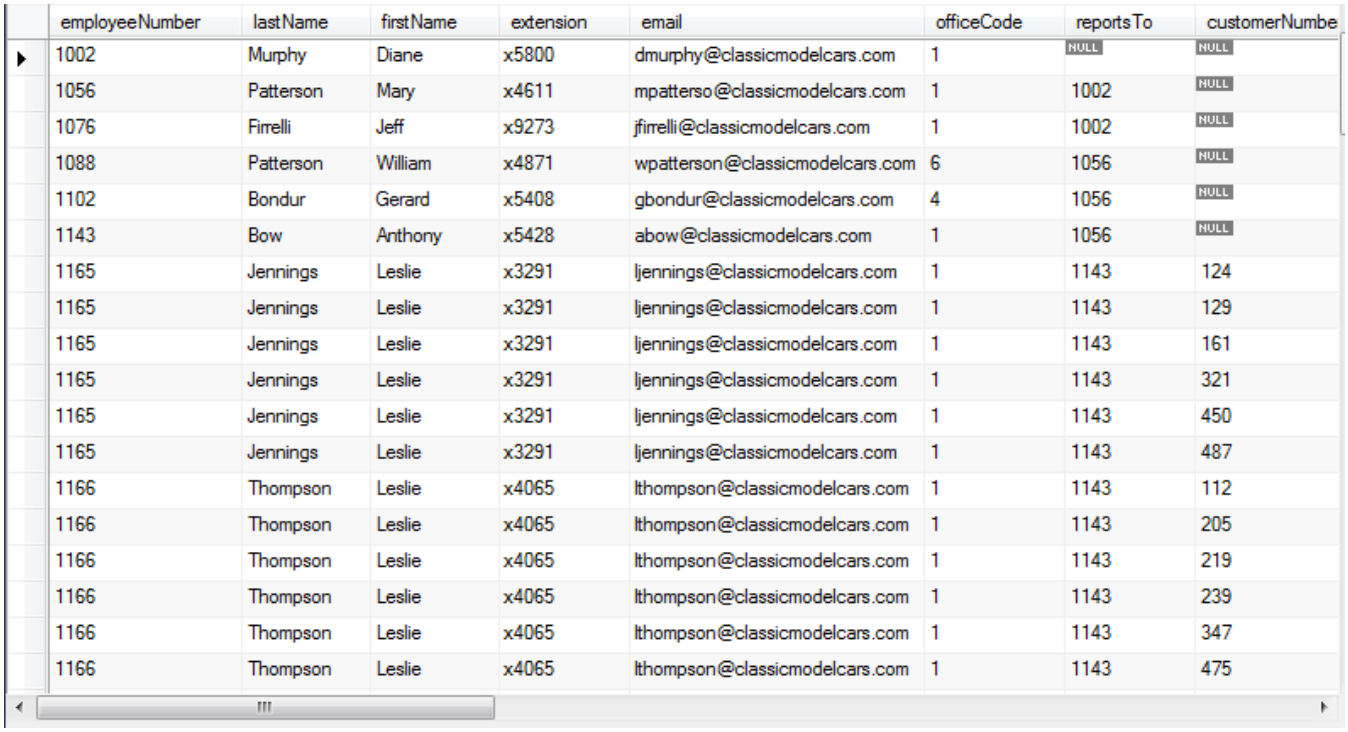
```
SELECT c.customerNumber, customerName,orderNUmber, o.status
FROM customers c
LEFT JOIN orders o ON c.customerNumber = o.customerNumber
WHERE orderNumber is NULL;
```

Result:

customerNumber	customerName	orderNUmber	status
125	Havel & Zbyszek Co	NULL	NULL
168	American Souvenirs Inc	NULL	NULL
169	Porto Imports Co.	NULL	NULL
206	Asian Shopping Network, Co	NULL	NULL
223	Natürlich Autos	NULL	NULL
237	ANG Resellers	NULL	NULL
247	Messner Shopping Network	NULL	NULL
273	Franken Gifts, Co	NULL	NULL
293	BG&E Collectables	NULL	NULL
303	Schuyler Imports	NULL	NULL

As such, the query only returns customers without any orders thanks to NULL values. Similarly, in order to find out the employees who are not in charge of customer service, initially perform the following query:

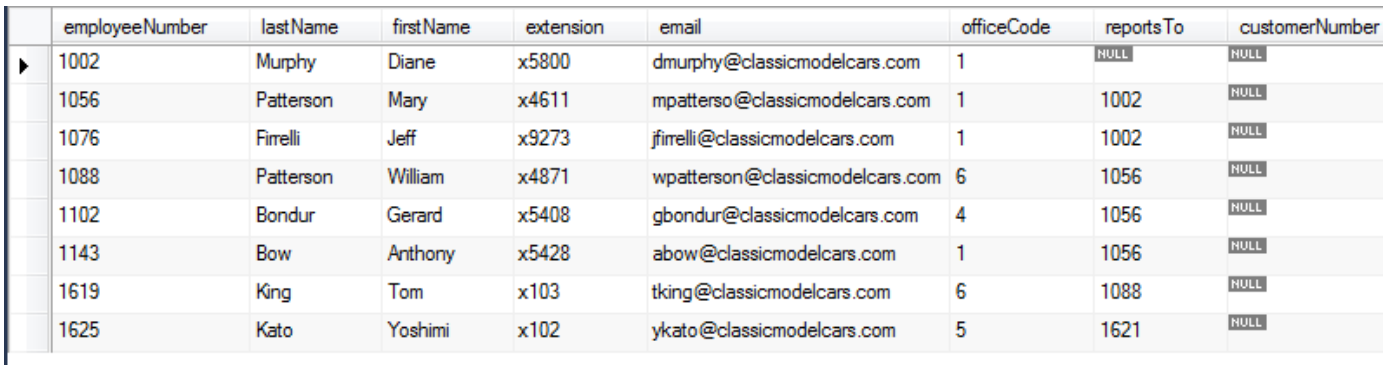
```
SELECT * FROM employees e
LEFT JOIN customers c
ON e.employeenumber=c.salesrepemployeenumber;
```



employeeNumber	lastName	firstName	extension	email	officeCode	reportsTo	customerNumbe
1002	Murphy	Diane	x5800	dmurphy@classicmodelcars.com	1	NULL	NULL
1056	Patterson	Mary	x4611	mpatterso@classicmodelcars.com	1	1002	NULL
1076	Firrelli	Jeff	x9273	jfirrelli@classicmodelcars.com	1	1002	NULL
1088	Patterson	William	x4871	wpatterson@classicmodelcars.com	6	1056	NULL
1102	Bondur	Gerard	x5408	gbondur@classicmodelcars.com	4	1056	NULL
1143	Bow	Anthony	x5428	abow@classicmodelcars.com	1	1056	NULL
1165	Jennings	Leslie	x3291	ljennings@classicmodelcars.com	1	1143	124
1165	Jennings	Leslie	x3291	ljennings@classicmodelcars.com	1	1143	129
1165	Jennings	Leslie	x3291	ljennings@classicmodelcars.com	1	1143	161
1165	Jennings	Leslie	x3291	ljennings@classicmodelcars.com	1	1143	321
1165	Jennings	Leslie	x3291	ljennings@classicmodelcars.com	1	1143	450
1165	Jennings	Leslie	x3291	ljennings@classicmodelcars.com	1	1143	487
1166	Thompson	Leslie	x4065	lthompson@classicmodelcars.com	1	1143	112
1166	Thompson	Leslie	x4065	lthompson@classicmodelcars.com	1	1143	205
1166	Thompson	Leslie	x4065	lthompson@classicmodelcars.com	1	1143	219
1166	Thompson	Leslie	x4065	lthompson@classicmodelcars.com	1	1143	239
1166	Thompson	Leslie	x4065	lthompson@classicmodelcars.com	1	1143	347
1166	Thompson	Leslie	x4065	lthompson@classicmodelcars.com	1	1143	475

Then filter out the records of NULL values in the *customerNumber* column, which is the result of the query.

```
SELECT * FROM employees e
LEFT JOIN customers c
ON e.employeenumber=c.salesrepemployeenumber
WHERE customerNumber is NULL;
```



employeeNumber	lastName	firstName	extension	email	officeCode	reportsTo	customerNumber
1002	Murphy	Diane	x5800	dmurphy@classicmodelcars.com	1	NULL	NULL
1056	Patterson	Mary	x4611	mpatterso@classicmodelcars.com	1	1002	NULL
1076	Firrelli	Jeff	x9273	jfirrelli@classicmodelcars.com	1	1002	NULL
1088	Patterson	William	x4871	wpatterson@classicmodelcars.com	6	1056	NULL
1102	Bondur	Gerard	x5408	gbondur@classicmodelcars.com	4	1056	NULL
1143	Bow	Anthony	x5428	abow@classicmodelcars.com	1	1056	NULL
1619	King	Tom	x103	tking@classicmodelcars.com	6	1088	NULL
1625	Kato	Yoshimi	x102	ykato@classicmodelcars.com	5	1621	NULL

3. SELF JOIN

A join is a type of join in which a table is connected to itself, namely when a table has a foreign key that references its primary key.

Example: The *employees* table has a foreign key, *reportsTo*, that references the primary key *employeeNumbe*r of the e*mployees* table itself.

It is necessary to use aliases for each copy of that table to avoid ambiguity

```
SELECT concat (e1.lastName ," ",e1.firstName) as fullname,
e1.email, concat (e2.lastName ," ",e2.firstName) as manager,
e2.email
FROM employees e1, employees e2
WHERE e1.reportsTo = e2.employeeNumber;
```

Result:

fullname	email	manager	email
Patterson Mary	mpatterso@classicmodelcars.com	Murphy Diane	dmurphy@classicmodelcars.com
Firrelli Jeff	jfirrelli@classicmodelcars.com	Murphy Diane	dmurphy@classicmodelcars.com
Patterson William	wpatterson@classicmodelcars.com	Patterson Mary	mpatterso@classicmodelcars.com
Bondur Gerard	gbondur@classicmodelcars.com	Patterson Mary	mpatterso@classicmodelcars.com
Bow Anthony	abow@classicmodelcars.com	Patterson Mary	mpatterso@classicmodelcars.com
Jennings Leslie	ljennings@classicmodelcars.com	Bow Anthony	abow@classicmodelcars.com
Thompson Leslie	lthompson@classicmodelcars.com	Bow Anthony	abow@classicmodelcars.com
Firrelli Julie	jfirrelli@classicmodelcars.com	Bow Anthony	abow@classicmodelcars.com
Patterson Steve	spatterson@classicmodelcars.com	Bow Anthony	abow@classicmodelcars.com
Tseng Foon Yue	ftseng@classicmodelcars.com	Bow Anthony	abow@classicmodelcars.com
Vanauf George	gvanauf@classicmodelcars.com	Bow Anthony	abow@classicmodelcars.com
Bondur Loui	lbondur@classicmodelcars.com	Bondur Gerard	gbondur@classicmodelcars.com
Hernandez Ger...	ghernande@classicmodelcars.com	Bondur Gerard	gbondur@classicmodelcars.com
Castillo Pamela	pcastillo@classicmodelcars.com	Bondur Gerard	gbondur@classicmodelcars.com
Bott Larry	lbott@classicmodelcars.com	Bondur Gerard	gbondur@classicmodelcars.com

❖ **Practical Exercises:**

1. Get the information about the employees and the office where they work.
2. Get the information about the items that noone has ordered.
3. Get the information about orders in March 2003 (including orderDate, requiredDate, Status) and total value of orders.
4. Get the information about the product lines and the total number of products of that product line, arranged in descending order of quantity.
5. Get the name of the customer and the total amount they purchased.