LAB 05

COMMON FUNCTIONS IN MYSQL

❖ **Main contents**

- String processing function: Substring, Concat, Replace
- Conditional function IF
- LAST_INSERT_ID function
- Time processing function: DATEDIFF, ADDDATE, EXTRACT

1. SUBSTRING processing function

Substring function allows you to extract a substring from another string, starting at a specific position and with a certain length. The following illustrates the different uses of this function.

```
SUBSTRING(str, pos);
SUBSTRING(str FROM pos);
```

The result of the above statement returns a substring from a string **str** starting at position **pos**

```
SUBSTRING(str, pos, len);
SUBSTRING(str FROM pos FOR len);
```

The above two statements return a substring from a string **str**, starting at position **pos** and the substring returning only **len** characters. Note that FROM is a standard SQL syntax keyword. Let's consider some of the following examples:

```
SELECT substring('MySQL Substring',7);
```
Result: `Substring`
```
SELECT substring('MySQL Substring' FROM 7);
```
Result: `Substring`
```
SELECT substring('MySQL Substring',7,3);
```
Result: `Sub`
```
SELECT substring('MySQL Substring' FROM 7 FOR 3);
```
Result: `Sub`

MySQL allows using negative values for pos parameters. If negative values are used for the pos parameter, the start of the substring is counted from the end of the string.

```
SELECT substring('MySQL Substring',-9);
```

Result: `Substring`

Sometimes the code snippet uses substr () instead of substring () function. Substr is a synonym for substring, so it has the same effect.

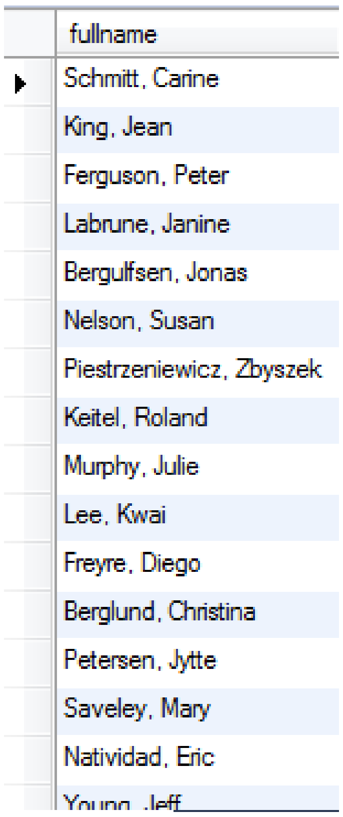
2. CONCAT function

`CONCAT` function is used to concatenate two or more strings. If the arguments are numbers, they will be converted to strings before concatenating. If any of the arguments in the argument list are NULL, the concat function will return NULL.

```
CONCAT(str1,str2,...);
```

Example: To display the first full name of a customer's contact we can use the concat function to concatenate the first and last names and the separator between them. Here is the query:

```
SELECT CONCAT(contactLastname,', ',contactFirstname)
fullname
FROM customers;
```



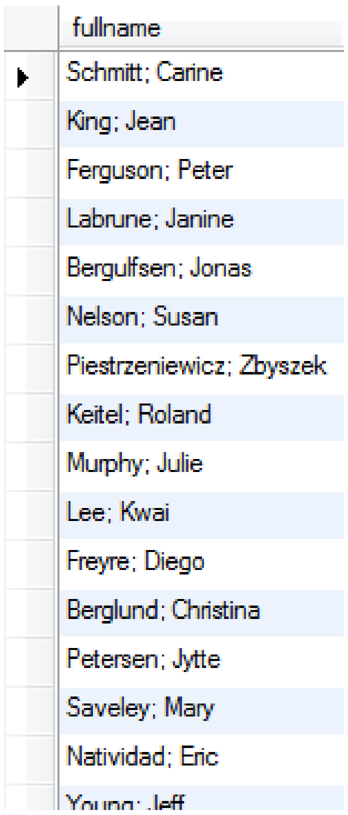
```
fullname
Schmitt, Carine
King, Jean
Ferguson, Peter
Labrune, Janine
Bergulfsen, Jonas
Nelson, Susan
Piestrzeniewicz, Zbyszek
Keitel, Roland
Murphy, Julie
Lee, Kwai
Freyre, Diego
Berglund, Christina
Petersen, Jytte
Saveley, Mary
Natividad, Eric
Young, Jeff
```

MySQL also supports the `CONCAT_WS` function, which allows us to concatenate two or more strings with a predefined delimiter. The syntax of the `CONCAT_WS` function:

```
CONCAT_WS(seperator, str1, str2,...)
```

The first parameter is the defined delimiter and then the strings you want to concatenate. The result is a concatenated string, with a separator between each paired component. For example, the same result can be achieved in the above example using `CONCAT_WS` instead of `CONCAT`.

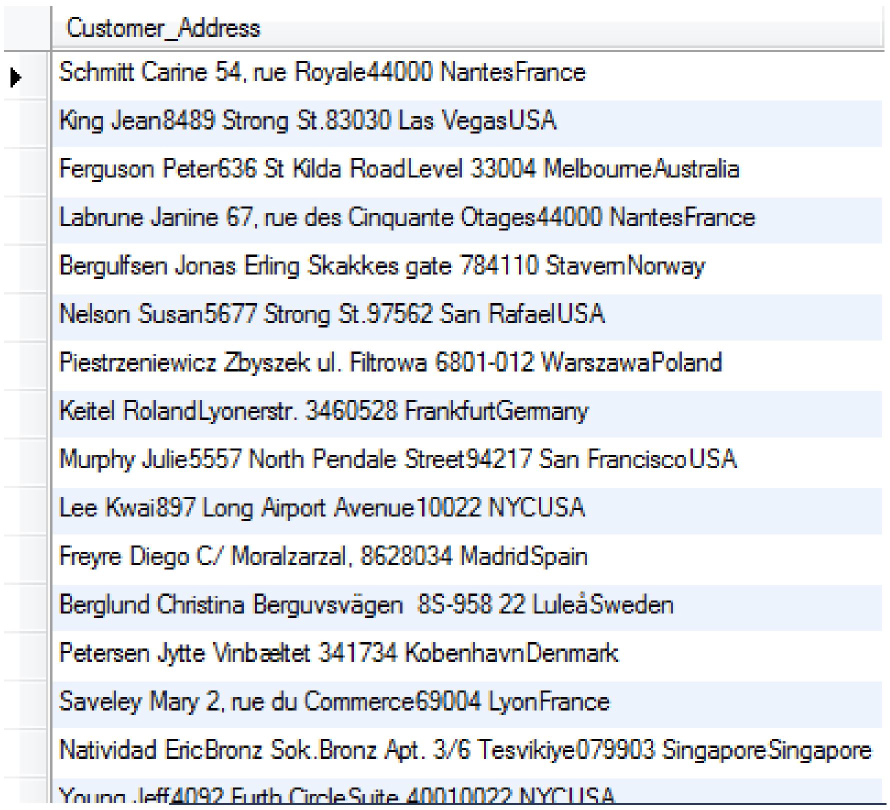
```
SELECT CONCAT_WS('; ', contactLastname, contactFirstname)
fullname
FROM customers;
```



fullname
Schmitt; Carine
King; Jean
Ferguson; Peter
Labrune; Janine
Bergulfsen; Jonas
Nelson; Susan
Piestrzeniewicz; Zbyszek
Keitel; Roland
Murphy; Julie
Lee; Kwai
Freyre; Diego
Berglund; Christina
Petersen; Jytte
Saveley; Mary
Natividad; Eric
Young; Jeff

Here is another example of using `CONCAT_WS` to get customer address format.

```
SELECT CONCAT_WS(char(10),
           CONCAT_WS(' ',contactLastname,contactFirstname),
addressLine1, addressLine2,
              CONCAT_WS(' ',postalCode,city),  country,
              CONCAT_WS(char(10),'')) AS Customer_Address
FROM customers;
```



Customer_Address
Schmitt Carine 54, rue Royale44000 NantesFrance
King Jean8489 Strong St.83030 Las VegasUSA
Ferguson Peter636 St Kilda RoadLevel 33004 MelbourneAustralia
Labrune Janine 67, rue des Cinquante Otages44000 NantesFrance
Bergulfsen Jonas Erling Skakkes gate 784110 StavernNorway
Nelson Susan5677 Strong St.97562 San RafaelUSA
Piestrzeniewicz Zbyszek ul. Filtrowa 6801-012 WarszawaPoland
Keitel RolandLyonerstr. 3460528 FrankfurtGermany
Murphy Julie5557 North Pendale Street94217 San FranciscoUSA
Lee Kwai897 Long Airport Avenue10022 NYCUSA
Freyre Diego C/ Moralzarzal, 8628034 MadridSpain
Berglund Christina Berguvsvägen 8S-958 22 LuleåSweden
Petersen Jytte Vinbæltet 341734 KobenhavnDenmark
Saveley Mary 2, rue du Commerce69004 LyonFrance
Natividad EricBronz Sok.Bronz Apt. 3/6 Tesvikiye079903 SingaporeSingapore
Young Jeff4092 Furth CircleSuite 40010022 NYCUSA

3. REPLACE function

MySQL provides a useful string handling function: `REPLACE`, that allows replacing a string in the original string with a new string. The syntax of the function is as follows:

```
REPLACE(origin string, search string, replacement string);
```

Example:

```
SELECT REPLACE('MySQL Replace', 'MySQL', 'MariaDB');
```

Result: `MariaDB Replace`

Besides, we can use `REPLACE` to replace a string in a column of a table with a new string.

The syntax of the function is as follows:

```
UPDATE <table name>
SET column name = REPLACE(column name, search string
replacement string)
WHERE <conditions>;
```

Note: When searching for alternatives, MySQL is case-sensitive.

For example, if you want to correct the spelling errors in the table *Product* in a sample database, use the `REPLACE` function as follows*:*

```
UPDATE products
SET productDescription =
REPLACE(productDescription,'abuot','about');
```

The query will look at the productDescription column and find all occurrences of the 'abuot' spelling and replace it with the exact word 'about'.

It is very important to note that in the `REPLACE` function, the first parameter is the field name not enclosed in quotation marks ‘’. If the field name is set as 'field_name', the query will update the content of the 'field_name' column, causing data loss.

4. IF function

IF is a control function, which returns a string or number based on a given condition. The syntax of the IF function is as follows:

```
IF(expr, if_true_expr, if_false_expr)
```

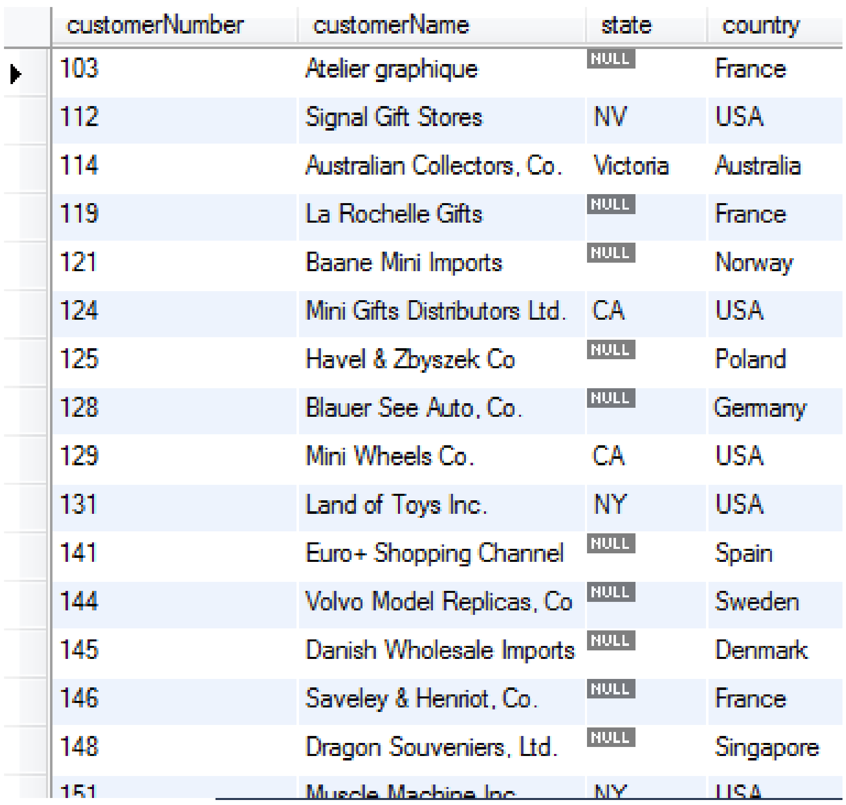
- The first parameter is expr to be checked for true or false. Actual value means that expr is not equal to 0 and expr is not equal to NULL. Note that NULL is a special value, not equal to anything else, even by itself.
- If expr is evaluated to be true, the IF function will return if_true_expr, otherwise it will return if_false_expr.

Example:

```
SELECT IF(1 = 2,'true','false');
Result: false
SELECT IF(1 = 1,' true','false');
Result: true
```

Example: In the table **Customer**, not all customers have state information. Therefore, when we select the customer, the state information will display NULL values, which are not significant for reporting purposes.

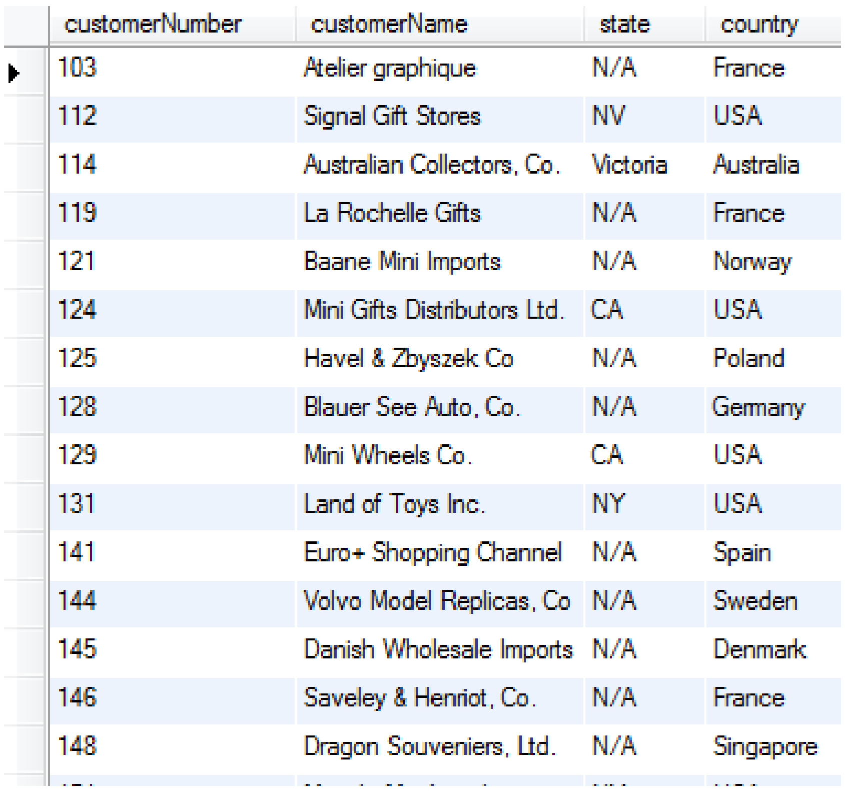
```
SELECT customerNumber,
       customerName,
        state,
        country
FROM customers;
```



customerNumber	customerName	state	country
103	Atelier graphique	NULL	France
112	Signal Gift Stores	NV	USA
114	Australian Collectors, Co.	Victoria	Australia
119	La Rochelle Gifts	NULL	France
121	Baane Mini Imports	NULL	Norway
124	Mini Gifts Distributors Ltd.	CA	USA
125	Havel & Zbyszek Co	NULL	Poland
128	Blauer See Auto, Co.	NULL	Germany
129	Mini Wheels Co.	CA	USA
131	Land of Toys Inc.	NY	USA
141	Euro+ Shopping Channel	NULL	Spain
144	Volvo Model Replicas, Co	NULL	Sweden
145	Danish Wholesale Imports	NULL	Denmark
146	Saveley & Henriot, Co.	NULL	France
148	Dragon Souveniers, Ltd.	NULL	Singapore
151	Muscle Machine Inc	NY	USA

We can use IF to display the client's status as N/A if it is NULL as follows:

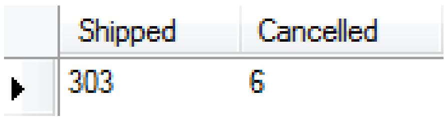
```
SELECT customerNumber,
        customerName,
        IF(state IS NULL,'N/A',state) state,
        country
FROM customers;
```



customerNumber	customerName	state	country
103	Atelier graphique	N/A	France
112	Signal Gift Stores	NV	USA
114	Australian Collectors, Co.	Victoria	Australia
119	La Rochelle Gifts	N/A	France
121	Baane Mini Imports	N/A	Norway
124	Mini Gifts Distributors Ltd.	CA	USA
125	Havel & Zbyszek Co	N/A	Poland
128	Blauer See Auto, Co.	N/A	Germany
129	Mini Wheels Co.	CA	USA
131	Land of Toys Inc.	NY	USA
141	Euro+ Shopping Channel	N/A	Spain
144	Volvo Model Replicas, Co	N/A	Sweden
145	Danish Wholesale Imports	N/A	Denmark
146	Saveley & Henriot, Co.	N/A	France
148	Dragon Souveniers, Ltd.	N/A	Singapore

Example: the IF function is also very useful with aggregate functions. Suppose if we want to know how many orders were ***Shipped*** and ***Canceled*** at the same time, we can use IF to count as follows:

```
SELECT SUM(IF(status = 'Shipped',1,0))   AS Shipped,
     SUM(IF(status = 'Cancelled',1,0)) AS Cancelled
FROM orders;
```



Shipped	Cancelled
303	6

In the above query, if the status of the order is SHIPPED or CANCELLED, IF will return the value 1, otherwise it returns 0. And then the SUM function will calculate the total to ship and be canceled based on on the return value of the IF function.

5. LAST_INSERT_ID function

`LAST_INSERT_ID` function returns the `ID` of the last record inserted into the table, provided that the ID of the column has the attribute `AUTO_INCREMENT`. In database design, we usually use an auto-increment column `AUTO_INCREMENT`. When inserting a new record into a table with column `AUTO_INCREMENT`, MySQL generates the ID for automatically based on its settings. This ID can be obtained by using the `LAST_INSERT_ID` function.

Example: Create a new table for testing called TBL. In the TBL table, we use the ID as the column `AUTO_INCREMENT`.

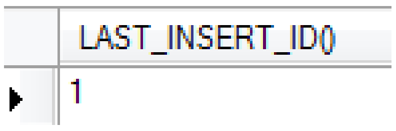
```
CREATE TABLE tbl(
     id INT AUTO_INCREMENT NOT NULL PRIMARY KEY,
        description varchar(250) NOT NULL
);
```

Then we use the `LAST_INSERT_ID()` function to get the newly inserted ID.

```
INSERT INTO tbl(description)
VALUES('MySQL last_insert_id');
```

Execute the query:

```
SELECT LAST_INSERT_ID();
```



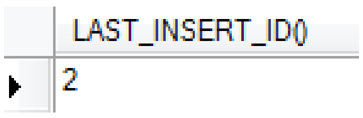
```
LAST_INSERT_ID()
1
```

It is important to note that if you insert multiple records into the table using the unique `INSERT` statement, the `LAST_INSERT_ID` function will return the value generated for the first inserted records. Try these steps:

```
INSERT INTO tbl(description)
VALUES('record 1'),
     ('record 2'),
     ('record 3');
```

Execute the query:

```
SELECT LAST_INSERT_ID();
```



```
LAST_INSERT_ID()
2
```

We have inserted 3 records using the `INSERT` statement and `LAST_INSERT_ID` function returns the ID of the first record as desired. MySQL `LAST_INSERT_ID` operates on a clientindependent principle. It means that the value returned by `LAST_INSERT_ID` for a particular client is the value that client generates. This ensures that each client can receive its own ID without having to take care of the activities of other clients and without using a lock or transaction mechanism (will be learned later).

6. DATEDIFF function

In some cases, it is necessary to calculate the number of days between two time points, for example the number of days from the shipping date to the required date in an order. In these cases, it is necessary to use the `DATEDIFF` function.

`DATEDIFF` syntax:

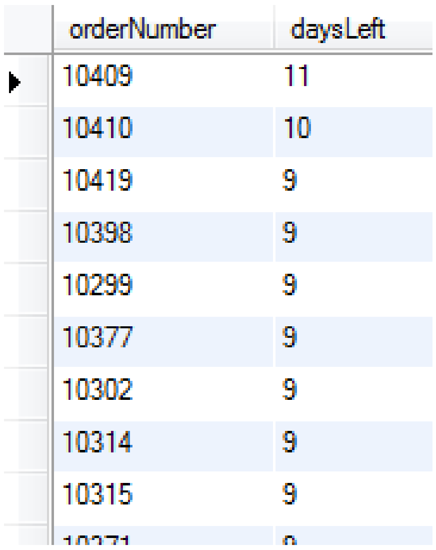
```
DATEDIFF(expr1,expr2)
```

expr1 and *expr2* are two milestones.
Example:

```
SELECT DATEDIFF('2011-08-17','2011-08-17');
Result: 0 day

SELECT DATEDIFF('2011-08-17','2011-08-08');
Result: 9 days

SELECT DATEDIFF('2011-08-08','2011-08-17');
Result: -9 days
```

Example: To calculate the number of days left between the shipping date and the required date to place an order, we use `DATEDIFF` as follows:

```
SELECT orderNumber,
DATEDIFF(requiredDate,shippedDate) AS daysLeft
FROM orders
ORDER BY daysLeft DESC;
```



```
orderNumber  daysLeft
10409        11
10410        10
10419        9
10398        9
10299        9
10377        9
10302        9
10314        9
10315        9
```

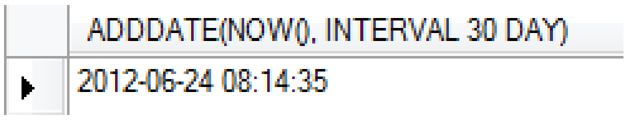
7. ADDDATE, EXTRACT function

MySQL also supports a number of other date processing functions like: ADDDATE, EXTRACT

ADDDATE function: returns a time value as a result of the operation on a different time value.

Example: Get the date 30 days after the current date and time:

```
SELECT ADDDATE(NOW(), INTERVAL 30 DAY);
```



```
ADDDATE(NOW(), INTERVAL 30 DAY)
2012-06-24 08:14:35
```

Use the keyword DAY to indicate that the value will be added as a date. Similarly we can use some of the following keywords:

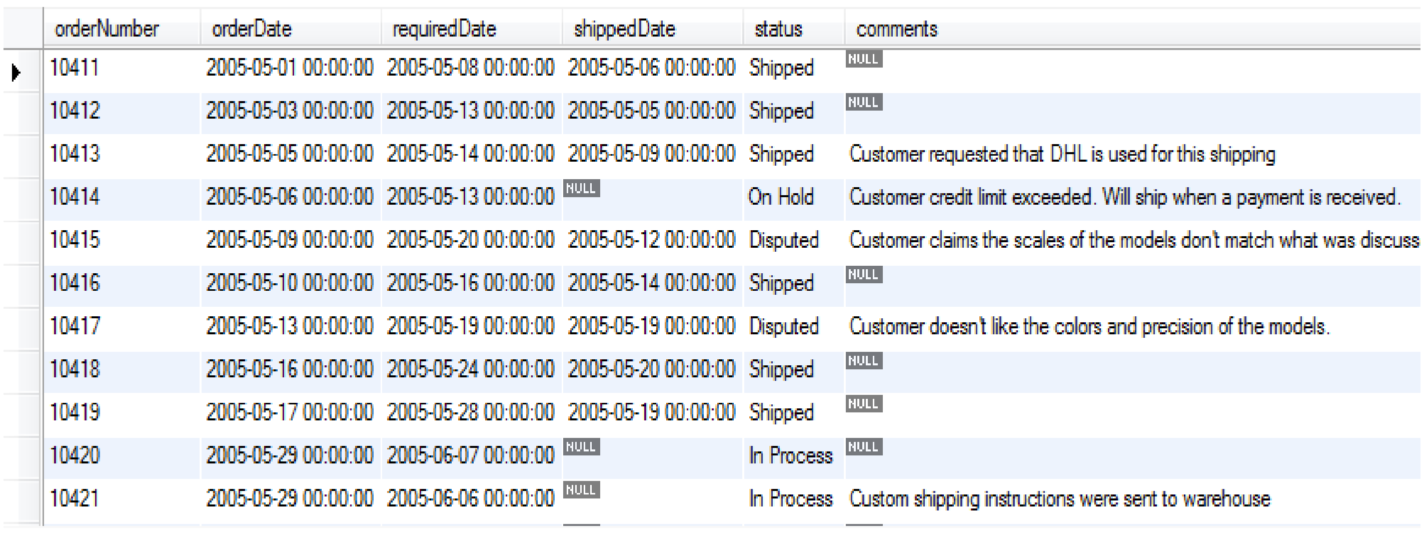
Value	Input format
MICROSECOND	MICROSECONDS
SECOND	SECONDS
MINUTE	MINUTES
HOUR	HOURS
DAY	DAYS
WEEK	WEEKS
MONTH	MONTHS
QUARTER	QUARTERS
YEAR	YEARS
SECOND_MICROSECOND	'SECONDS.MICROSECONDS'

Value	Input format
MINUTE_MICROSECOND	'MINUTES:SECONDS.MICROSECONDS'
MINUTE_SECOND	'MINUTES:SECONDS'
HOUR_MICROSECOND	'HOURS:MINUTES:SECONDS.MICROSECONDS'
HOUR_SECOND	'HOURS:MINUTES:SECONDS'
HOUR_MINUTE	'HOURS:MINUTES'
DAY_MICROSECOND	'DAYS HOURS:MINUTES:SECONDS.MICROSECONDS'
DAY_SECOND	'DAYS HOURS:MINUTES:SECONDS'
DAY_MINUTE	'DAYS HOURS:MINUTES'
DAY_HOUR	'DAYS HOURS'
YEAR_MONTH	'YEARS-MONTHS'

Example: Get the orders within 30 days from May 1, 2005

```
SELECT *
FROM orders
WHERE orderDate>= '2005-5-1' AND orderDate < ADDDATE('2005-5-
1', INTERVAL 30 DAY);
```

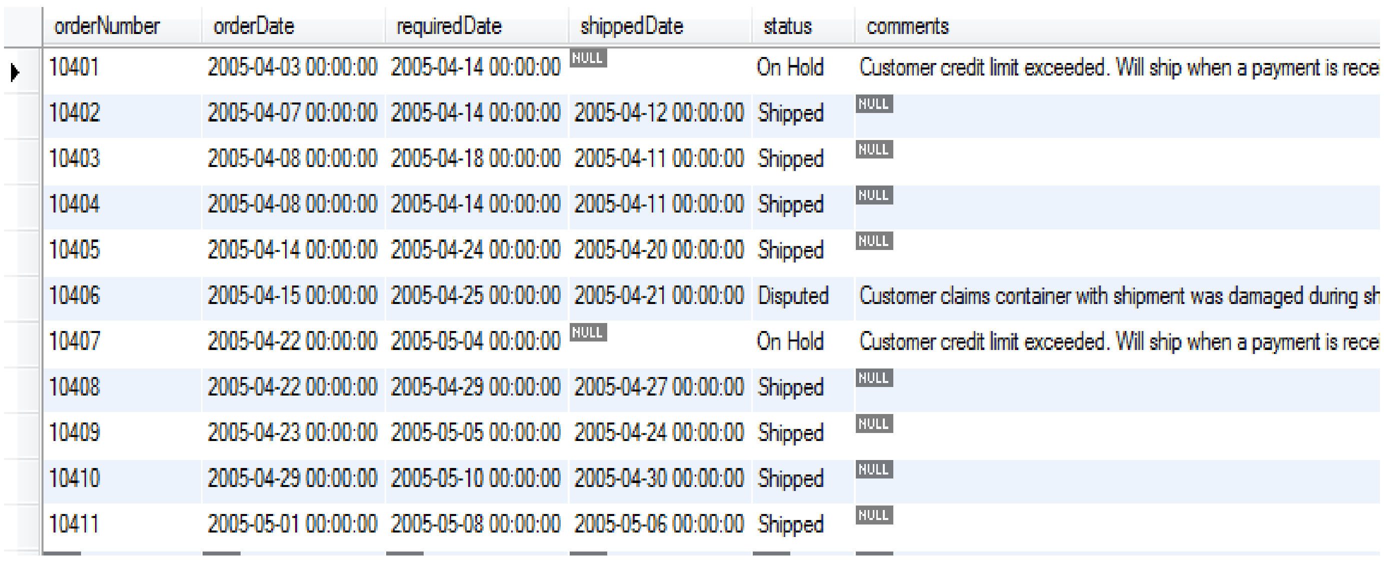
Result:



orderNumber	orderDate	requiredDate	shippedDate	status	comments
10411	2005-05-01 00:00:00	2005-05-08 00:00:00	2005-05-06 00:00:00	Shipped	NULL
10412	2005-05-03 00:00:00	2005-05-13 00:00:00	2005-05-05 00:00:00	Shipped	NULL
10413	2005-05-05 00:00:00	2005-05-14 00:00:00	2005-05-09 00:00:00	Shipped	Customer requested that DHL is used for this shipping
10414	2005-05-06 00:00:00	2005-05-13 00:00:00	NULL	On Hold	Customer credit limit exceeded. Will ship when a payment is received.
10415	2005-05-09 00:00:00	2005-05-20 00:00:00	2005-05-12 00:00:00	Disputed	Customer claims the scales of the models don't match what was discuss
10416	2005-05-10 00:00:00	2005-05-16 00:00:00	2005-05-14 00:00:00	Shipped	NULL
10417	2005-05-13 00:00:00	2005-05-19 00:00:00	2005-05-19 00:00:00	Disputed	Customer doesn't like the colors and precision of the models.
10418	2005-05-16 00:00:00	2005-05-24 00:00:00	2005-05-20 00:00:00	Shipped	NULL
10419	2005-05-17 00:00:00	2005-05-28 00:00:00	2005-05-19 00:00:00	Shipped	NULL
10420	2005-05-29 00:00:00	2005-06-07 00:00:00	NULL	In Process	NULL
10421	2005-05-29 00:00:00	2005-06-06 00:00:00	NULL	In Process	Custom shipping instructions were sent to warehouse

Example: Get the orders from May 1, 2005 30 days before to May 1, 2005

```
SELECT *
FROM orders
WHERE orderDate<= '2005-5-1' AND orderDate > ADDDATE('2005-5-
1', INTERVAL -30 DAY);
```

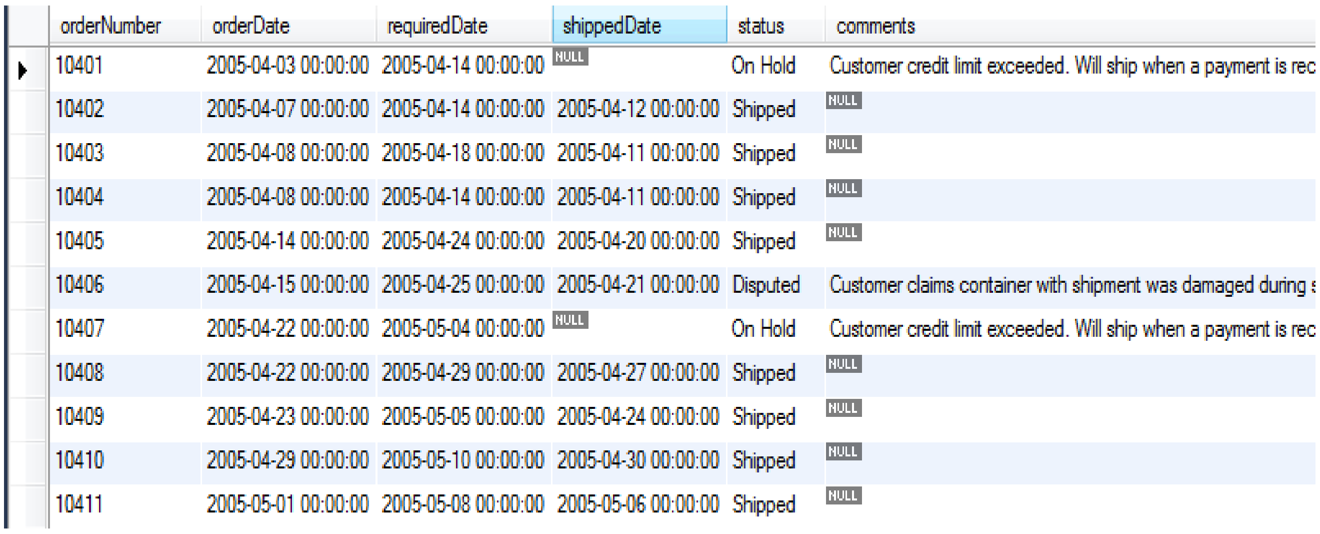


orderNumber	orderDate	requiredDate	shippedDate	status	comments
10401	2005-04-03 00:00:00	2005-04-14 00:00:00	NULL	On Hold	Customer credit limit exceeded. Will ship when a payment is rece
10402	2005-04-07 00:00:00	2005-04-14 00:00:00	2005-04-12 00:00:00	Shipped	NULL
10403	2005-04-08 00:00:00	2005-04-18 00:00:00	2005-04-11 00:00:00	Shipped	NULL
10404	2005-04-08 00:00:00	2005-04-14 00:00:00	2005-04-11 00:00:00	Shipped	NULL
10405	2005-04-14 00:00:00	2005-04-24 00:00:00	2005-04-20 00:00:00	Shipped	NULL
10406	2005-04-15 00:00:00	2005-04-25 00:00:00	2005-04-21 00:00:00	Disputed	Customer claims container with shipment was damaged during sh
10407	2005-04-22 00:00:00	2005-05-04 00:00:00	NULL	On Hold	Customer credit limit exceeded. Will ship when a payment is rece
10408	2005-04-22 00:00:00	2005-04-29 00:00:00	2005-04-27 00:00:00	Shipped	NULL
10409	2005-04-23 00:00:00	2005-05-05 00:00:00	2005-04-24 00:00:00	Shipped	NULL
10410	2005-04-29 00:00:00	2005-05-10 00:00:00	2005-04-30 00:00:00	Shipped	NULL
10411	2005-05-01 00:00:00	2005-05-08 00:00:00	2005-05-06 00:00:00	Shipped	NULL

If the time added is month, year, the corresponding keywords used are MONTH, YEAR.

The example above can be rewritten as follows:

```
SELECT *
FROM orders
WHERE orderDate<= '2005-5-1' AND orderDate > ADDDATE('2005-5-
1', INTERVAL -1 MONTH);
```

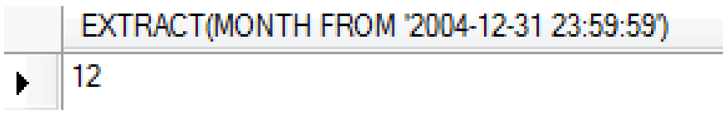


orderNumber	orderDate	requiredDate	shippedDate	status	comments
10401	2005-04-03 00:00:00	2005-04-14 00:00:00	NULL	On Hold	Customer credit limit exceeded. Will ship when a payment is rec
10402	2005-04-07 00:00:00	2005-04-14 00:00:00	2005-04-12 00:00:00	Shipped	NULL
10403	2005-04-08 00:00:00	2005-04-18 00:00:00	2005-04-11 00:00:00	Shipped	NULL
10404	2005-04-08 00:00:00	2005-04-14 00:00:00	2005-04-11 00:00:00	Shipped	NULL
10405	2005-04-14 00:00:00	2005-04-24 00:00:00	2005-04-20 00:00:00	Shipped	NULL
10406	2005-04-15 00:00:00	2005-04-25 00:00:00	2005-04-21 00:00:00	Disputed	Customer claims container with shipment was damaged during s
10407	2005-04-22 00:00:00	2005-05-04 00:00:00	NULL	On Hold	Customer credit limit exceeded. Will ship when a payment is rec
10408	2005-04-22 00:00:00	2005-04-29 00:00:00	2005-04-27 00:00:00	Shipped	NULL
10409	2005-04-23 00:00:00	2005-05-05 00:00:00	2005-04-24 00:00:00	Shipped	NULL
10410	2005-04-29 00:00:00	2005-05-10 00:00:00	2005-04-30 00:00:00	Shipped	NULL
10411	2005-05-01 00:00:00	2005-05-08 00:00:00	2005-05-06 00:00:00	Shipped	NULL

EXTRACT function: separates values like day, month, year from a value of time type. (Note that MONTH or YEAR may be used instead.)

Example: Get the month of a time value:

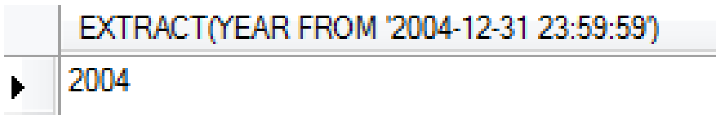
```
SELECT EXTRACT(MONTH FROM '2004-12-31 23:59:59');
```



EXTRACT(MONTH FROM '2004-12-31 23:59:59')
12

Example: Get the year of a time value:

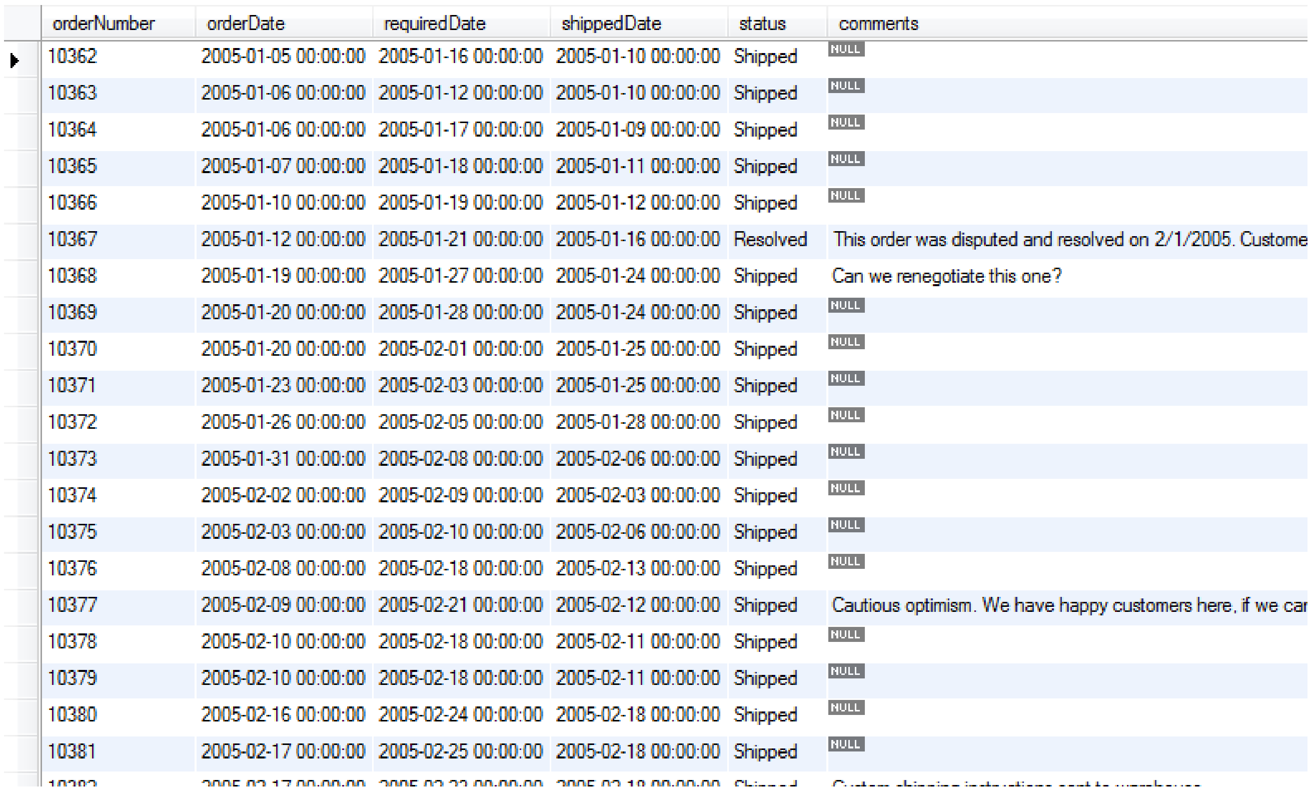
```
SELECT EXTRACT(YEAR FROM '2004-12-31 23:59:59');
```



EXTRACT(YEAR FROM '2004-12-31 23:59:59')
2004

Example: Get the orders placed in 2005:

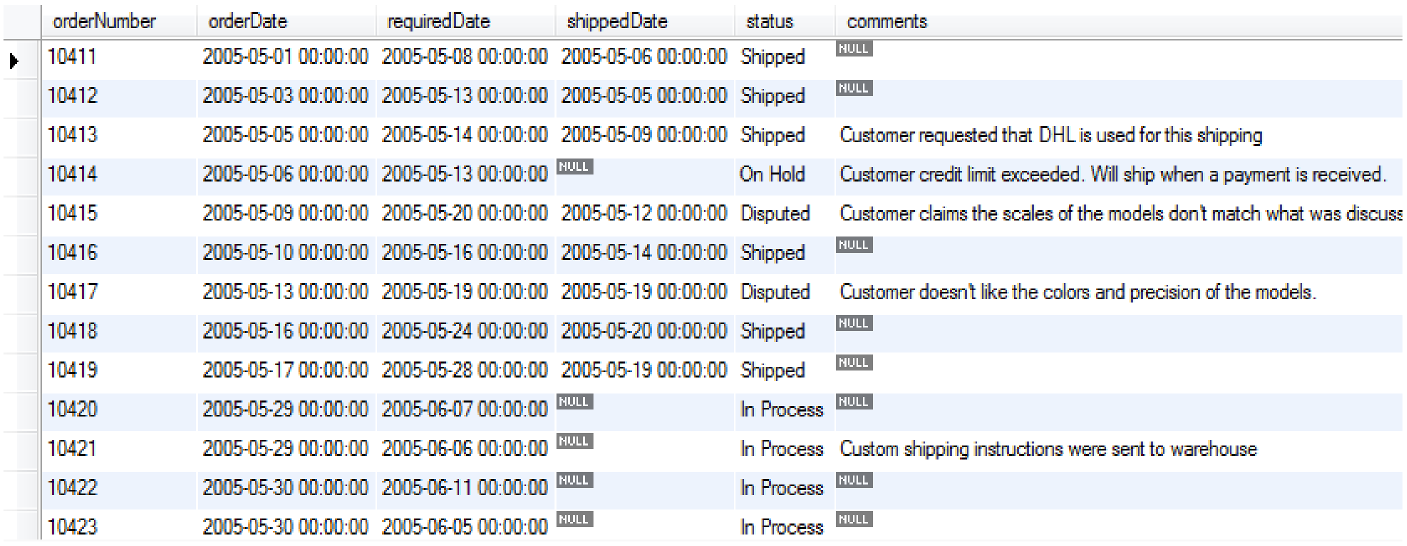
```
SELECT *

FROM orders

WHERE EXTRACT(YEAR FROM orderDate) = 2005;
```



orderNumber	orderDate	requiredDate	shippedDate	status	comments
10362	2005-01-05 00:00:00	2005-01-16 00:00:00	2005-01-10 00:00:00	Shipped	NULL
10363	2005-01-06 00:00:00	2005-01-12 00:00:00	2005-01-10 00:00:00	Shipped	NULL
10364	2005-01-06 00:00:00	2005-01-17 00:00:00	2005-01-09 00:00:00	Shipped	NULL
10365	2005-01-07 00:00:00	2005-01-18 00:00:00	2005-01-11 00:00:00	Shipped	NULL
10366	2005-01-10 00:00:00	2005-01-19 00:00:00	2005-01-12 00:00:00	Shipped	NULL
10367	2005-01-12 00:00:00	2005-01-21 00:00:00	2005-01-16 00:00:00	Resolved	This order was disputed and resolved on 2/1/2005. Custome
10368	2005-01-19 00:00:00	2005-01-27 00:00:00	2005-01-24 00:00:00	Shipped	Can we renegotiate this one?
10369	2005-01-20 00:00:00	2005-01-28 00:00:00	2005-01-24 00:00:00	Shipped	NULL
10370	2005-01-20 00:00:00	2005-02-01 00:00:00	2005-01-25 00:00:00	Shipped	NULL
10371	2005-01-23 00:00:00	2005-02-03 00:00:00	2005-01-25 00:00:00	Shipped	NULL
10372	2005-01-26 00:00:00	2005-02-05 00:00:00	2005-01-28 00:00:00	Shipped	NULL
10373	2005-01-31 00:00:00	2005-02-08 00:00:00	2005-02-06 00:00:00	Shipped	NULL
10374	2005-02-02 00:00:00	2005-02-09 00:00:00	2005-02-03 00:00:00	Shipped	NULL
10375	2005-02-03 00:00:00	2005-02-10 00:00:00	2005-02-06 00:00:00	Shipped	NULL
10376	2005-02-08 00:00:00	2005-02-18 00:00:00	2005-02-13 00:00:00	Shipped	NULL
10377	2005-02-09 00:00:00	2005-02-21 00:00:00	2005-02-12 00:00:00	Shipped	Cautious optimism. We have happy customers here, if we car
10378	2005-02-10 00:00:00	2005-02-18 00:00:00	2005-02-11 00:00:00	Shipped	NULL
10379	2005-02-10 00:00:00	2005-02-18 00:00:00	2005-02-11 00:00:00	Shipped	NULL
10380	2005-02-16 00:00:00	2005-02-24 00:00:00	2005-02-18 00:00:00	Shipped	NULL
10381	2005-02-17 00:00:00	2005-02-25 00:00:00	2005-02-18 00:00:00	Shipped	NULL
10382	2005-02-17 00:00:00	2005-02-23 00:00:00	2005-02-18 00:00:00	Shipped	Custom shipping instructions sent to warehouse

Example: Get the orders placed in May 2005:

```
SELECT *
FROM orders
WHERE EXTRACT(YEAR FROM orderDate) = 2005 and EXTRACT(MONTH
FROM orderDate) = 5;
```



orderNumber	orderDate	requiredDate	shippedDate	status	comments
10411	2005-05-01 00:00:00	2005-05-08 00:00:00	2005-05-06 00:00:00	Shipped	NULL
10412	2005-05-03 00:00:00	2005-05-13 00:00:00	2005-05-05 00:00:00	Shipped	NULL
10413	2005-05-05 00:00:00	2005-05-14 00:00:00	2005-05-09 00:00:00	Shipped	Customer requested that DHL is used for this shipping
10414	2005-05-06 00:00:00	2005-05-13 00:00:00	NULL	On Hold	Customer credit limit exceeded. Will ship when a payment is received.
10415	2005-05-09 00:00:00	2005-05-20 00:00:00	2005-05-12 00:00:00	Disputed	Customer claims the scales of the models don't match what was discuss
10416	2005-05-10 00:00:00	2005-05-16 00:00:00	2005-05-14 00:00:00	Shipped	NULL
10417	2005-05-13 00:00:00	2005-05-19 00:00:00	2005-05-19 00:00:00	Disputed	Customer doesn't like the colors and precision of the models.
10418	2005-05-16 00:00:00	2005-05-24 00:00:00	2005-05-20 00:00:00	Shipped	NULL
10419	2005-05-17 00:00:00	2005-05-28 00:00:00	2005-05-19 00:00:00	Shipped	NULL
10420	2005-05-29 00:00:00	2005-06-07 00:00:00	NULL	In Process	NULL
10421	2005-05-29 00:00:00	2005-06-06 00:00:00	NULL	In Process	Custom shipping instructions were sent to warehouse
10422	2005-05-30 00:00:00	2005-06-11 00:00:00	NULL	In Process	NULL
10423	2005-05-30 00:00:00	2005-06-05 00:00:00	NULL	In Process	NULL

❖ **Practical exercises:**

1. Get the first 50 characters of the product description, naming it ‘Title of products’.

2. Get the descriptions of employees in the format ‘Fullname, jobTitle’.

3. Update the employees’ information whose jobTitle is ‘Sales Rep’ to ‘Sales Representative’.

4. Get 5 orders shipped sooner than the required date.

5. Get the orders in May 2005 with an unspecified shipped date.