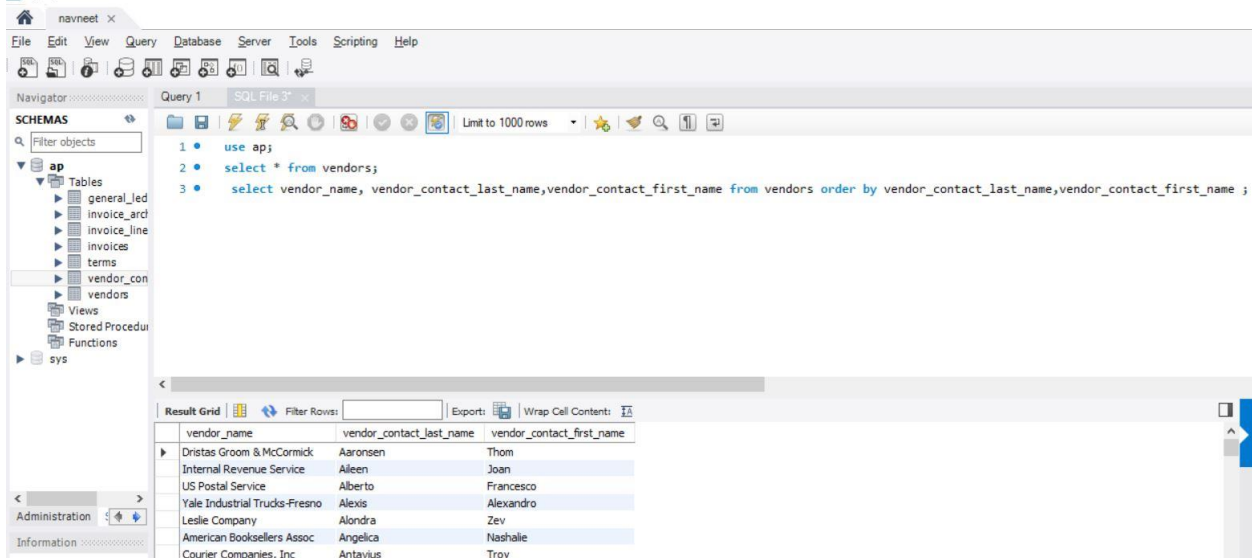


MYSQL LAB 3

6) Write a **SELECT** statement that returns three columns from the Vendors table: **vendor_name**, **vendor_contact_last_name**, and **vendor_contact_first_name**.

Then, run this statement to make sure it works correctly.

Add an **ORDER BY** clause to this statement that sorts the result set by last name and then first name, both in ascending sequence.



7) Write a **SELECT** statement that returns one column from the Vendors table named **full_name** that joins the **vendor_contact_last_name** and **vendor_contact_first_name** columns.

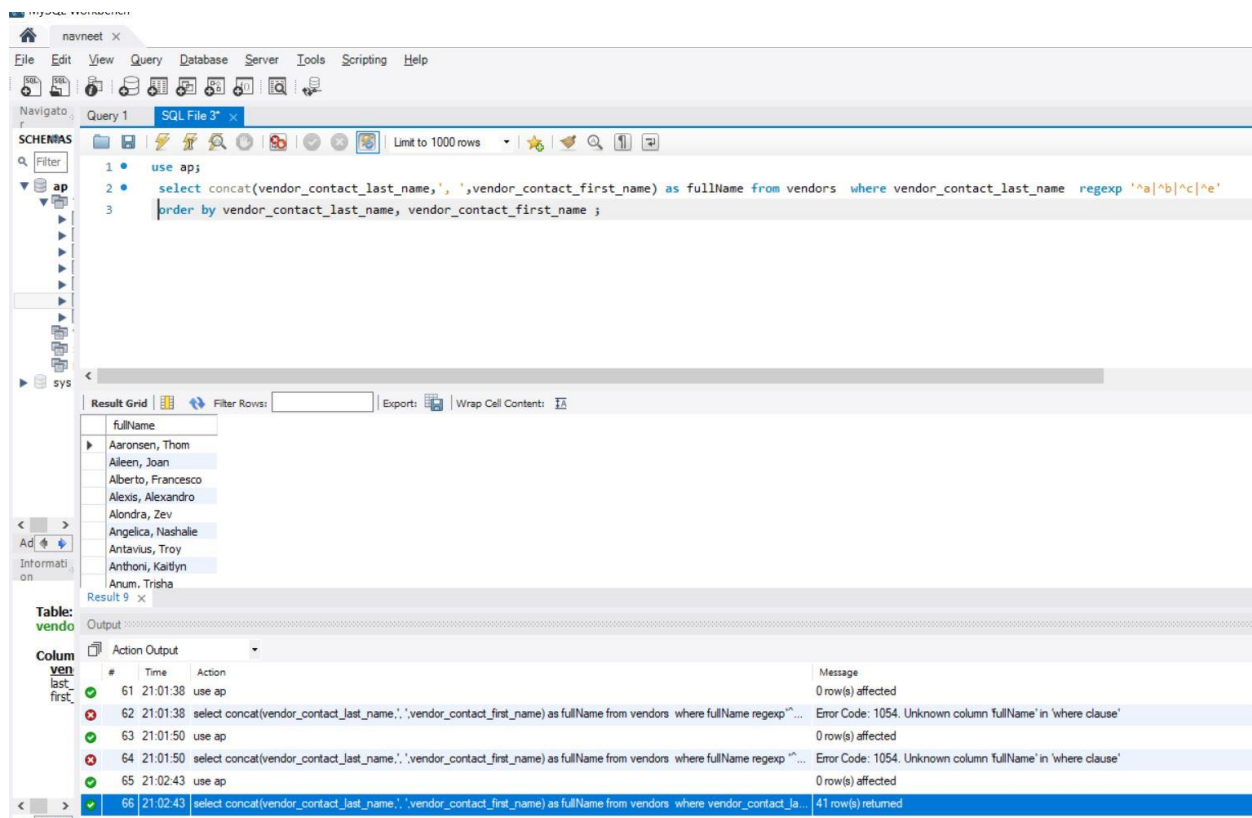
Format this column with the last name, a comma, a space, and the first name like this: Doe, John Sort the result set by last name and then first name in ascending sequence.

Return only the contacts whose last name begins with the letter A, B, C, or E. This should retrieve 41 rows

Answer:

```
select concat(vendor_contact_last_name, ', ', vendor_contact_first_name) as fullName from vendors  
where vendor_contact_last_name regexp '^a|^b|^c|^e'
```

```
order by vendor_contact_last_name, vendor_contact_first_name ;
```



8) Write a SELECT statement that returns these column names and data from the Invoices table:

the Invoices table:

Due Date	The invoice_due_date column
Invoice Total	The invoice_total column
10%	10% of the value of invoice_total
Plus 10%	The value of invoice_total plus 10%

Return only the rows with an **invoice total** that's greater than or equal to 500 and less than or equal to 1000. This should retrieve 12 rows.

Sort the result set in descending sequence by invoice_due_date.

Answer:

```
select invoice_due_date as "Due Date",invoice_total as "Invoice Total",(0.10* invoice_total) as "10%",(1.1*invoice_total) as "Plus 10%" from
invoices
```

```
where invoice_total>=500 and invoice_total<=1000
```

```
order by invoice_due_date desc
```

The screenshot shows the Navicat SQL editor interface. On the left, the 'SCHEMAS' pane displays a tree view of the database structure, including tables like 'general_ledger_accounts', 'invoice_archive', 'invoice_line_items', 'invoices', 'terms', 'vendor_contacts', and 'vendors'. The main window displays a SQL query in the 'Query 1' tab:

```

1 use ap;
2 select invoice_due_date as "Due Date",invoice_total as "Invoice Total", (0.10* invoice_total) as "10%", (1.1*invoice_total) as "Plus 10%" from invoices
3 where invoice_total>500 and invoice_total<1000
4 order by invoice_due_date desc
5 ;

```

Below the query, the 'Result Grid' shows the results of the query. The columns are 'Due Date', 'Invoice Total', '10%', and 'Plus 10%'. The results are sorted by 'Due Date' in descending order. The first few rows are:

Due Date	Invoice Total	10%	Plus 10%
2014-08-23	503.20	50.3200	553.520
2014-08-10	579.42	57.9420	637.362
2014-08-09	600.00	60.0000	660.000
2014-08-06	739.20	73.9200	813.120
2014-07-25	904.14	90.4140	994.554

At the bottom, the 'Action Output' pane shows the execution log. The last action is 'select invoice_due_date as "Due Date",invoice_total as "Invoice Total", (0.10* invoice_total) as "10%", (1.1*invoice_total) as "Plus 10%" from invoices LIMIT 0, 1000', which returned 12 rows. The duration of the action is 0.000 seconds.

9) Write a SELECT statement that returns these columns from the Invoices table:

invoice_number	The invoice_number column
invoice_total	The invoice_total column
payment_credit_total	Sum of the payment_total and credit_total columns
balance_due	The invoice_total column minus the payment_total and credit_total columns

Return only invoices that have a balance due that's greater than \$50.

Sort the result set by balance due in descending sequence.

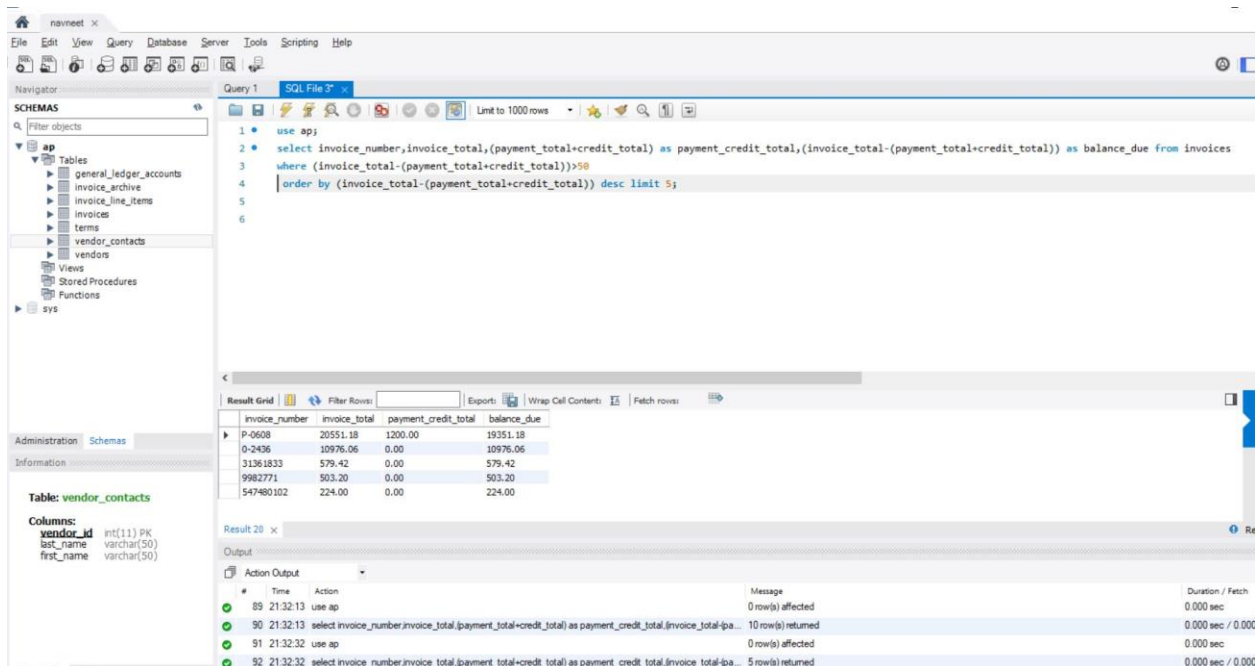
Use the LIMIT clause so the result set contains only the rows with the 5 largest balances.

Answer:

```

select invoice_number,invoice_total,(payment_total+credit_total) as
payment_credit_total,(invoice_total-(payment_total+credit_total)) as balance_due from invoices
where (invoice_total-(payment_total+credit_total))>50
order by (invoice_total-(payment_total+credit_total)) desc limit 5;

```



Work with nulls and test expressions

10) Write a SELECT statement that returns these columns from the Invoices table

invoice_number	The invoice_number column
invoice_date	The invoice_date column
balance_due	The invoice_total column minus the payment_total and credit_total columns
payment_date	The payment_date column

Return only the rows where the payment_date column contains a null value. This should retrieve 11 rows.

Answer:

```

select invoice_number,invoice_date,(invoice_total-payment_total-credit_total) as
balance_due,payment_date from invoices

```

where payment_date is null ;

Query 1 SQL File 3'

```
1 use ap;  
2 select invoice_number,invoice_date,(invoice_total-payment_total-credit_total) as balance_due,payment_date from invoices  
3 where payment_date is null ;  
4
```

invoice_number	invoice_date	balance_due	payment_date
39104	2014-07-10	85.31	NULL
963253264	2014-07-18	52.25	NULL
31361833	2014-07-21	579.42	NULL
263253268	2014-07-21	59.97	NULL
263253270	2014-07-22	67.92	NULL

Result 30 x

Output

#	Time	Action	Message
107	21:40:44	use ap	0 row(s) affected
108	21:40:44	select invoice_number,invoice_date,(invoice_total-payment_total-credit_total) as balance_due,payment_d...	Error Code: 1525. Incorrect DATE value: 'NULL'
109	21:41:04	select invoice_number,invoice_date,(invoice_total-payment_total-credit_total) as balance_due,payment_d...	0 row(s) returned
110	21:41:06	select invoice_number,invoice_date,(invoice_total-payment_total-credit_total) as balance_due,payment_d...	0 row(s) returned
111	21:41:13	select invoice_number,invoice_date,(invoice_total-payment_total-credit_total) as balance_due,payment_d...	0 row(s) returned
112	21:43:03	use ap	0 row(s) affected
113	21:43:03	select invoice_number,invoice_date,(invoice_total-payment_total-credit_total) as balance_due,payment_d...	114 row(s) returned
114	21:43:34	use ap	0 row(s) affected
115	21:43:34	select invoice_number,invoice_date,(invoice_total-payment_total-credit_total) as balance_due,payment_d...	0 row(s) returned
116	21:44:37	use ap	0 row(s) affected
117	21:44:37	select invoice_number,invoice_date,(invoice_total-payment_total-credit_total) as balance_due,payment_d...	11 row(s) returned

11) Write a SELECT statement without a FROM clause that uses the CURRENT_DATE function to return the current date in its default format.

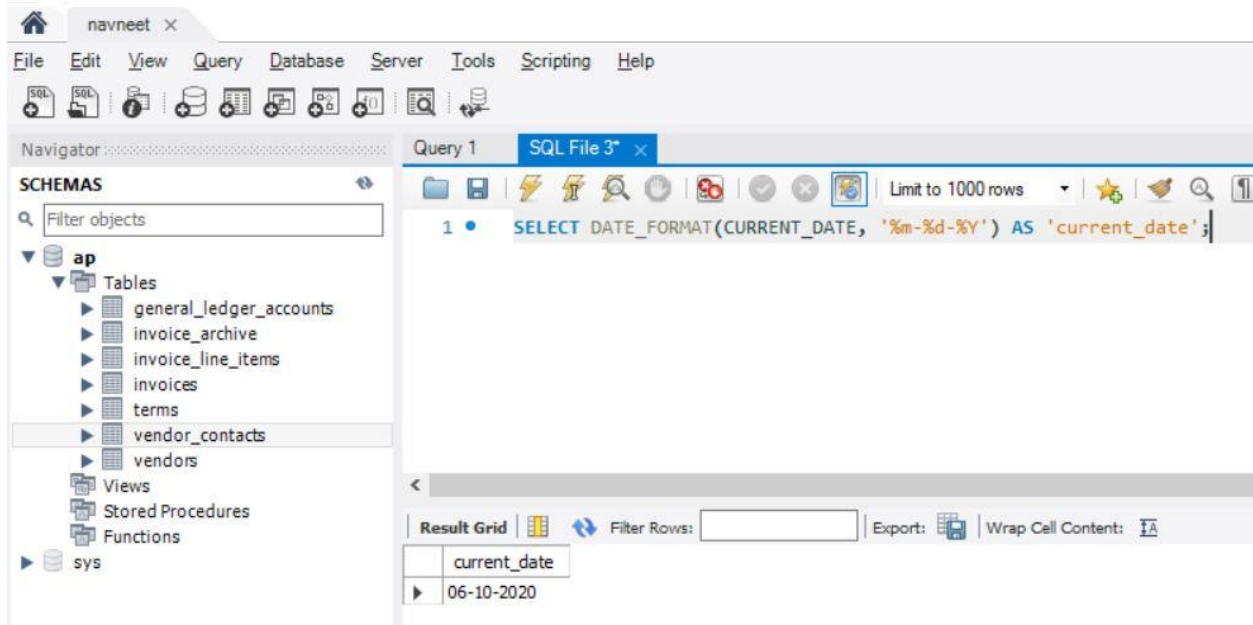
Use the DATE_FORMAT function to format the current date in this format: mm-dd-yyyy .

This displays the month, day, and four-digit year of the current date.

Give this column an alias of current_date.

To do that, you must enclose the alias in quotes since that name is already used by the CURRENT_DATE

Answer: SELECT DATE_FORMAT(CURRENT_DATE, '%m-%d-%Y') AS 'current_date';



12) Write a SELECT statement without a FROM clause that creates a row with these columns:

starting_principal	Starting principal of \$50,000
interest	6.5% of the principal
principal_plus_interest	The principal plus the interest

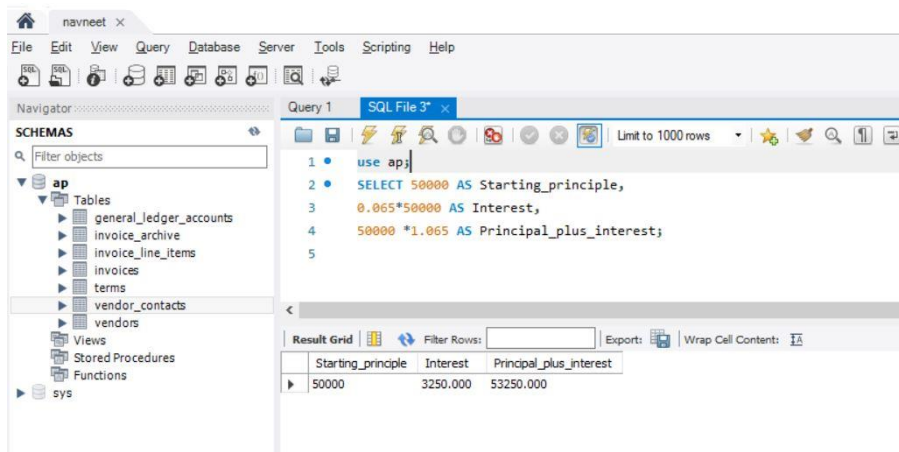
To calculate the third column, add the expressions you used for the first two columns.

Answer:

SELECT 50000 AS Starting_principal,

0.065*50000 AS Interest,

1.065*50000 AS Principal_plus_interest;



ANSWER FOR EXERCISE

```
6) SELECT vendor_name, vendor_contact_last_name, vendor_contact_first_name
FROM vendors
ORDER BY vendor_contact_last_name, vendor_contact_first_name

7) SELECT CONCAT(vendor_contact_last_name, ', ', vendor_contact_first_name) AS full_name
FROM vendors
WHERE vendor_contact_last_name < 'D' OR vendor_contact_last_name LIKE 'E%'
ORDER BY vendor_contact_last_name, vendor_contact_first_name

8) SELECT invoice_due_date AS "Due Date",
       invoice_total AS "Invoice Total",
       invoice_total / 10 AS "10%",
       invoice_total * 1.1 AS "Plus 10%"
FROM invoices
WHERE invoice_total >= 500 AND invoice_total <= 1000
ORDER BY invoice_due_date DESC

9) SELECT invoice_number,
       invoice_total,
       payment_total + credit_total AS payment_credit_total,
       invoice_total - payment_total - credit_total AS balance_due
FROM invoices
WHERE invoice_total - payment_total - credit_total > 50
ORDER BY balance_due DESC
LIMIT 5;

10) SELECT invoice_number,
       invoice_date,
       invoice_total - payment_total - credit_total AS balance_due,
       payment_date
FROM invoices
WHERE payment_date IS NULL

11) SELECT DATE_FORMAT(CURRENT_DATE, '%m-%d-%Y') AS "current_date"

12) SELECT 50000 AS starting_principle,
       50000 * .065 AS interest,
       (50000) + (50000 * .065) AS principle_plus_interest
```

USE EX DATABASE

1) IS NULL

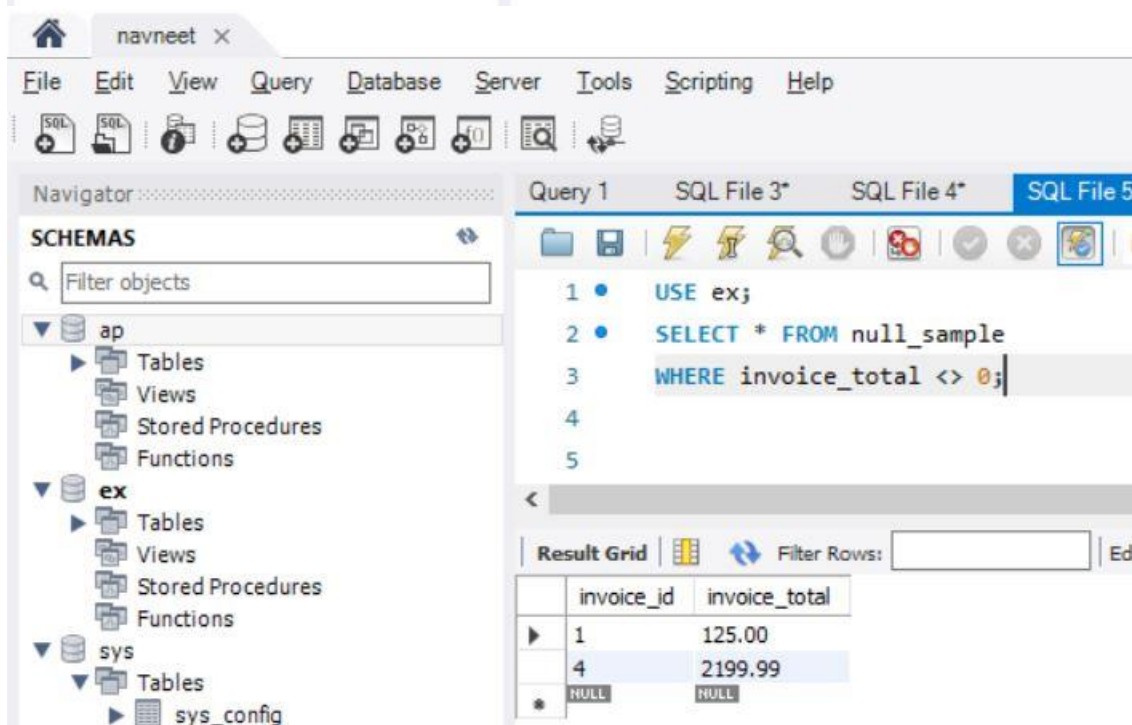
The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'SCHEMAS' tree with databases 'ap', 'ex', and 'sys'. The 'ex' database is expanded, showing 'Tables', 'Views', 'Stored Procedures', and 'Functions'. The right pane shows a query window with the following SQL code:

```
USE ex;

SELECT * FROM null_sample;
```

The query results are displayed in the 'Result Grid' below the query window. The grid has two columns: 'invoice_id' and 'invoice_total'. The results are as follows:

invoice_id	invoice_total
1	125.00
2	0.00
3	NULL
4	2199.99
5	0.00
NULL	NULL



navneet x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

ap

- Tables
- Views
- Stored Procedures
- Functions

ex

- Tables
- Views
- Stored Procedures
- Functions

sys

- Tables
- sys_config
- Views

Query 1 SQL File 3* SQL File 4* SQL File 5* x ap ap - Schema

1 • USE ex;

2 • SELECT * FROM null_sample

3 WHERE invoice_total IS NULL;

4

5

Result Grid

invoice_id	invoice_total
3	NULL
*	NULL

navneet x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

ap

- Tables
- Views
- Stored Procedures
- Functions

ex

- Tables
- Views
- Stored Procedures
- Functions

sys

- Tables
- sys_config
- Views
- Stored Procedures
- Functions

Query 1 SQL File 3* SQL File 4* SQL File 5* x ap

1 • USE ex;

2

3 • SELECT *

4 FROM null_sample

5 WHERE invoice_total IS NOT NULL;

Result Grid

invoice_id	invoice_total
1	125.00
2	0.00
4	2199.99
5	0.00
*	NULL

