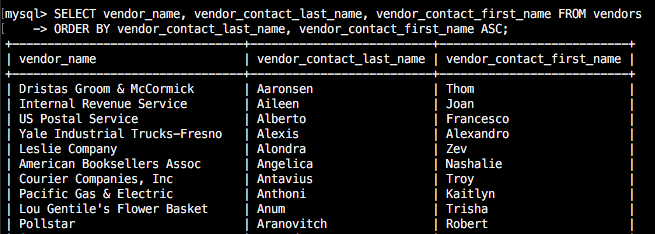
Use the AP database on our server DBA120.abtech.edu to execute the following SELECT statements.

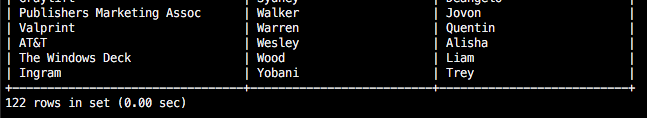
\*Screen clip your query and results below each instruction**. (16.6 points each).**

1. 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. Then, run this statement again to make sure it works correctly. This is a good way to build and test a statement, one clause at a time.

**Just clip the first 10 lines and the last 5 lines along with the row count.**

****

****

1. 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.

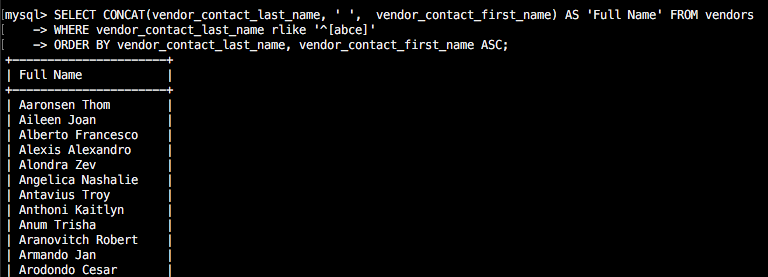
Format this column with the last name, a comma, a space, and the first name like this:

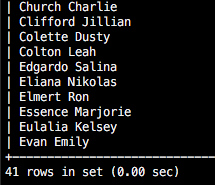
Doe, John

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.

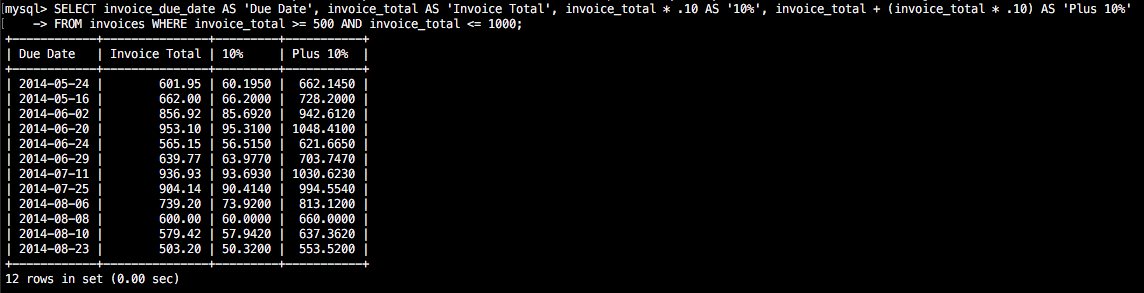
****

****

1. Write a select statement that returns these column names and data from the Invoices table:

|  |  |
| --- | --- |
| Column Name | Contains ….. |
| 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.

****

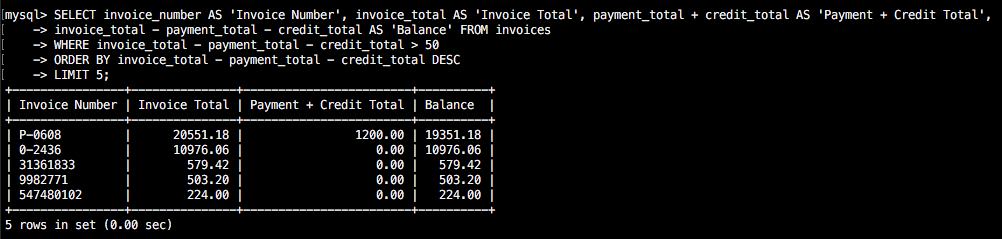
1. Write a SELECT statement that returns these column name and data from the Invoices table:

|  |  |
| --- | --- |
| Column Name | Contains ….. |
| 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 | 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.

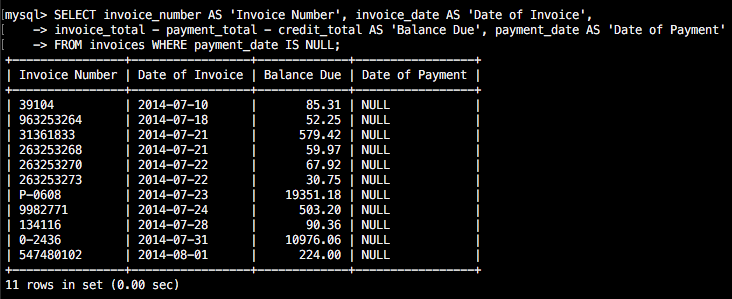
****

1. Write a SELECT statement that returns these columns from the Invoices table:

|  |  |
| --- | --- |
| Column Name | Contains ….. |
| Invoice Number | The invoice\_number column |
| Date of Invoice | The invoice\_date column |
| Balance Due | The invoice\_total column minus the payment\_total and credit\_total columns |
| Date of Payment | The payment\_date column |

Return only the rows where the payment\_date column contains a null value.

This should retrieve 11 rows.



1. Write a SELECT statement without a FROM clause that creates a row with these columns:

starting\_principle starting principle of $50,000

interest 6.5% of the principal

principal\_plus\_interest The principle plus the interest

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

**SELECT starting\_principle, starting\_principle \* .065 AS interest, starting\_principle + (starting\_principle \* .065) AS principle\_plus\_interest;**