INST 327 – Database Design and Modeling

Assignment 4 - See Canvas for the deadline

See the assignment on Canvas for the partially completed code that you need for Q2 & Q3. You will need those before you begin working on these questions.
(There are three questions in this document)

Q.1) (35 points)

We would like to track which vendors have been paying for their invoices late so create a view called `late_invoices` that lists invoices that were paid after their due date. We would also like to include invoices that have not yet been paid in our result set. Since we know all invoices come from the year 2014, format all date columns as the written out month and day (ex: “July 4th”). Include the following columns:

- The vendor’s name
- The invoice number
- The invoice’s due date
- The date the invoice was paid
- Number of days late - the DATEDIFF() function returns the numerical difference between two date values
- Invoice’s total - include a $ and proper comma formatting

Use aliases to provide neat, descriptive, user-friendly column names for each column. Sort the results by days overdue and invoice total with both in descending order. Including the unpaid invoices the result set should return 57 rows.

Q.2) (30 points)

Develop a stored procedure that will take a state abbreviation as a parameter and list the name of the state, vendor, and total amount still owed on invoices for the vendor with the highest amount still owed in that state, i.e. list the vendor (along with their state and amount owed) with the most money still owed for a given state. (List only vendors to whom a positive amount is still owed; omit vendors to whom no amount is owed). The Canvas assignment includes a link to a script for a partially completed stored procedure called owed_to_state_vendors. Review the code, including the comments, to understand the requirements of the stored procedure. You will complete the stored procedure, using the comments as a guide. Use of a CTE is advised to accomplish this task.

HINT: test your stored procedure using a statement such as:

```
CALL owed_to_state_vendors('CA');
```

Q.3) (35 points)

Create a new table in the ap database using the code below.

```
CREATE TABLE `new_invoice_records` (
     `new_invoice_record_id` int(11) NOT NULL AUTO_INCREMENT,
      `new_invoice_record_text` varchar(255) DEFAULT NULL,
     `new_invoice_record_timestamp` datetime DEFAULT NULL, PRIMARY
     KEY (`new_invoice_record_id`)
) ENGINE=InnoDB
```

Develop a trigger that will execute when a new record is added into the invoice table, and will add to the `new_invoice_records` table a new row with the following values:

1) An auto-incremented ID value into the `new_invoice_records` column,
2) A text message value, "`You have added a new invoice from <vendor name> with an invoice total of $<invoice total value>`", into the `new_invoice_records` column,
3) The timestamp value at the time of the record insertion into the `new_invoice_record_timestamp` column.

The Canvas assignment includes a link to a script for a partially completed trigger called `new_invoice_row`. Review the code, including the comments, to understand the requirements of the trigger. You will complete the trigger, using the comments as a guide.

HINT: test your trigger by INSERTing a new invoice record using a query such as the one below, and then inspecting the `new_invoice_records` table to see if a new row was added to that table, and whether the values on the new row are the values you would expect to see.

```
INSERT INTO invoices
  VALUES
  (118,34,'ZXA-080','2018-02-01',14092.59,0,0,3,'2018-03-01',
  NULL);
```