CSCI 5559 - Database Systems

Spring 2017

Assignment 1 - Part B

Assignment Goals

The main goal of this part of the assignment is to familiarize you with the developing applications using databases. You are provided with schema of a database for the application you considered in Part A of Assignment 1. Your main tasks will be to implement that model (creating tables and implementing restrictions), to populate that database with test data and to develop queries and other database objects like procedures and triggers, to satisfy the proposed requirements. This implementation will be made on MySQL, a popular open-source DBMS. You are provided with a virtual machine with a web application framework developed on PHP running on an Apache Server on Ubuntu Linux. You will extend this web application adding queries and maintenance windows to add, delete and modify the information on the database.

Relational Model for Database

This relational model is based on the restaurant "*De Vie Sains*" real-world application proposed in Part A of Assignment 1.

Employee (empId[N], name[C150], address[C250], zipCode[N] DoB[D], phone[C20], dateOfJoin[D], SSN[C11], worksInAddress[C250])

$\Pi_{\text{worksInAddress}}$ (Employee) $\subseteq \Pi_{\text{address}}$(Restaurant)

$\forall t1, t2 \; / \; t1 \in Employee \wedge t2 \in Employee.\; t1.ssn = t2.ssh \leftrightarrow t1.empId = t2.empId$

Restaurant (address[C250], city[C128], zipCode[N], phone[C20])

Cashier (empId[N], password[C128])

Π_{empId} (Cashier) $\subseteq \Pi_{\text{empId}}$ (Employee)

KitchenStaff (empId[N], position[C250])

Π_{empId} (KitchenStaff) $\subseteq \Pi_{\text{empId}}$ (Employee)

Waiter (empId[N], manager[N])

Π_{empId} (Waiter) $\subseteq \Pi_{\text{empId}}$ (Employee)

Π_{manager} (Waiter) $\subseteq \Pi_{\text{empId}}$ (Waiter)

DinnerTable (address[C250], tableNumber[N], state[C50], chairs[N], waiter[N])

Π_{address} (Table) $\subseteq \Pi_{\text{address}}$ (Restaurant)

Π_{waiter} (Table) $\subseteq \Pi_{\text{waiter}}$ (Waiter)

Customer (name[C150], phone[C20])

Reservation (reservationDateTime[TS], reservationId[N], phone[C20], address[C250], tableNumber[N], arrivalTime[T])

Π_{phone} (Reservation) $\subseteq \Pi_{\text{phone}}$ (Customer)

$\Pi_{\text{address,tableNumber}}$ (Reservation) $\subseteq \Pi_{\text{address,tableNumber}}$ (DinnerTable)

Terminal (brand[C128], model[C128], serialNo[C128], lastInvoideNo[N])

DailyOperation (empId[N], brand[C128], model[C128], serialNo[C128], operation[C1], operDate[D], cash[D5,2])

Π_{empId} (DailyOperation) $\subseteq \Pi_{\text{empId}}$ (Cashier)

$\Pi_{\text{brand,model,serialNo}}$ (DailyOperation) $\subseteq \Pi_{\text{brand,model,serialNo}}$ (Terminal)

Invoice (invNumber[N], discount[D5,2], totalTax[D5,2], invoiceDateTime[TS], empId[N], brand[C128], model[C128], serialNo[C128], operation[C1], operDate[D], address[C250], tableNumber[N], customerPhone[C20])

$\Pi_{\text{empId,brand,model,serialNo,operation,operDate}}$ (Invoice) $\subseteq \Pi_{\text{empId,brand,model,serialNo,operation,operDate}}$ (DailyOperation)

$\Pi_{\text{address,tableNumber}}$ (Invoice) $\subseteq \Pi_{\text{address, tableNumber}}$ (DinnerTable)

$\Pi_{\text{customerPhone}}$ (Invoice) $\subseteq \Pi_{\text{phone}}$ (Customer)

Items (<u>invNumber</u>[N], <u>code[C5]</u>, qty[N])

$\Pi_{\text{invNumber}}(\text{Items}) \subseteq \Pi_{\text{invNumber}}(\text{Invoice})$

$\Pi_{\text{code}}(\text{Items}) \subseteq \Pi_{\text{code}}(\text{MenuItem})$

MenuItem (<u>code[C5]</u>, description[C250], price[D5,2])

<u>Datatype legend:</u>

[N] – Integer	[C*x*] – Varchar of *x* characters	[D*x,y*] – Decimal *x* precision, *y* scale.
[D] – Date	[TS] – Timestamp	[T] – Time

Queries to Implement

A – Data Query Section

A1. For all tables (dinner tables!), list the address, number, state, name, and SSN of the waiter responsible for each one.

A2. List all the values of position for the kitchen staff, as well as the total number of employee in each position.

A3. List all the employees who live in zip codes 80013, 80014, and 80017, including name, address, zip code, SSN and the role they play in the organization (waiter, cashier, kitchen position, etc.). *Tip*: Remember you can set a fix value as a column in the result of a query.

A4. Display customer information for those who made reservations in the last 2 months but did not arrived in time!

A5. Display names and phone numbers (without duplicates) of the customers that arrive at the restaurant in the last ten minutes before the expiration of their reservation.

A6. Display the menu items (description and price) that are never ordered!

A7. List all the invoices generated in the last 15 days, displaying the number, date of issued, the cashier name and the subtotal of the items.

A8. For the last five days, display the daily balance for each terminal; i.e., for each terminal show the cash difference between opening and closing (assuming that terminals open and close once per day).

A9. Generate a report listing the lazy waiters (those that have not helped any customer in a day!) for the last 3 days.

A10. Display the information (including the name) of the manager for the waiters that have helped some of the customers (by reservation) that were helped at least once by the waiter 'John Smith'.

A11. The restaurant located at 1380 Lawrence St, Denver, 80204 wants to increase its sale by distributing coupons among customers who had made at least five reservations in the past six months. Accordingly, the restaurant wants to give a $20 gift to those customers who have spent up to $300 in the last month; $50 to those who have spent up to $500; and $100 for those who have spent more. Provide a report showing the customer name, phone number, and the total money spent in the last month along with the corresponding gift value.

A12. Generate a report that for each waiter shows all the tables assigned to him/her. Also for a table that there is a reservation for today, list the reservation information too. Display the data sorted by the waiter name, table number, and reservation time if available. The output should be as follows:

Waiter	Table	Reservation No	Date & Time	Customer Name	Customer Phone
John (#345)	XXXX #1				
John (#345)	XXXX #34	254c65	02/15 6:30P	Emily	720-458-5555
John (#345)	XXXX #34	185b84	02/15 9:30P	Alice	720-555-5555
Susie (#184)	YYYY #2	5485c5	02/15 9:00P	John	303-548-5555
Susie (#184)	YYYY #6				

B – Data Manipulation Section

B1. Create a form to generate a new reservation. All the needed information should be requested from the user. If the customer is not registered on the database, a customer account should be created with the information provided.

B2. List all the standing reservations (with expiration time in the future). Add a link to delete the selected reservation.

B3. Create a form to update the price of the menu items. The form should receive a price and a percentage value as input. Then, for all the menu items with a price lower than the indicated price value, increase their price with the given percentage. For example, a price value of 25 and a percentage of 10, will increment a 10% to all the menu items with price under of $25.

B4. Implement a database object that automatically updates the last invoice number on the terminal.

Requirements, Deliverables, and More!

How to Implement your Solution:

1. Generate a sql script with all the DDL statements required to create all the objects for the database, with all the restrictions denoted above. **Do not change the names of the tables and/or attributes,** as your code will be tested against our database.
2. Generate test data for the queries. Save a sql script with all the DML statements for loading the database with that data.
3. Application Development:
    a. Modify the proper php script so the application will connect to your database with the user *dbApp* locally. You should review the grant statement on the MySQL documentation.
    b. Modify the proper php page to add all the reports requested in Part A (Data Query Section) of this document.
    c. Add the php pages to implement all the requirements in Part B (Data Manipulation Section) of this document.

- Make sure you review the "Tips and Documentation" Section below.
- You can change the look of the app, but the focus of this assignment is on use of the database and not how "fancy" the pages look!
- Document any assumption you make (which are not on contradiction with the assumptions made in this document).

Deliverable Items:

- Scripts: *.sql* files. One file for each script.
- Code: *.php* files. All the new or modified files.
- PDF document with the sql statements used to implement each of the requirements for each section (Data Query and Data Manipulation) as requested. Additionally, in this file you should include any pertinent documentation or assumption.
- Please include (as a comment) your name (last, first) and student id in all the documents you submit!

Grading:

This assignment will be graded with the rubric posted on canvas over 100 points.

If you do not embed your implementation of a query in the application code, or you embed it but it does not work properly, you will only receive up to 50% of the grade for that query implementation assuming that your query implementation is correct when directly executed on the DBMS.

The assignment will be graded against our environment running on the VM provided; make sure that your solution runs there, otherwise it will not be graded!

Submission Guideline:

You must submit your assignment through Canvas **before the deadline (date published on Canvas)**; late submissions are not accepted. You may submit your assignment multiple times, but only the last submission (before the deadline) will be recorded and graded.

You should download your assignment after submission and check the files to make sure they are not corrupted during the upload. Please make sure that your entire submission is readable

You are highly encouraged to pose your questions on Piazza under the "assignment-1.b" folder. **DO NOT include your codes, diagrams or solutions** in the comments you share on Piazza. Feel free to help other students with general questions. Remember that Piazza is a Student Collaboration Tool.

Tips and Documentation:

In this section, we describe some tools that would make your life easier while developing this assignment. Also, this section includes a description of the VM environment provided.

The application is based on PHP running over Apache2 Server. PHP is an object oriented/structured language. In this webapp, when run, each PHP script (.php file) generates the HTML code that your browser is going to display. So, basically you run the program sequentially and each "echo" prints a portion of the HTML code. The main menu of the application is implemented in the index.php file. Then, links to other files with the parameters (using the GET option) are used to run other parts of the application. Web "forms" with a POST submit are used to request information from the user and process it in another program. These forms and the actions that are associated with them could be implemented in a single file (with a state parameter) or in several files.

To add new functionality to the application you can copy and edit a portion of all the code available for this submission. You are welcome to search for ideas and resolutions in the Internet, but you cannot copy code from the internet or from another student's solution. Doing that will infringe the university policies (honor code) and will be considered plagiarism. We will use any necessary tool to identify any attempt of plagiarism.

Useful tools:

Sublime Text 3

https://www.sublimetext.com/3

Enables php and sql edition with syntax recognition. With this app, you can create a project and add the path where the php/sql files are, so edition will be easier.
Available for: Windows, Linux, Mac OsX

MySQL Workbench

https://www.mysql.com/products/workbench/

GUI for admin and develop on MySQL.
Available for: Windows, Mac OsX, and Some distros of Linux.
Requires login to Oracle Technical Network (free registration) to download.

SSH client

There are several ssh clients that allows to run command remotely against the server. For windows the most used is Putty (http://www.putty.org/)

Linux and Mac users do not need a specific client as is available on terminal (may require to install openssh-client)
Connection:> ssh dba@ip
(more info available through man pages or http://linuxcommand.org/man_pages/ssh1.html)
Default IP on virtualbox should be 192.168.56.101, but check it with *ifconfig –a* command.

SCP Client

SCP Clients allows to upload and download files to a server. Several options available. Command line clients available on Windows (pscp.exe on putty), or Mac and Linux (scp).
Syntax:

copy to server:	scp <*source_file*> user@ip:<*path/file_name*>
copy from server:	scp user@ip:<*path/file_name*> <*target directory*>

For Windows is also available an explorer like application called *WinSCP*
For Mac: Filezilla or Cyberduck, and for Linux a version of Filezilla is available too.

VM Configuration:

Shared Folders

The **www** folder **(/var/www/html**) is shared for the user dba. It is NOT recommended to open the file remotely for edition, but you can use this share folder for copying and overwriting files to update the webapp.

MySQL

MySql server version 5.7 is installed.
Doc page: https://dev.mysql.com/doc/refman/5.7/en/

Apache 2 and PHP version 7 are installed.

PHP documentation at http://php.net/manual/en/index.php

Webserver root home at:	/var/www/html
Apache Configuration:	/etc/apache2/sites-enabled
Apache error log:	/var/log/apache2/error.log

Bash Reference

For those who never work with Linux server before, the following links could help with some basic bash commands, however, for this class the following commands should suffice for completing all the assignment: *ls, cp, mv, rm, cat, vi* or *joe, chmod, scp*

http://www.tldp.org/LDP/gs/node5.html
http://www.comptechdoc.org/os/linux/usersguide/linux_ugbasics.html
https://www.youtube.com/watch?v=I9F0S2ZZUTA

Access Configuration:

Login user **dba** with password **dba**
MySQL admin user **root** also with password **dba** and local access.

Access Web application through http://192.168.56.101/index.php
The VM is configured with two network adapters. The first one (enp0s3) is configured as NAT so the VM can access the internet. The second one (enp0s8) is configured as Host Only, so you can access it from the host. Check the VM settings and the Virtual Box Networks (preferences) for the configuration parameters if you are having any problem with the connectivity.