

Name: _____

Submission file name: _____

(As in earlier assignments, fill in the above information on a hard-copy of this document and submit it at the beginning of class on the due date. You are required to upload your submission **before you submit the hard-copy in class.**)

Overview: This assignment asks you to extend the simple Web application from the previous assignment. (Thus, in addition to what is described here, everything described in the previous assignment must work properly in your submission.) The goal is to gain experience in programming with some additional SQL features, such as triggers, schema modifications, and dynamic SQL. You will need to make some changes to the functions implemented in PHW03. In addition, some new functions (identified by the prefix Function below) must be implemented.

Artificial Keys Earlier we had assumed that names uniquely identify people. We wish to do away with this assumption. That is, we must be able to store information about two people who have identical names (and possibly identical values for all other attributes). To uniquely identify people in this scenario, we have decided to add an integer column called PID (person identifier) to the Contacts table. Modify your implementation of the createTable function to reflect this change. The PID attribute should be system-generated; that is, its value is *not* supplied by the user in the add function. Thus, the add function will not change from the user perspective. However, you must modify all other functions as follows (when applicable): Whenever the earlier specification asks that a name be printed, you must now print the PID right after the name. For example, the output of the searchEmail function will now generate triples of the form (email address, name, PID). Make sure that the PIDs you generate are always unique: There must never be two rows with the same PID in the Contacts table.

Function creditAcct(P, A) This function should add the amount A to the money owed attribute of the person identified by PID P. Note that A may be negative.

Audit Trail We wish to keep an *audit trail* of all changes made to the *money owed* column of the Contacts table. For this purpose, we will use a table called AuditTrail with columns PID, amount, and timestamp. Whenever there is *any change* to the money owed column of the Contacts table, a tuple (p, a, t) is to be inserted into the AuditTrail table, where p is the PID of the affected tuple (in Contacts), a is the amount credited (negative for debits), and t is a timestamp indicating the date and time of the change. In addition to updates of the money owed column, your implementation must also handle the insertion and deletion of tuples: An insertion is equivalent to the money owed column being updated from 0 to

the value inserted; similarly, a deletion is equivalent to an update from the old value to 0. You must *not* assume that the Contacts table is modified only through your application. For example, you must produce the proper audit trail entry when someone uses *SQLPlus* to modify the table. You must implement the following functions:

Function createAuditTrail() This function should create the AuditTrail table described above. You must also take any other actions necessary for implementing the audit trail as described above (create triggers, etc.). This function produces no output.

Function destroyAuditTrail() This function should result in the audit trail table and all its contents being destroyed. This function produces no output.

Function showAuditTrail() This function should display the contents of the AuditTrail table. For each row, you should print the name of the person associated with the PID in that row. Thus, the output tuples are of the form (name, PID, amount, timestamp). The timestamp must be in the default format used by the Unix `date` program (e.g., `Thu Nov 16 03:28:04 EST 2000`). This function will only be invoked when the audit trail table is required to exist.

Schema Modifications We wish to let users customize the Contacts table by adding and removing columns. The following functions must be implemented. You may assume that these functions will be invoked only when the Contacts table is required to exist.

Function addColumn(N, T) This function should extend the Contacts table by adding a column with name N and type T. If Contacts already has column with name N, no changes should be made and the string “column name clash” should be displayed. Otherwise, there should be no output. (However, you are encouraged to print diagnostic information as described in PHW03.) Existing rows in Contacts get a null value for the new column. You may assume that T will be a valid Oracle type specification.

Function listColumns() This function should list the name and type for each column of the (current) Contacts table. The output should be a set of (column name, column type) pairs. The types should be printed in Oracle syntax.

Function setColumn(P, N, V) If the Contacts table currently has a column named N then that column should be set to V for the tuple identified by PID P; otherwise, the string “column does not exists” should be displayed.

Function getColumn(P, N) If the Contacts table currently has a column named N then that column’s value for the tuple identified by PID P should be displayed; otherwise, the string “column does not exists” should be displayed. Null values should be displayed as an empty entry (an empty cell in an HTML table).

Function dropColumn(N) If there is a column named N in Contacts then that column should be removed, else the string “column does not exist” should be displayed. Only columns added using the addColumn function can be dropped.

Reminders You are to implement the functionality described in this assignment *in addition* to that described in PHW03. Thus, everything described in PHW03 must work properly (including proper starting and stopping of the HTTP server, HTML validation, and the other functions). Please be careful when testing your code. In particular, please make sure you do not leave httpd processes running on the class machines. If you need clarifications, please post a message on the class newsgroup.

Submission Follow the submission instructions for PHW03, replacing phw03 with phw04 appropriately. Please make sure that your submission can be uncompressed, unpacked, compiled, and executed properly. Test it carefully!