1 Application Overview

The pizza delivery database you designed for your Homework 1 will now be implemented with a Web interface. It is to keep track of CUSTOMERs, their PREFERENCEs, and for each preference the INGREDIENTs (onions, ham, bacon, etc.). Each customer may have several preferences. Each preference CONTAINs one or more ingredients. The pizza store will also deliver soda, water, and beer! The database has to keep track of the DRIVERs delivering, dates, and money collected.

You are free to change the details, as long as the main functions described here are supported. In case of drastic changes, you should talk to us first. The project description is intentionally somewhat vague; you are free to decide what kind of interfaces to support etc.

2 Application Details

2.1 Data

A brief description of the data that needs to be stored in the database follows. You may need to augment this with additional information/tables to implement the tasks.

1. **Customers**: ID, name (first, last), address, account info such as amount spent, method of payment (credit card, check, or cash). If credit card, then card number.

2. **Pizza**: Pizza Sizes (small, medium, large), ingredients (peperoni, sausage, mushrooms, onions, extra cheese, etc.), price data, happy hour prices, thin or thick crust, etc. Use your imagination for the ingredients or get it from an online pizza place.

3. **Drinks**: product ID, description, manufacturer, supplier, price, sales tax (6% for MD address delivery, 2.5% for VA).

4. **Delivery people**: personal info, such as ID, name, address, work hours, salary (hourly), transportation info, vehicle, cost per mile, current location or route, etc.

5. **Transactions**: All the history of purchases through the system, records of pizzas with their ingredient purchased including price and delivery dates.

2.2 Functions to be supported

There are several sets of tasks/queries that you need to support. The main tasks are provided below. You may choose to add and/or substitute other tasks depending on the emphasis of your project.

2.2.1 Basic Web Interface

1. **Customer Set up**: This task permits the customer to enter his/her data. The data is analyzed off-line, delivery address is assigned to a main route, and his/her distance is computed. Integrating this task with a web map program to obtain distances will win you some extra credit. Otherwise you are allowed to simplify the task of computing the distances (simplest option might be generate a distance value randomly).
2. **Customer Data Update:** This task permits the customer to update some of his/her data, i.e. address, credit card, etc.

3. **De-activate a customer:** The customer may de-activate his/her account. This task does not delete the customer. Just archives the relevant data.

4. **Re-activate a de-activated customer:** Reinstate the customer for delivery.

5. **Order Placement:** The customer connects and places his order. It should provide a menu with the pizza size, ingredients, drinks, prices, number of each item ordered, and method of payment. This function must provide a listing of prior orders which can be easily repeated or modified before reordering.

6. **Order Fulfillment:** When an order is accepted, a “dispatch ticket” and a “bill” are generated for it.

7. **Receive payment:** A payment has been received from a customer, the driver logs in the payment and time of the delivery.

8. **Happy Hour:** From time to time, the stores makes a 2 for 1 Happy Hour on some pizzas (only). This should be displayed in the ordering form and appropriately computed in the bill.

### 2.2.2 Queries

You have to accommodate several queries that you think are appropriate including (at least the following):

1. **Order Query:** Customer requests to see his/her order(s) and the expected delivery time.

2. **Customer Transaction History:** The store manager requests to see a customer’s transactions during a specified time period.

### 2.2.3 Statistical Queries

If the description below is unclear, feel free to make assumptions. As long as the basic functionality is supported, minor variations in the specifics are acceptable.

1. **Time-Window Revenue:** One of the most important statistical information for the store manager is the cash flow at different times: hour, day, week, etc. This report should be generated on the fly for a period specified by the store manager and it should provide the gross for each product category (pizza and drinks). Furthermore, the report should include similar figures for the same time period of the previous day, week, month, and year. *Note, if the time-window is larger than 24 hours, the previous day is not meaningful. Similarly, if it is larger than a week, the previous week has no meaning.*

2. **Happy Hour Analysis Report:** This report will compare statistics of each happy hour offered with the corresponding hours of the days it was not offered for a period of two weeks, one week before and one week after the happy hour.

3. **Preferred Customers Report:** This report generates a list of the top customers (in terms of revenue), how often they order in the past 30 days, the amount of money per category collected from them, and the average amount collected per delivery.

4. **Inactive Customers Report:** The system generates a list of all inactive customers.
2.2.4 A stand-alone program for XML report generation

Some tasks are better done using stand alone programs that sit atop the database system. These include "data ingest", i.e., data that is arriving from external sources and needs to be integrated into the database. Similarly, if data needs to be sent to a different location (say the Headquarters), an external program may be needed to convert the data to a common, most often XML, format.

Your goal here is to write a simple program that would generate a monthly XML report of the total sales at the store. Here is an example format:

```xml
<report storeid='xyz' month='January'>
  <date id='01/01/2012'>
    <sales>1000.00</sales>
  </date>
  <date id='01/02/2012'>
    <sales>1200.00</sales>
  </date>
  ...
</report>
```

The program should take as input the month and the year, should connect to the database to fetch the relevant data, and then generate the report as shown here. The program is perhaps best written in Java, but other languages like Ruby, Perl, PHP also support connecting to the database.

3 Project Details and Documentation

- **Groups**: The project is to be done in groups of 2 students. The groups are “self-policing” (e.g., each group is responsible for its own division of labor, scheduling, etc.).

- **Assumptions**: In cases where the above description of the application is incomplete, it is acceptable to make assumptions about the application providing that: 1) they are explicitly stated in the report, 2) they don’t conflict with any of the requirements specified above, and 3) they are “reasonable”. If you have a question about the acceptability of any of your assumptions, check with the professor or TA.

- **Reports**: A report should be handed in at the end of each phase (Due dates below).

- **Implementation**: The final phase of the project requires a working implementation of the system to be built, tested, and demonstrated. A large part of the project grade depends on the quality of this implementation. The implementation will be done as a client-server system in which a web server runs on your cluster UNIX account, accepts web queries, and connects to the Oracle DBMS to retrieve the data (you are free to use other tools).

4 Project Phases and Deliverables

4.1 Phase 0 (Due March 1, 2012):

Group membership information due. Send an email to the TAs listing the two members and cc’ing both of them.

4.2 Phase I (Due March 29, 2012):

- **Document**: A document should be handed in for corrections/suggestions at the end of the first phase. It will be graded but you will not see these grade. You will get the grade for the whole project only
at the completion of the project. You will get comments and individualized instructions on how to proceed. The first phase is equally important to the second but it is worth very little if the project is not completed.

The report must contain:

1. Assumptions you have made the enterprise, or any additions/changes made to the specs.
2. The conceptual graphical schema of the database (E-R model), including a list of the attributes for each entity and relationship.
3. The relational schema obtained by converting E/R to relations, and their BCNF or 3NF forms. Make sure you define all keys primary and foreign.
4. Integrity constraints and how they are to be handled (DDL, triggers, assertions).
5. A description of how you plan to populate the tables.

• In addition, you have to implement and demonstrate a toy end-to-end application that uses the database. The sole purpose of this is to familiarize yourself with how to build such an application. E.g. you could populate the database with the data from the SQL assignment, and the webpage could provide a text box for typing in a Customer name (like Google search box), and when the “submit” button is clicked, the results of “select * from states where statecode = ‘AB’ ” would be displayed (where ‘AB’ is the statecode entered in the text box).

Details on how to “submit” this part will be announced later.

4.3 Phase 2 (Due May 1, 2012):

The second phase of the project requires a working implementation of the system to be built, tested, and demonstrated. A large part of the project grade depends on the quality of this implementation. The implementation can be done using ORACLE as a backend database server and an Apache server to provide the Web interface to the database. Alternative system implementations could be acceptable but they require an approval and, in this case, you will have to assume NO support of any kind from neither the TA nor the OIT people.

A portion of the project grade will be based on the user interface and the quality of error handling of data entry errors.

Deliverables for the second phase include:

1. Phase I Document with any revisions made to the relational schema definition from Phase I.
2. A description of the problems encountered in this phase and justification for the solutions.
3. The documentation produced in this phase, i.e.:
   (a) a users manual for the system.
   (b) a description of the system’s limitations and the possibilities for improvements.
4. A demo of the system is required. Details to be announced later.

5 Due Dates

<table>
<thead>
<tr>
<th>Deliverable</th>
<th>Due Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group membership due</td>
<td>3/1</td>
</tr>
<tr>
<td>Phase I due</td>
<td>3/29</td>
</tr>
<tr>
<td>Phase II due</td>
<td>5/1</td>
</tr>
</tbody>
</table>