COP 5725 – Principles of Database Management Systems Professor Shu-Ching Chen

CIS 5725 - Fall 2015

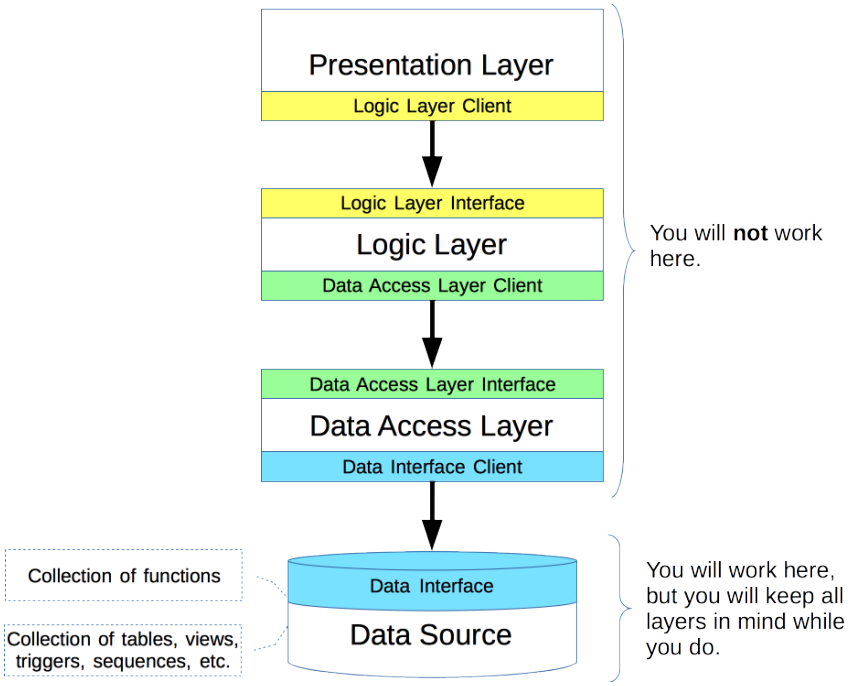
e-Commerce Website Database and Data Interface

Term Project

Database
The database contains most of the information used by the web application. A database is a collection of related data in the form of schemas, tables, views, stored procedures, triggers, indexes, and other objects.

Data Interface
The data interface is the point of contact between the application data access layer and the database. Similar to the software package interfaces you already know, the data interface provides a pre-defined set of functions that will serve as the gateway to the data from the application's data access layer.

COP 5725 – Principles of Database Management Systems Professor Shu-Ching Chen



Universe of Discourse

A database represents some aspect of the real word called the miniworld or the universe of discourse. You will design a database based on the description of the miniworld contained in this section. The subject matter of this project was selected to increase the likelihood of students being familiar with its main concepts and make it easier to find relevant information online to guide the design decisions; however, when explicit requirements given in this description contradict information you find from other sources regarding this subject matter, your solution must satisfy the requirements given here. You must explicitly state and justify all assumptions that are not part of this description.

Your task consists of the design of the database and data interface of an e-commerce website. As such, you will deal with customers, suppliers, products, product categories, carriers, orders, shipments, returns, exchanges, payments, and refunds.

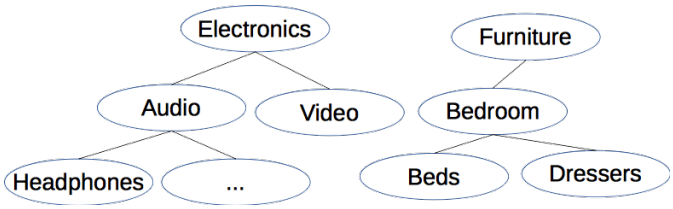
You will need to store information on the full name of your customers and one or more addresses to be used for shipping and billing. Each customer may have one or more credit cards on file, each one associated with a billing address. You may not duplicate addresses in the database, although the same

COP 5725 – Principles of Database Management Systems Professor Shu-Ching Chen

address may be used for multiple purposes (e.g. the billing address of the card used to pay for an order and the shipping address of the order). You must allow customers to specify their preferred shipping address and payment method. In order to communicate with your customers, you must store their e-mail address and have the option of adding their phone number. You will control access to customers' accounts with credentials consisting of username and password. The username will be the customer's e-mail address and the password will be stored encrypted in the database. You must allow customers to change their e-mail address (and hence, username) and password at any time. Each customer will have a non-negative store credit balance. This balance can be topped-up by the customer or increased by the company as a result of a refund.

The company running the e-commerce business does not manufacture any products. Instead, it purchases all products from suppliers. In order to expedite orders, the company maintains a stock of all the products it offers for sale. You need to keep a list of suppliers and store the company name, business address, sales representative's contact information, and discount percentage (i.e., a fixed percent discount applied to all purchases from this supplier). Each product must have at least one supplier. Each supplier in the database must supply at least one product. Each supplier may supply products at different prices.

All products must have a name, description, picture, and price. Each product must be associated with exactly one category. You will design a way of storing trees of categories. Even though you should think of the categories hierarchy as trees, you will store them as a table in the database. Populate the table with a set of categories of your choice. Your chosen category hierarchy must be at least three levels deep. Products may be associated with any category, not just with the leaves of your trees. The following are two very basic trees:



Customers will add products to the shopping cart and you must store the contents of the cart so that it can be preserved across visits. Each order must be associated with exactly one debit/credit card. You can assume that cards are valid and have enough funds to cover the transaction. Store credit balances are automatically applied to payments when an item ships; only the outstanding balance after using up the store credit is charged to the customer's card. Since customers can update or delete cards and addresses on their accounts, you must store a copy of the payment and shipping information for each order so that the system can generate invoices in the future (you will not model invoices, though). It

must be possible to determine the amount of store credit and card charge used to pay for an item. When a customer completes the purchase of the items in the shopping cart, those items must be cleared from the cart. The company inventory is updated only when an item is shipped; however, you must keep track of the items ordered but not yet shipped so that you don't allow any item to be oversold. Since product pricing may change at any time, you must store the price of each product at the time the order is placed so you know what price to charge when the item is shipped. Each product will have a low inventory threshold. When shipping an order brings the count of a product below the threshold, a restocking reminder must be generated by the database.

Restocking reminders are used by the company to place orders with suppliers. You will not model orders to suppliers beyond the details of restocking reminders. A restocking reminder must have information about the product version and the best supplier to order from. The system will determine the best supplier at the time of creation of the reminder by comparing the final cost of the item to the company taking into account the supplier's price for the item and the arranged discount, if any.

A customer may return any part of an order or the entire order within 30 days of placing the order. You must create and store each return request with its associated unique Return Merchandize Authorization code and status (initiated, denied, completed). We will use the simplifying assumption that all accepted return items can be re-sold immediately. When a return is complete, you need to issue a refund (to the customer's store credit balance), record the refund transaction, and update the store's inventory.

A customer may also exchange an item for one of equal or less price (you are not required to model or check restrictions on eligible items for exchanges beyond their price). If the new item costs less than the original item, you must refund the difference to the customer's store credit and store the transaction for future reference. You must update the store inventory accordingly.
Once an order is shipped, it can no longer be modified for any reason. Returns and exchanges are related to orders, but are recorded separately. Customers can change an order before it is shipped and this does not constitute an exchange. Changes to unshipped orders are reflected directly in the original order. Given the possibility of changes, you may not charge the customer's account or update the store inventory until the order ships. We will use the simplifying assumption that packages arrive immediately after shipping.

The e-commerce company hires different carriers to deliver orders to customers. Not all carriers ship all kinds of products to all destinations. You must at least model the following services offered by carriers: (a) international/domestic (all US territories) shipping, (b) hazardous/non-hazardous materials. All items in an order must be shipped to a single address; however, the order may be shipped in multiple packages using multiple carriers if it contains multiple items. Some products may require separate shipment due to their size or weight. Each product must indicate whether it contains hazardous

COP 5725 – Principles of Database Management Systems Professor Shu-Ching Chen

materials and whether it can be shipped with other products. You must store all shipments associated with each order and include information about the carrier and ship date. When an order is fulfilled in multiple shipments, all shipments must be marked as being "partial." In your design, it must be possible to find out which shipment contained any item of any order.

Data Access Interface

For each one of the entities in the miniworld, you must create all the appropriate maintenance functions depending on the allowable actions over the given entity. For instance, customers may be created, modified, and deleted; therefore, you must create the following functions:
create_customer(…), create_product(…),create_manufactor(…), create_company(…), …
update_customer(…), update_product(…), update_manufactor(…), update_company(…), …
delete_customer(…), delete_product(…), delete_manufactor(…), delete_company(…), …
Identifying the appropriate arguments to each of these functions is part of your design job.
Every entity must have an ID. All IDs must be automatically drawn from a sequence, which you must create explicitly. All create_* functions must return the ID of the entity that was created.

You must at least create two functions to retrieve entities. One will return all entities of a type (e.g., all customers) and the other will retrieve a specific one by ID. You should create additional functions to retrieve entities using different combinations of attributes as needed.
retrieve_customers(id)
retrieve_custormers_all()

In general, information about the items that make up an order is spread across several tables. An item may have been shipped, exchanged and shipped again, and finally returned. In order to ease the querying of order items, you must create a view with the information you already have about items in an order plus and additional "status". The possible values of this field and their definitions are given below:

1. Pending: The order containing this item was placed, but the item has not shipped yet.
2. Shipped: A shipment containing this item has been created.
3. Pending Exchange Shipment: An exchange for this item has been created, but the item has not been shipped.
4. Exchanged: Both an exchange and a new shipment have been created for this item.
5. Returned: A return for this item has been created.

In addition to the maintenance functions specified above, you must create the following functions in order to meet the project's requirements:

COP 5725 – Principles of Database Management Systems Professor Shu-Ching Chen

1. login: Provide the user name and password, check if the credential is correct and return success or fail message.
2. update_password: Provide the user name and original password, check if it is valid and get the newly password to replace the original one. There should be a return message to confirm the update.
3. set_preferred_shipping_address: Mark one of the shipping address record as preferred
4. set_preferred_payment_method: Mark one of the payment method as preferred
5. get_availability (takes into account inventory and unshipped orders): Provide the product id to check how many items available to be sold.
6. add_to_cart: Provide the product info and add it into cart of the specific user.
7. remove_from_cart: Provide the product info and remove it from the cart of the specific user.
8. place_order: Provide the product info and user info to create a new order info.
9. charge_customer: Provide the price info and user payment info to do the charge process.
10. ship_item: This function must charge the customer, adjust the inventory, and create a shipment as a single transaction, which you must create explicitly.
11. ship_order: Similar to ship_item, but the shipment contains all items in an order and the shipment is not marked as partial.
12. update_inventory (will increase or decrease the product count depending on sign of argument): provide the product id and other needed info to update the inventory and return message to confim.
13. top-up_store_credit: Provide the user id to check the store credit and to use it.
14. get_products_by_category: Provide the category id to retrieve the products with the same category.

You should be able to demonstrate your data interface by providing a sequence of functions to fulfill each use case. For example: Start from the user login procedure, view some products and add some of them into the user's cart, check out the cart and get the shipment information.
You must also create a trigger to call the function that creates a restocking reminder when updating the inventory for a product brings its count below the low inventory threshold.

COP 5725 – Principles of Database Management Systems Professor Shu-Ching Chen

Deliverables

1) Entity Functions: Turn in as one sql file for each entity.
   a) Create function
   b) Update function
   c) Delete function

2) Retrieve Functions: Turn in as one sql file for each entity
   a) Retrieve all
   b) Retrieve by ID

3) Operation functions: Turn in as 14 sql files. (Demo is required)
   a) The above-mentioned 14 functions

4) Restocking reminder trigger and function (Demo is required)

Submission Due date: Dec. 1, 2015 (Tuesday)
Demo Due date: Dec. 1, 2015 (Tuesday) and Dec. 3, 2015 (Thursday)

Demonstration

For all the students, you will need to demonstrate your web interface to the TA at SCIS Grad Lab (ECS 252) during

- **December 1, 2015 (Tuesday) 2PM ~ 5 PM. (10 min / person)**
- **December 3, 2015 (Thursday) 2PM ~ 7 PM. (10 min / person)**

Please sign your name for one time slot via the following link: http://ppt.cc/MxPNp

Submission

Submit your compressed zip file with the format **Deliverable#_YourName_Fall2015** as the filename and upload it to Moodle system.