EDA216 - Database Technology - Project Report

Lund University, Faculty of Engineering, Computer Science
Jakob Berglund (ada10jbe@student.lu.se)
Filip Lindqvist (ada10fl2@student.lu.se)
Richard Simko (ada09rsi@student.lu.se)

March 2014

Introduction

This is an implementation of a computerised system for handling production and delivery of cookies produced by the fictional company *Krusty Kookies Sweden AB*. This pilot implementation will allow the company to create and manage pallet where cookies are packed upon. In future implementation the system can be extended to handle customers, orders, poduction and deliveries.

Requirements

These are the requirements presented by the customer. All of the requirements are implemented, except when explicitly stated.

Production

A pallet is considered to be produced when the pallet label is read at the entrance to the deep- freeze storage. (*Since no reader is available the pallet is considered to be produced when created in the UI*) The pallet number, product name, and date and time of production is registered in the database. The pallet number is unique.

At any time, we must be able to check how many pallets of a product that have been produced during a specific time.

Raw materials

When a pallet is produced, the raw materials storage must be updated. We must be able to check the amount in store of each ingredient, and to see when, and how much of, an ingredient was last delivered into storage. (*UI not implemented as it was not part of the assignment, the update of raw materials in storage is done however*).

Recipes

We need an interface to the collection of recipes, where we can study and update recipes. We also need a facility for entering new recipes. We don't change recipes during production. (*UI not implemented as it was not part of the assignment*)

Produced Pallets

As we mentioned earlier, pallets in the deep-freeze storage may be blocked. An order to block a pallet will always come before the pallet has been delivered. This is due to the new investments in our laboratory, where the analysis process is completely automated.

We must be able to trace each pallet. For instance, we need to see all information about a pallet with a given number (the contents of the pallet, the location of the pallet, if the pallet is delivered and in that case to whom, etc.).

We must also be able to see which pallets that contain a certain product and which pallets that have been produced during a certain time interval. Blocked products are of special interest. We need to find out which products that are blocked, and also which pallets that contain a certain blocked product.

Finally, we must be able to check which pallets that have been delivered to a given customer, and the date and time of delivery.

Orders and Production Planning

Orders must be registered in the database. For production planning purposes, we must have a facility to see all orders that are to be delivered during a specific time period. (*UI not implemented as it was not part of the assignment*)

The production planning is manual. At the end of each week, production for the following week is planned, using the orders for the following weeks as input. We cannot produce "on demand", since it takes time to set up a production line for a new kind of cookie (mixers have to be cleaned, for example).

Delivery

Before pallets are loaded into the freezer trucks, a loading order is created. (*UI not implemented as it was not part of the assignment*) The order contains information regarding the customers and the number of pallets to be delivered.

When pallets are taken out of deep-freeze storage the pallet label is read. When the truck is loaded, the driver receives a loading bill (identical to the loading order, but contains a field where the customer can acknowledge reception of the delivery). The loading bill data need not be saved in the database. (*UI not implemented as it was not part of the assignment*)

When the loading bill has been printed, the data regarding delivered pallets must be updated with customer data and date of delivery.

System Outline

The system was developed using PHP and its integrated PDO connection for communicating with a MySQL database.

The main logic of the system is gathered in `database.inc`. This class handles the entire connection to the database, providing the views with PHP-formatted data to display in the different pages on the site.

The site contains functionality for listing and searching for pallets (`index.php`), viewing a specific pallet (`showpallet.php`), creating a new pallet (`createpallet.php`), and blocking a pallet (`block.php`). All of these except for `block.php` corresponds to a view on the site. `block.php` only handles the form submission after the user clicks the block button on `index.php`.

Database Design

E/R Diagram

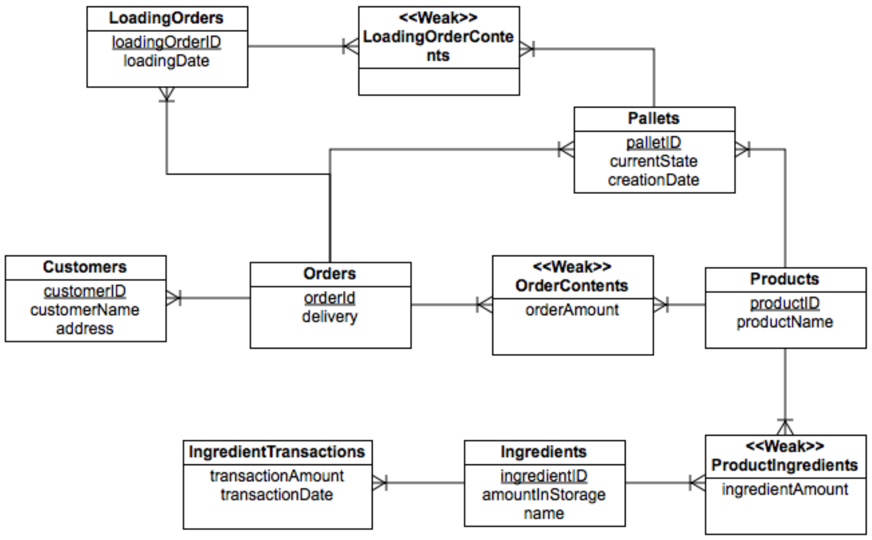

Figure 1: Entity/Relation diagram

Relations

Customers(customerID, customerName, address)
Orders(orderID, delivery, *customerID*)
Products(productID, productName)
OrderContents(*orderID*, *productID*, orderAmount)
Ingredients(ingredientID, amountInStorage, name)
ProductIngredients(*productID*, *ingredientID*, ingredientAmount)
IngredientTransactions(*ingredientID*,transactionAmount,transactionDate)
Pallets(palletID,*productID*,*orderID*,currentState,creationDate)
LoadingOrders(loadingOrderlD,*orderID*,loadingDate)
LoadingOrderContents(*loadingOrderlD*,*palletID*)

Since there are no functional dependencies except for except for the key dependencies, all relations are in BCNF.

SQL implementation

```sql
/* ----------------- ORDERs ----------------- */
create table customers (
       customerID INT NOT NULL AUTO_INCREMENT,
       customerName VARCHAR(255),
       address VARCHAR(255) ,
       PRIMARY KEY(customerID)
);

create table orders (
       orderID INT NOT NULL AUTO_INCREMENT,
       delivery DATE,
       customerID INT NOT NULL,
       PRIMARY KEY(orderID),
       FOREIGN KEY(customerID) REFERENCES customers(customerID)
);

create table products (
       productID INT NOT NULL AUTO_INCREMENT,
       productName VARCHAR(255),
       PRIMARY KEY(productID)
);

create table orderContents (
       orderID INT NOT NULL,
       orderAmount INT,
       productID INT NOT NULL,
       FOREIGN KEY(orderID) REFERENCES orders(orderID),
       FOREIGN KEY(productID) REFERENCES products(productID)
);

/* ----------------- INGREDIENTS ----------------- */
create table ingredients (
       ingredientID INT NOT NULL AUTO_INCREMENT,
       amountInStorage INT,
       name VARCHAR(255),
       PRIMARY KEY(ingredientID)
);

create table productIngredients (
       productID INT NOT NULL,
       ingredientID INT NOT NULL,
       ingredientAmount INT,
       FOREIGN KEY(productID) REFERENCES products(productID),
       FOREIGN KEY(ingredientID) REFERENCES ingredients(ingredientID),
       PRIMARY KEY(productID, ingredientID)
);
```

```
create table ingredientTransactions (
       ingredientID INT NOT NULL,
       transactionAmount INT,
       transactionDate Date,
       FOREIGN KEY(ingredientID) REFERENCES ingredients(ingredientID)
);

/* ----------------- LOADING ----------------- */
create table pallets (
       palletID INT NOT NULL AUTO_INCREMENT,
       productID INT NOT NULL,
       orderID INT NOT NULL,
       currentState VARCHAR(255),
       creationDate DateTime,
       FOREIGN KEY(productID) REFERENCES products(productID),
       FOREIGN KEY(orderID) REFERENCES orders(orderID),
       PRIMARY KEY(palletID)
);

create table loadingOrders (
       loadingOrderlD INT NOT NULL AUTO_INCREMENT,
       orderID INT NOT NULL,
       loadingDate Date,
       FOREIGN KEY(orderID) REFERENCES orders(orderID),
       PRIMARY KEY(loadingOrderlD)
);

create table loadingOrderContents (
       loadingOrderlD INT NOT NULL,
       palletID INT NOT NULL,
       FOREIGN KEY(loadingOrderlD) REFERENCES
           loadingOrders(loadingOrderlD),
       FOREIGN KEY(palletID) REFERENCES pallets(palletID)
);
```

User's manual

The system is available for use at `http://www.krusty.se`. Great care was taken when desingning the UI to make it as user friendly as possible.

Pallets can be searched for using the filtering methods available on the front page.

Pallets can be created using the "Create pallet" button.

Pallets can be blocked using either the "Block" button for each individual pallet or the "Block all visible pallets" button which will block all pallets currently being displayed in the results.