DB Design Report

For this project, which a simple ecommerce admin site is built, following functions are required to be implemented:

- **Login**:
  - view/update my profile
- **Category**:
  - manage products categories: *create* new, *update*, *delete* categories
- **Products**:
  - *view/update/insert* new products/*remove* out stock products
- **Users**:
  - manage users means can alter user's info including *payment methods*, *shipping address*, *order history*
- **Orders**:
  - *view/update* orders

In order to implement these functions, a detailed, granular and normalized database design is preferred. The following sections will describe the design of the database.

For the purpose of have the database in BCNF, there are 22 tables in total. The main tables are **user**, **product**, **address**, **order**. While the other tables are used to store the information of the main tables and used to implement the *many-to-many* relationship by table junctions.

In this report, only the main tables are referred to, and the other tables are not mentioned as they are implied by the relationship between the main tables. The detailed design can be found in attached sql schema, and ER diagrams.

Three functional groups are considered when design the database:

User Management

The users of the system are divided into two groups: **admin** and **customer**. The admin group is the one who can manage the system, while the customer group is the one who can buy products from the system.

The admin group are simple enough, has login credentials, personal information and a list of services such as product, category, user and order management. Which is a lot of functions to be implemented but not too complicated on the database structure.

The customer group, on the other hand, is more complicated as they are the central part of a ecommerce system. Besides the login credentials and personal information, they also have a list of **payment** methods, shipping **address** and **order** history.

For more modularization, **address** itself is a table with *many-to-many* relationship with customer, in which a customer can have multiple addresses and an address can be used by multiple customers. Since the address is standalone, it can also be used by other tables such as the product inventory and shipping address.

Payment details, is an *one-to-many* relationship with customer, as a customer may have multiple payment methods but a payment method can only be used by one customer.

Product Management

Products, as the core of the ecommerce system, are the most complicated part of the database design. During the shooping process, a product will be moving through at least three different parties: the *vendor*, the *logistic company* and the *customer*. Each party sees the product in a different way. To the *vendor* and *customer*, each product has a price, and can be searched, filtered or sorted by categories, keywords, price and tags.

Products are stored in *inventory*, which is like a warehouse, it has an address, and associated management staff. A product can be stored in multiple inventories, and an inventory can store multiple products. This design would simplify the process of moving products from one party to another.

Category

A product would fall under one category, which might be a sub-category of another category. Each category would have a name, description, possibly a parent category and a list of sub-categories. The category table is a *one-to-many* relationship with itself, in which a category can have multiple sub-categories and a sub-category can only have one parent category. The category table is also a *one-to-many* relationship with product, in which a product can be under exactly one category and a category can have multiple products.

However, to simply the user experience, a product would also have a list of **tags**, which in turn can be looked up by associated **keywords**. A product can have multiple tags, and a tag can be associated with multiple products. A keyword can be associated with multiple tags, and a tag can be associated with multiple keywords. By searching one or more keywords, the system can quickly find products that the user is interested in.

Shopping Process

The shopping process can be divided into two parts depending on whether it is completed or not. The first parts involves customer browsing the products and adding products to the shopping cart. The second part involves customer checking out the products in the shopping cart and vendor shipping the products to the customer.

Shopping Cart

One user can have at most one shopping cart at a time, although the content of the shopping cart are changed frequently. Each entry in the shopping cart is a product referenced by its id, and its quantity and price. The shopping cart would also have a total price, which is the sum of the price of all products in the shopping cart. The shopping cart is a *one-to-many* relationship with product, in which a product can be in at most one shopping cart and a shopping cart can have multiple products. A shopping cart has a *one-to-one* relationship with the customer.

Order

When user checks out the shopping cart, the selected products are evicted from the cart and stored in the **order** table. The order table is similar to the shopping cart table, it would two prices, *amout due* and *amount paied*, with associated payment detail. Also the order has *one-to-many* relationship with the customer that a customer can have multiple orders and an order can only be placed by one customer.

Products within an order need to be shipped to the customer, which contains information of which inventory this shipment is from, and the customer's address that the product will be shipped to. In an order, there might be multiple shipments as different products in the same order might be stored and shipped from different inventories. The shipment would also have details of logistics company and the tracking number.