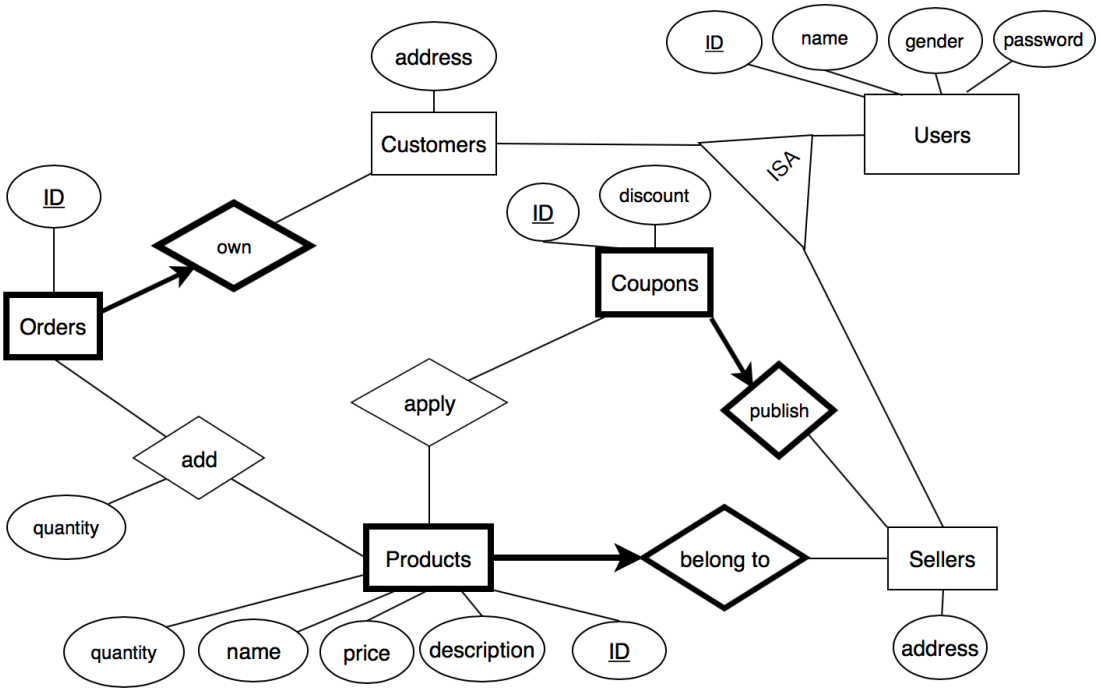
Final Description

1. ER diagram



2. Database Table Design

comments:
Application: Realization in web server while not in database design
Underline: Primary Key

Tables:
users (user_id, name, gender, password) (application: name is unique)
customers (user_id, address)
sellers (user_id, address)
products (product_id, name, description, seller_id, price, quantity)
coupons (coupon_id, discount, seller_id)
orders (order_id, customer_id)
coupon_applied (coupon_id, product_id)
orders_products (order_id, product_id, quantity)

为什么选择单独列出 table customesr 和 sellers:
方便代码拓展

为什么增加 order_products:
重新定义需求后，一个订单可以有多类产品， 一类产品也可以在多个订单中

3. Details in Application

Overview:

如何生成并 insert 是 primary key 的 id：
因为 id 应是 server 分配而不是用户输入，为方便起见，统一规定每次 insert 一条新的记录, 插入的 primary key 应为当前最大的 id + 1.
例如数据库中已有 id 为 1,2,3 的 user, 则插入的新 user_id 应为 4. 其他也是如此。如果例如出现 1,3 的情况，因为数据库较小，不会影响数据库性能，此优化暂时不作考虑。

如何在设计页面逻辑时获取用户登录的个人信息（名字等）：
储存在全局变量 session 中。从一个人 login 成功到一个人 logout 这个时间段称为一个 session. Flask 提供了全局变量 session（本质是一个 dictionary）来储存用户的个人登录信息。

应充分考虑用户输入出错的情况，如注册的姓名已经存在等，需要在页面上显示出错误信息。以下业务流程均为常规正确流程，用户输入出错情况每个用例都需单独考虑。

应充分考虑页面之间的跳转，如卖家添加产品到一半突然不想添加了，需要一个按钮可以直接返回卖家主界面。

url 和 html 文件命名规范: 使用_代替空格，空格容易出问题

sign up：
input on web: name, gender, password, select customer / seller, address
Insert: (user_id, name, gender, password) into users
    if customer: insert (user_id, address) into customers
    if seller insert (user_id, address) into sellers
success: to login page
failure: to sign up page

login:
input on web：name, password, select customer / seller
check name 是否存在，password 和选择是否正确
success: to customer / seller main page
failure: to login page

Customer:

customer main page:
directly list all the available products, price, coupon and let customer select and input quantity

a button: to my order

order main page:
should list all the orders belonging to the customer, let customer select one
a button: to customer main page

order xxx page:
should list all the products and corresponding coupons (don't forget to provide a button for removing product and a blank to update the quantity of a particular product) and provide a button at bottom to calculate the total price of the order
a button: to return to my order main page
a button: pay the order (redirect to purchase success page)

purchase success page:
payment success!
a button: back to customer main page

Seller:

seller main page:
should contain the following buttons:
add product: to add product page
delete product: to delete product page
update product: to update product page
add coupon: to add coupon page
delete coupon: to delete coupon page
update coupon: to update coupon page

add product page:
input: name, description, price, quantity
insert (product_id, name, description, price, quantity, seller_id) into products
a button: back to seller main page

delete product page:
list all the products, quantity and price belonging to the seller first. A button 'delete' for each product.
After click the button, the product should be removed from the displayed list.
a button: back to seller main page

update product page:
list all the the products, quantity and price belonging to the seller first.
Let seller input the name, new price and new quantity.
update table products
a button: back to seller main page

add coupon page:
Similar to update product page.

delete coupon page:
Similar to delete product page.

update coupon page:
Similar to add coupon page.