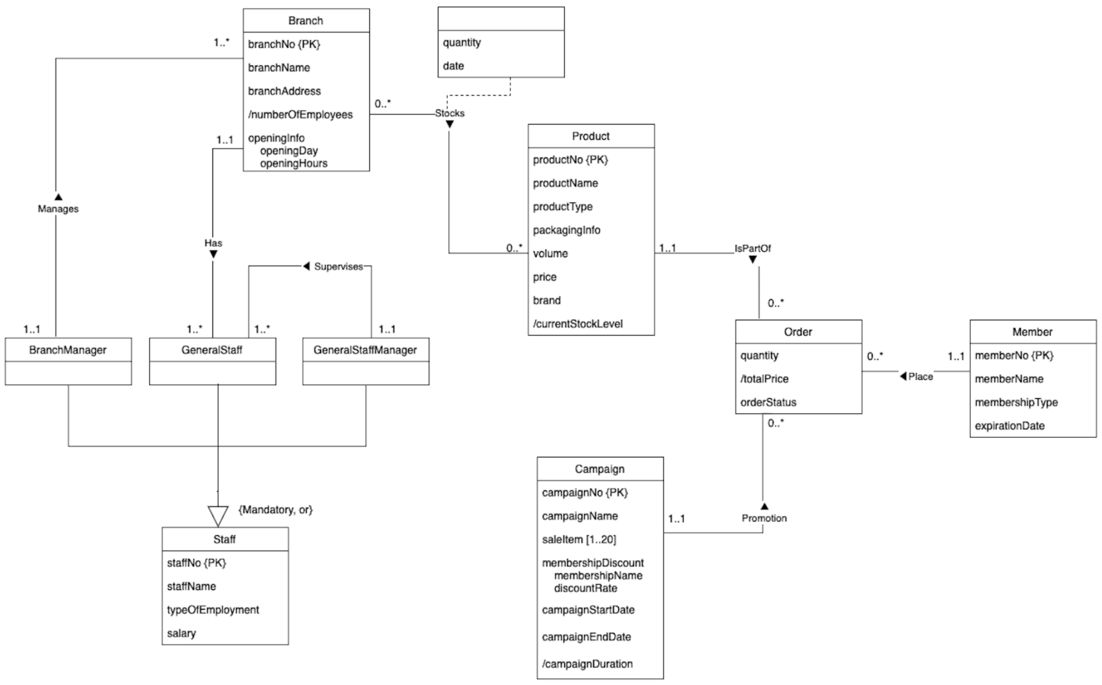
Database Modelling and Design Report

Task 1: Conceptual data model

-- Screenshot of conceptual data model



-- Generalization /specialisation

Staff's roles are specialized into three subclasses from superclass (Staff).

A General Staff is a staff working in each branch, and each branch has one to several General Staffs.

General Staff Manager is a staff who supervises one to several General Staffs.

A branch manager exists in each branch and is a staff who manages the sales and operation of one to several branches.

There are no other kinds of Staff different than these three Staff. Also, each staff member can have only one role. Therefore, it has {mandatory, or} constraint.

-- Attributes on relationship

When the branch is stocking the products, quantity and date are recorded.

-- Assumptions

1. Only the General Staff need to report when something happens, and they report to the "General Staff Manager". The one who “General Staff” should report to will be indicated by Fk derived from GeneralStaffManager (GeneralStaff.staffNo) after the Logical data model transforms in task2.

2. Sale items are limited from 1 to 20 per sale campaign.

3. The total price of the Order entity is calculated by considering the product price, quantity, membership type of the Member (customer), and the discount rate of the Campaign.

Task 2: Logical data model

Product (ProductNo, productName, productType, packagingInfo, volume, price, brand)

Primary key ProductNo
Derived currentStockLevel
Alternate Key productName, brand

Branch (branchNo, branchName, branchAddress, openingDay, openingHours, staffNo)

Primary key branchNo
Foreign key staffNo *references* BranchManager(staffNo)
Derived numberOfEmployees

Campaign (campaignNo, campaignName, membershipName, discountRate, campaignStartDate, campaignEndDate)

Primary key campaignNo
Derived campaignDuration (campaignEndDate – campaignStartDate)

Order (productNo, campaignNo, memberNo, quantity, orderStatus)

Primary key productNo, campaignNo, memberNo
Foreign key productNo *references* Product(productNo)
Foreign key campaignNo *references* Campaign(campaignNo)
Foreign key memberNo *references* Member(memberNo)
Derived totalPrice ((1-Campaign.discountRate)*Product.price*quantity)

BranchManager (staffNo, staffName, typeOfEmployment, salary)

Primary key staffNo

GeneralStaff (staffNo, staffName, typeOfEmployment, salary, branchNo, superviseStaffNo)

Primary key staffNo
Foreign key branchNo *references* Branch(branchNo)
Foreign key superviseStaffNo *references* GeneralStaffManager(staffNo)

GeneralStaffManager(staffNo, staffName, typeOfEmployment, salary)

Primary key staffNo

Stocking (branchNo, productNo,quantity, date)

Primary key branchNo, productNo
Foreign key branchNo *references* Branch(branchNo)
Foreign key productNo *references* Product(productNo)

Member (memberNo, memberName, membershipType, expirationDate)

Primary key memberNo

SaleItem (saleItemNo, campaignNo)

Primary Key saleItemNo
Foreign key campaignNo *references* Campaign(campaignNo)

-- How to translate the conceptual data model into the logical data model

There are few steps that can apply to Task1 CDM for Logical Data Model transform in task2.
Create relation for each strong entity:
Staff, Branch, Product, Member, Campaign
Create relation for Superclass/Subclass:
Since BranchManager, GeneralStaff, and GeneralStaffManger have {mandatory, disjoint} constraint from the Superclass (Staff), get every attributes from the Superclass and discard Staff relation. superviseStaffNo attribute in the GeneralStaff relation indicates the manager that the General Staff should report to.
One-to-many:
The entity on one side's PK is becoming FK of the Many side. Branch relation gets staffNo as FK from BranchManager. GeneralStaff relation gets branchNo as FK from Branch and superviseStaffNo as FK from GeneralStaffManager.
Many-to-many:
A Stocking relation is created by taking the branchNo of the Branch entity and the productNo of the Product entity as FK and using it as a composite PK.
Complex:
An Order relation is created using the productNo of the Product entity, the memberNo of the Member entity, and the campaignNo of the Campaign entity as FK and using it as a composite PK.
Multi-valued:
The saleItem attribute in the Campaign entity becomes a new relation using campaignNo as Fk.