VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

DATABASE SYSTEMS (CO2013)

Assignment 1

FABRIC AGENCY DATABASE

Advisor: PhD. Phan Trọng Nhân
Class: CC02
Group: 5
Students: Đinh Việt Thành - 2152966
Trần Nhật Tân - 2112259
Vũ Châu Duy Quang - 2153730
Trần Bảo Nguyên - 2153637
Dương Trọng Phúc - 2152237

HO CHI MINH CITY, OCTOBER 2023

Contents

1 Member list & Workload

No.	Fullname	Student ID	Percentage of work
1	Đinh Việt Thành	2152966	100%
2	Vũ Châu Duy Quang	2153730	100%
3	Trần Nhật Tân	2112259	100%
4	Trần Bảo Nguyên	2153637	100%
5	Dương Trọng Phúc	2152237	100%

2 Introduction

This is a design of Fabric Agency Database, which has been given by advisor Phan Trọng Nhân. In this report, there are Entity Relational Diagram of this database as well as its Relational Database Schema through mapping. Furthermore, the report also contains some constraints that are not shown in our ERD designation. The description of this database will be demonstrated again for easy tracking in further parts.

FABRIC AGENCY DATABASE

The agency Y supplies the wholesale fabric by bolts for their customers. Each bolt belongs to a specific category such as: silk, khaki, crewel, jacquard, faux silk, and damask. A bolt has a code that is unique within a category, and a length. Each category of fabric has a unique code, name, color, current price(s) (including the price, and the date when that price was made), and quantity (the number of bolts of this category in the warehouse).

The agency takes fabric sources from many suppliers. Each supplier provides many different categories of fabric for the company. However, each category is stemmed from only one supplier. The database needs to store some information about suppliers such as: a unique code, name, address, bank account, tax code, phone number(s). Whenever fabric sources are imported into the warehouse, the quantity of each category, the date, the purchase price must be stored in the database.

A customer has a unique code, name (first and last), address, phone number(s), arrearage (unpaid debt), and partial payments (including the date and amount of money). For example, a customer has $1000 in arrears, he or she is allowed to partially pay for the agency (e.g., he or she pays $200 at the first time, and then $300 for the next, and so on till he or she gets out of debt).

A customer makes an order. Each order contains one or more bolts, and processed by an employee at a specific date and time. An order has a unique code, and a total price. Information about employee consists of a unique code, name (first and last), gender, address, and phone. When a customer makes an order, the system needs to track the order status by time, including "new", "ordered", "partial paid", "full paid", or cancelled". If the order is cancelled, the agency staffs need to input the reason for that cancellation.

Moreover, the agency wants to track the history payment of a customer for each order he or she successfully made. In case the arrearage is over $2000, the system has to put that customer in “warning” mode and alert the agency. If that case stays for more than 6 months, the arrearage is marked as “bad debt”.

The agency has different types of employees: managers, partner staffs, operational staffs, and office staffs. One partner staff will take care of one or more suppliers while one office staff will be in charge of one or more customers. One supplier is taken care by only one partner staff whereas one customer is cared by only one office staff. The operational staff will be in charge of customer order.

Note: ‘Bolt’ is a unit of measurement used as an industry standard for a variety of materials from wood to canvas, typically materials stored in a roll.

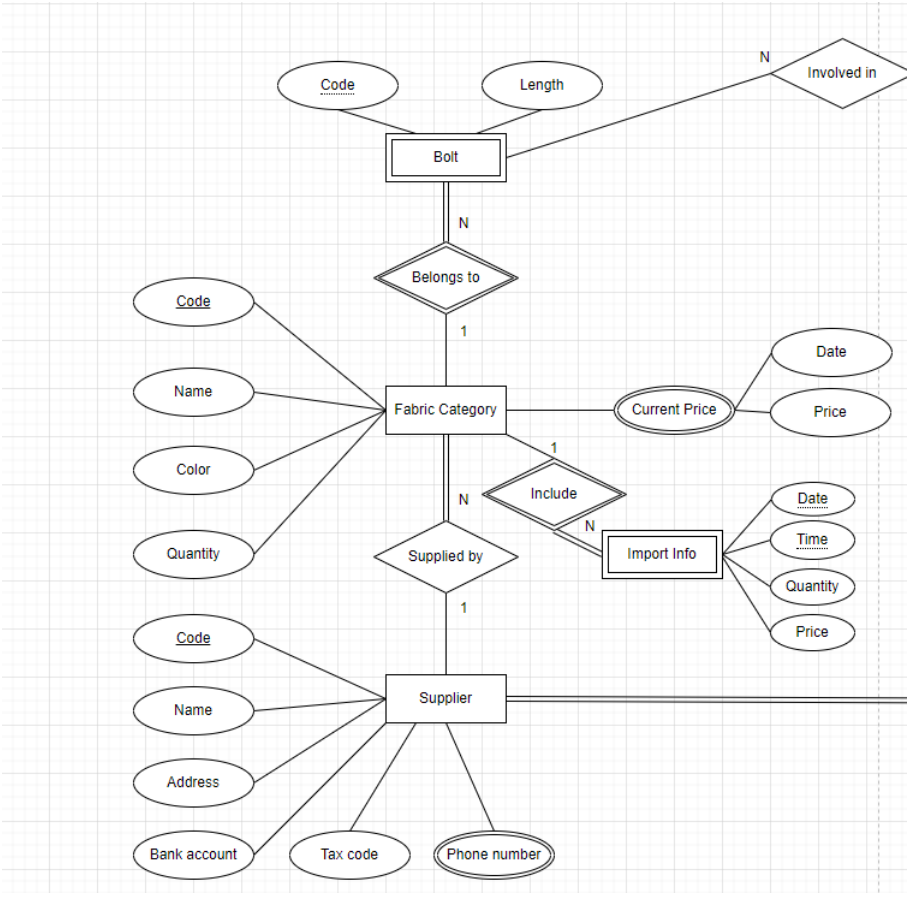
3 Entity-Relationship Diagram

In this part, the Entity Relation Diagram will be demonstrated and its formation will be explained.

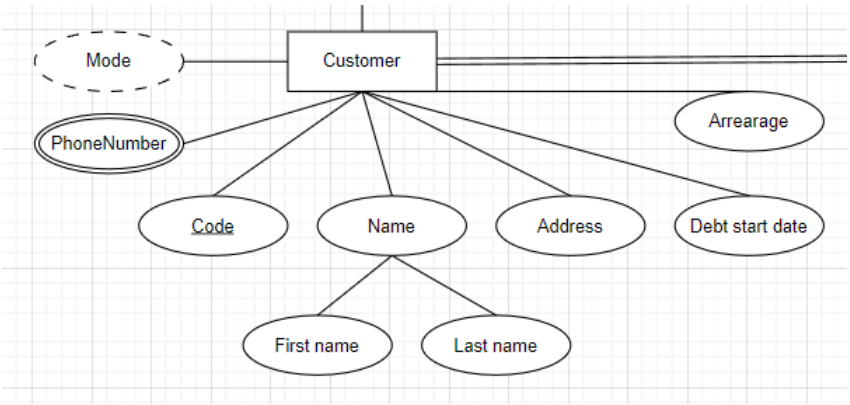
Firstly, **Category** is set as a strong entity with own unique *Code* and other basic attributes: *Name*, *Color*, *Quantity*. The **Current price** is multiple-value including attributes *Date* and *Price*. Meanwhile, Bolts are only unique within a category, therefore, **Bolt** is set as a weak entity belongs to **Category** with the *Code* partial key. The description mentions **Category** is in silk, khaki, etc, so it is possible to create a specialization for **Category**. However, there is not any clear confirmation that **Category** only belongs to those specific category, and the process of creating specialization without any extended attributes tends to be expensive so we decide not to include this specialization in this condition.

Supplier is another strong entity with its basic attributes (*Code* is key and *Phone number* is multiple-value). **Fabric Category** is supplied by **Supplier** so they relate via ”is supplied” relation ship (N:1 with N on Category side base on the description). Also, **Fabric** has to have supplier so it will be mandatory on **Supplier** side.

To track the import information, we consider **Import Info** as another weak entity belongs to **Category** with partial key is *Date* and *Time* (as there can be more than 1 import in a day). Other attributes are *Quantity* and *Price*.

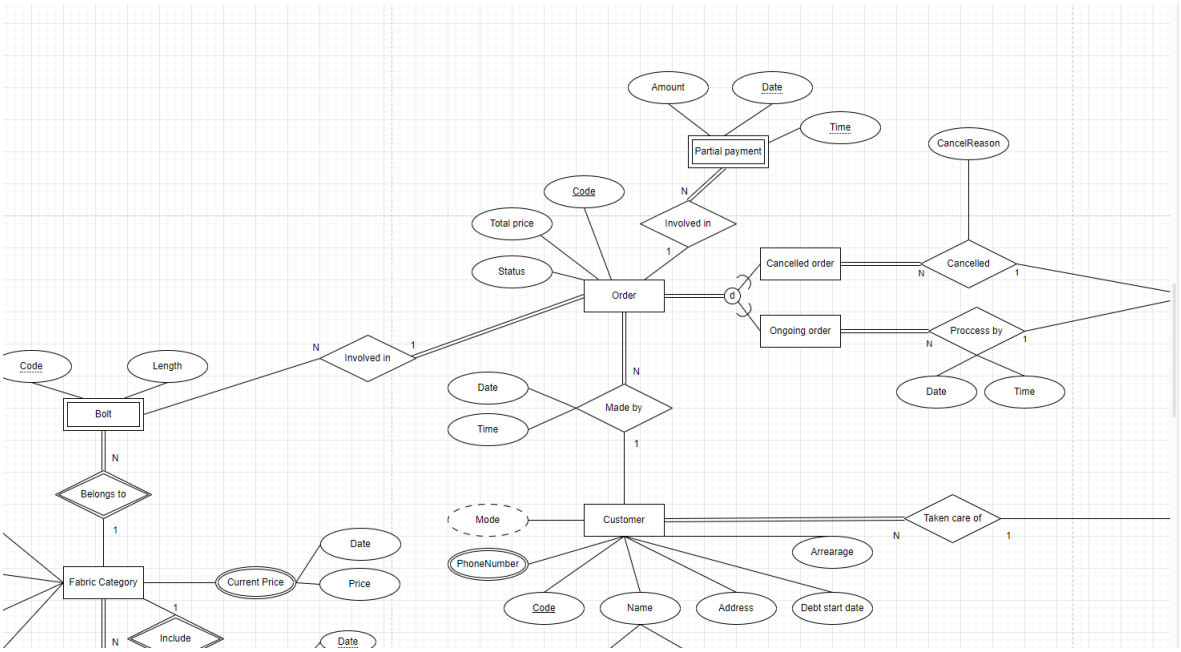


The third strong entity is **Customer** with primary key is **Code**, composite attribute *Name* (*First name* and *Last name*), multiple-value *Phone number*, *Arrearage*, *Address*. We add *Debt start date* attribute for further use. This attribute will be discussed in the Additional Constraints section. *Mode* is an derived attribute base on *Arrearage* and *Debt start date*. *Partial payments* will be discussed later.



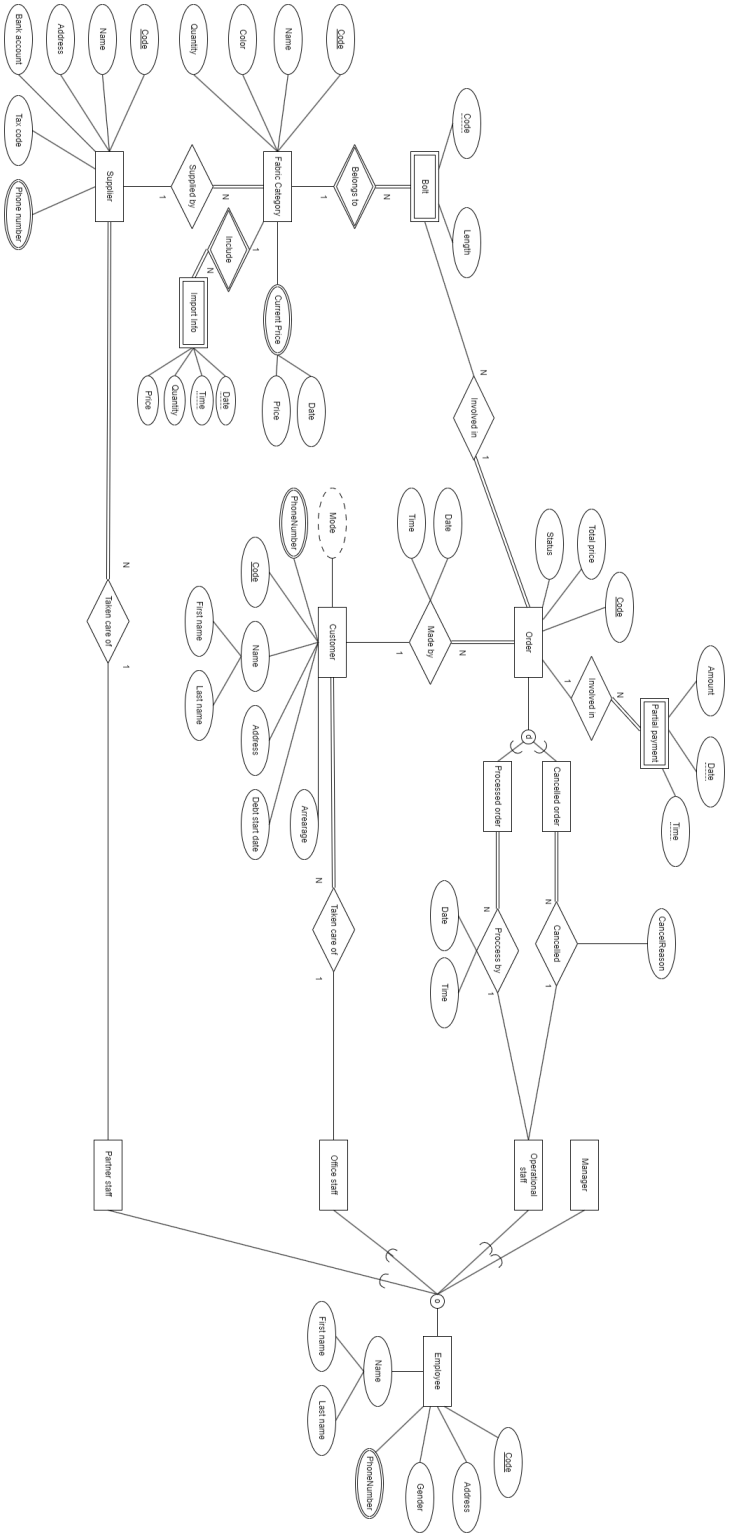
Order is another strong entity, beside basic attributes, we specialize **Order** into **Processed order** and **Cancelled order** (disjoint specialization). Both of these have relation with **Operational staff** (describe later), **Cancelled order** has *Cancel reason* attribute and **Processed order** has process *Date* and *Time* attributes. Each **Order** contains one or more **Bolt** so we create a N:1 relation between them (N on **Bolt** side) and B side is mandatory as there is at least one **Bolt** in an **Order**. Also, **Order** is made by **Customer** so there is another N:1 relation between them (N on **Order** side) and **Order** is mandatory as an **Order** only exists because of a request from **Customer**.

Moreover, the agency wants to track the history payment for each order a customer successfully made so we set the **Partial payment** as a weak entity belongs to **Order** instead of **Customer**.



The last strong entity is **Employee**, with simple attributes as the description mentioned (similar to above so we skip for avoiding repetition). We specialize **Employee** in four group: **Manager**, **Operational staff**, **Office staff** and **Partner staff** (we set it as overlap specialization as no disjoint constraint mentioned in the description).

One **Partner staff** take care at least one **Supplier** and one **Supplier** is taken by only one **Partner staff** so we set the N:1 relation between them (N on **Supplier** side) and **Supplier** side should be mandatory. The relationship between **Customer** and **Office staff** is similar. Here is our whole ERD (see the image file or the draw.io file to see more easily):


Bank account
Address
Name
Code
Tax code
Phone number
Supplier
Supplied by
Quantity
Color
Name
Code
Fabric Category
Belongs to
Code
Bolt
Length
Include
Import info
Price
Quantity
Time
Date
Current Price
Price
Date
Involved in
Taken care of
PhoneNumber
Mode
Code
First name
Name
Last name
Address
Debt start date
Arrearage
Customer
Time
Date
Made by
Status
Total price
Code
Order
Involved in
Partial payment
Amount
Date
Time
Processed order
Cancelled order
Date
Time
Process by
Cancelled
CancelReason
Taken care of
Partner staff
Office staff
Operational staff
Manager
First name
Name
Last name
Employee
PhoneNumber
Gender
Address
Code

4 Relational Database Schema

Major part of the Relational Database Schema is mapped following mapping rules. Therefore, we will try to be as brief as possible.

5 strong entities: **Fabric_Category**, **Supplier**, **Customer**, **Order**, **Employee** are stored in 5 tables with their own basic key and attributes.

- **Bolt** is a weak entity belongs to **Fabric_Category** so **Bolt** has a table with its key is a combination of **Bolt**.`Code` and **Fabric_Category**.`Code` (foreign key).

- Similarly, **Import_Info**'s key composes of `Date`, `Time`, **Fabric_Category**.`Code` (foreign key).

- Similarly, **Partial_Payment** with key include `Date`, `Time`, and **Order**.`Code` (**Order**.`Code` is foreign key).

- **Fabric_Category** and **Supplier** has N:1 relationship with mandatory on **Fabric_Category** side so we put **Supplier**.`Code` as foreign key in **Fabric_Category** table.

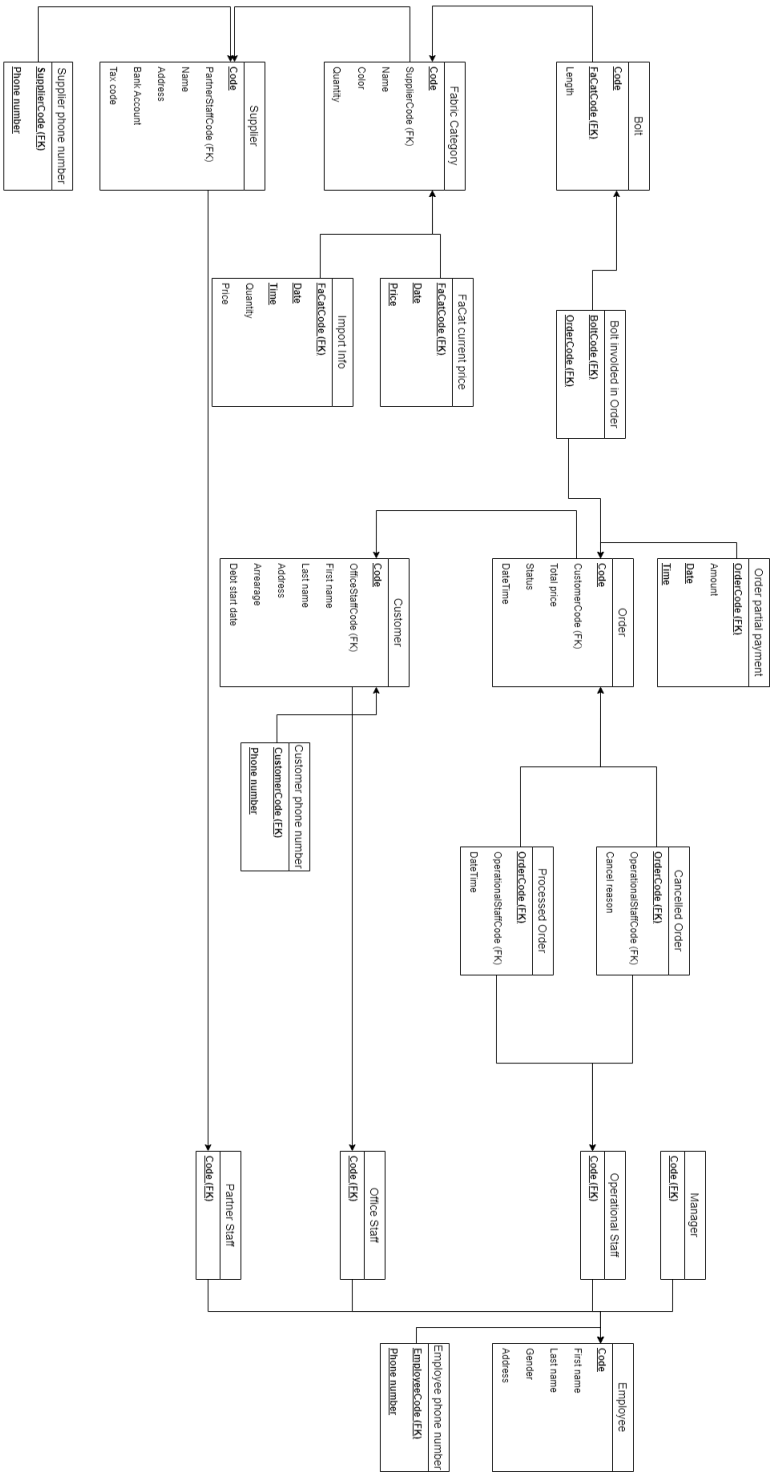
- Similarly for **Bolt** and **Order**, **Order** and **Customer**, **Customer** and **Office_Staff**, **Supplier** and **Partner_Staff**.

- For multiple-value attributes (`Current_price`, `Phone_number`, etc), we create new table with primary key is all the components of that table (include the foreign key points to the owner of that particular attribute).

- For specialization of **Employee**, we create 4 more tables for each of these, with the primary key is **Employee**.`Code` (foreign key).

- For specialization of **Order**, we decide to create 2 more tables for **Cancelled_Order** and **Processed_Order**, with primary key is **Order**.`Code` foreign key and another column for **Operational_Staff**.`Code` foreign key.

To see the Mapping to Relational Database Schema more clearly, please refer to the image file or the draw.io file.


Supplier phone number
SupplierCode (FK)
Phone number
Supplier
Code
PartnerStaffCode (FK)
Name
Address
Bank Account
Tax code
Fabric Category
Code
SupplierCode (FK)
Name
Color
Quantity
Bolt
Code
FaCatCode (FK)
Length
Import info
EaCatCode (FK)
Date
Time
Quantity
Price
FaCat current price
EaCatCode (FK)
Date
Price
Bolt invoiced in Order
BoltCode (FK)
OrderCode (FK)
Customer
Code
OfficeStaffCode (FK)
First name
Last name
Address
Arrearage
Debt start date
Order
Code
CustomerCode (FK)
Total price
Status
DateTime
Order partial payment
OrderCode (FK)
Amount
Date
Time
Customer phone number
CustomerCode (FK)
Phone number
Processed Order
OrderCode (FK)
OperationalStaffCode (FK)
DateTime
Cancelled Order
OrderCode (FK)
OperationalStaffCode (FK)
Cancel reason
Partner Staff
Code (FK)
Office Staff
Code (FK)
Operational Staff
Code (FK)
Manager
Code (FK)
Employee phone number
EmployeeCode (FK)
Phone number
Employee
Code
First name
Last name
Gender
Address

5 Additional Constraints

There are some constraints that are not shown in the Entity Relationship Diagram as well as Relational Database Schema:

- **Fabric_Category**.Name belongs to {silk, khaki, crewel, jacquard, faux silk, damask}
- **Order**.Status in {new, ordered, partial paid, full paid, cancelled}
  - When *order*.Status == "cancelled", data of this *order* is inserted in **Cancelled_Order** table. *order*.Status can only be "cancelled" when previously *order*.Status was "new"
  - When *order*.Status == "ordered", data of this *order* is inserted in **Processed_Order** table, and the corresponding *customer*.Arrearage increases by *order*.Total_price
  - When *order*.Status == "partial paid", this *order* must have at least one *partial_payment*
  - *order*.Status = "full paid" if and only if:

$$order.\texttt{Total\_price} == \texttt{sum}(order.\textbf{Partial\_payment}.\texttt{Amount}) \tag{1}$$

- When *customer* makes a *partial_payment* for an *order*:
  - Restrict the *partial_payment*.Amount to the following condition, otherwise the payment is unsuccessful:

$$partial\_payment.\texttt{Amount} <= order.\texttt{Total\_price} - \texttt{sum}(order.\textbf{Partial\_payment}.\texttt{Amount}) \tag{2}$$

  - If the payment is successful, decrease the *customer*.Arrearage by *partial_payment*.Amount
- **Customer**.Arrearage:
  - If *customer*.Arrearage > 2000, record *customer*.Debt_start_date as the current date, and *customer*.Mode is marked as "warning", then alert the agency. When *customer*.Arrearage $<= 2000$, set null the corresponding Debt_start_date and Mode. *customer*.Arrearage is checked after a *customer*'s *order* is processed or the *customer* makes a *partial_payment*
  - If Debt_start_date is not null and the difference between the current date and Debt_start_date > 6 months, *customer*.Mode is marked as "bad debt". To do that, the **Customer** table needs to be checked periodically at the start of each day