# Foundations of Databases A.Y. 2021-2022
# Homework 2 – Conceptual and Logical Design

## Master Degree in Computer Engineering
## Master Degree in Cybersecurity
## Master Degree in ICT for Internet and Multimedia

Deadline: November 26, 2021

| Team acronym | TAGMS | |
|---|---|---|
| **Last Name** | **First Name** | **Student Number** |
| Arslan | Tunahan | 2023640 |
| Esposito | Vittorio | 2005795 |
| Basso | Marco | 2005796 |
| Giuliani | Amedeo | 2005797 |
| Collado | Martin | 2039907 |
| Quiroz | Giannina | 2041427 |
| Zanini | Samuele | 2019038 |

# Conceptual Design

## Variations to the Requirement Analysis

An employee must necessarily have one and only one role. A role is typically associated with multiple employees but may not have any. For example, the "intern" role is created but without having hired anyone in that position yet.

An employee must work in at least one department. Each department can have multiple workers, but none if the department is new and newly created.

A manager can stipulate one or more contracts with many suppliers. A supplier can stipulate many contracts with the manager as well, but a new supplier can also not have stipulated anything yet. Our system, therefore, is able to also store information relating to suppliers with whom the company has not yet entered into agreements.

Each contract specifies one or more items (e.g., ingredients, packaging materials, etc) and the respective quantity. An item can be specified in at least one contract, so it can be provided by many different suppliers. Each item belongs to only one item category, and to an item category can belong zero or many products.

A product, which is a finished good ready to be sold, is made up of one or more items (e.g., one glass bottle, a hundred grams of sugar, a hundred milliliters of water, ...) with the respective quantities. For example, a product called "Coke J" can consist of one aluminum can, 50ml of water, 10g of sugar, etc. Another product, called "Coke B" for example, may have the same ingredients as the previous example but can be packaged with a glass bottle. The expiration date of a product is specified in the various batches that include that particular product and the date may differ in each lot. The stock quantity of a product is not explicitly specified, but it can be obtained by checking the specified product quantities in each batch not yet sold or shipped. An item can be utilized in many products (also none if, for example, the item is brand new). Each product belongs to one and only one product category, that is used to distinguish them. To a product category can belong zero or many products. The expiration date of the ingredients (items) is not tracked as the company guarantees to keep them stored for a short period of time because the ingredients are used shortly after their purchase and a FIFO policy is used.

A package is composed by one or more packaging materials (i.e., an item used for packaging, such as a box, a meter of plastic tape, a kilogram of polystyrene, etc) with the respective quantities. For example, a package named "PK1" can consist of 4 boxes of dimensions 30cm x 30cm x 10cm, 2 meters of plastic tape and 200 g of polystyrene. A "PK2" package can consist of 6 boxes of dimensions 30cm x 30cm x 10cm, 4 meters of plastic tape and 300 g of polystyrene. An item (e.g., packaging material in this case) can be utilized in many packages (also none if, for example, the item is brand new). Each package belongs to one and only one package category, that is used to distinguish them. To a package category can belong zero or many packages.

In a lot they can be stocked many products (i.e., only one type of product in a certain quantity) and one or more packages (i.e., only one type of package in a certain quantity). The number of products in each batch depends both on the dimensions of the package of the individual product (e.g., bottle of glass) and on the features of the package (in particular, the size of the box and the number of boxes that make up the package). Each product can be stocked in many lots (in none if, for example, the product is brand new) and the same holds true for packages. Each lot is characterized by an expiration date. As some lots may be produced in advance to reduce lead times, some of them may not sell on time and therefore expire. The data analyst will perform analysis in this regard to reduce waste.

The customer decides with the seller regarding the products to be bought. The salesman, then, after communicating the products (with respective quantity) to the warehouse worker, will place the order only when all lots are ready. A seller can place zero or many orders for a customer, so a customer can make many order (none if, for example, the customer is new). An order can be place by only one salesman for only one customer.

An order includes one or more lots. Each lot can be included by only one order (none if the lot is produced in advanced and waiting to be ordered). The invoice (after-sales document) will be automatically generated by the application linked to the DBMS as soon as the order is paid. When the order is ready, a worker will ship it: a worker can ship zero or many orders, and an order can be shipped by at most one worker (none if the order is waiting to be shipped). The cancellation of the order is not foreseen since the goods can be produced on commission and the company wants to minimize the waste caused by the expiry of the products. The customer can be informed about the status of the order by contacting the seller. If the goods have been shipped, the customer can use the tracking code provided to him to monitor the shipment from the courier's web portal.
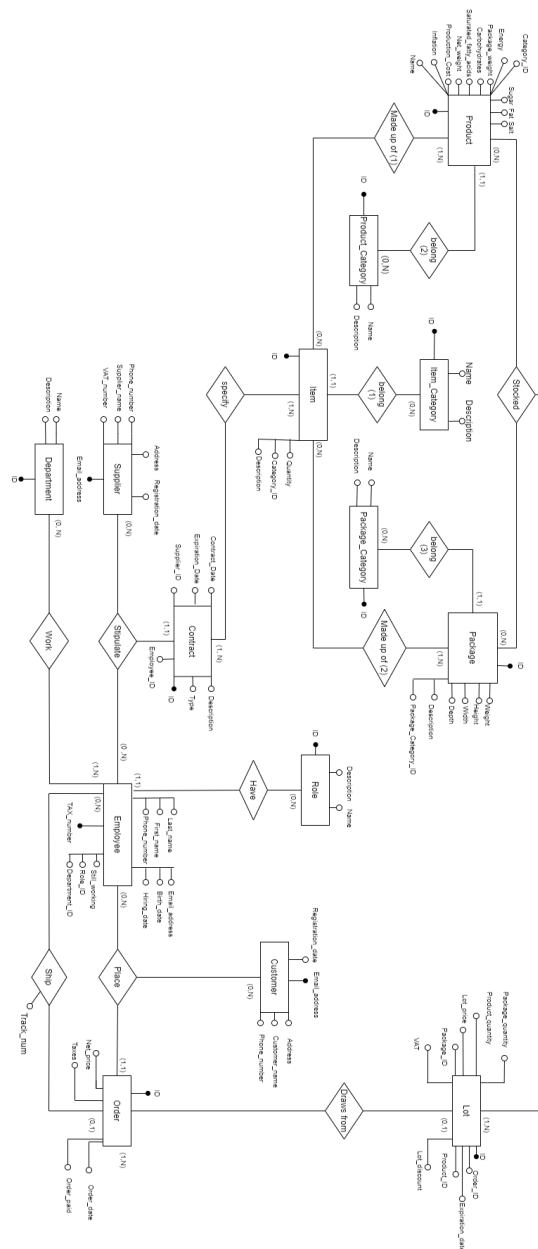
# Entity-Relationship Schema



Figure 1: Entity-Relationship schema.

# Data Dictionary

## Entities Table

| Entity | Description | Attributes | Identifier |
|---|---|---|---|
| Employee | Represents data of an employee who works in the company and needs access to the system | <ul><li>TAX_number: TAX code of the employee as badge ID, text</li><li>First_name: name of the employee, text</li><li>Last_name: surname of the employee, text</li><li>Phone_number: phone number of the employee, int</li><li>Email_address: email address of the employee, text</li><li>Birth_date: birth date of the employee, datetime</li><li>Hiring_date: hiring date of the employee, datetime</li><li>Still_working: To know if employee is still working for the company, boolean</li><li>Role_ID: role identifier, int</li><li>Department_ID: department identifier, int</li></ul> | TAX_number |
| Role | Represents data on the role of employees who work in the company | <ul><li>ID: role ID of the employee, serial</li><li>Name: name of the role, text</li><li>Description: technical description of the role, text</li></ul> | ID |

| | | | |
|---|---|---|---|
| Department | Represents data on the departments in which employees work | • ID: department ID of the company, serial<br><br>• Name: name of the department, text<br><br>• Description: description of the department's function, text | ID |
| Customer | Represents data about a customer of the company | • Customer_name: name of the customer, text<br><br>• Phone_number: phone number of the customer, int<br><br>• Email_address: email address of the customer, text<br><br>• Address: billing address of the customer, text<br><br>• Registration_date: customer registration date in the database, datetime | Email_address |

| Contract | Represents data about a contract stipulated between a supplier and a manager for the supply of items | <ul><li>ID: contract ID, serial</li><li>Type: type of the contract, text</li><li>Description: description of the contract, text</li><li>Contract_date: date of signature of the contract with the supplier, datetime</li><li>Expiration_date: expiration date of the contract with the supplier, datetime</li><li>Supplier_ID: VAT as supplier ID, int</li><li>Employee_ID: employee ID, text</li></ul> | ID |
|---|---|---|---|
| Supplier | Represents data about a supplier of the company | <ul><li>VAT_number: VAT number of supplier, int</li><li>Supplier_name: name of the supplier company, text</li><li>Phone_number: phone number of the supplier company, int</li><li>Email_address: email address of the supplier company, text</li><li>Address: address of the supplier company, text</li><li>Registration_date: recording date of the supplier company, datetime</li></ul> | Emai_address |

| Order | Represents the order placed by a salesman for a customer | <ul><li>ID: order ID, serial</li><li>Order_date: date in which the order has been processed, datetime</li><li>Order_paid: status of the payment, boolean</li><li>Net_price: total amount including discount(VAT excluded), float</li><li>Taxes: total amount of taxes to be payed, float</li></ul> | ID |
|---|---|---|---|
| Lot | Represents the inventory of the company, containing products and items | <ul><li>ID: lot ID, serial</li><li>Order_ID: order ID, serial</li><li>Product_ID: product ID, serial</li><li>Product_quantity: quantity of the produced product, int</li><li>Package_ID: package ID related to the specific lot, serial</li><li>Package_quantity: lot packaging quantity, int</li><li>Expiration_date: expiration date of the specific product, datetime</li><li>Lot_price: lot price without the discount, float</li><li>Lot_discount: total discount of a specific lot, int</li><li>VAT: value added tax at the time o sale, int</li></ul> | ID |

| Product | Represents the final product that is marketed | <ul><li>ID: product ID, serial</li><li>Name: name of the product, text</li><li>Energy: energy nutritional value, float</li><li>Fat: fat nutritional value, float</li><li>Saturated fatty acids: saturated fatty acids nutritional value, float</li><li>Carbohydrates: carbohydrates nutritional value, float</li><li>Sugar: sugar nutritional value, float</li><li>Protein: protein nutritional value, float</li><li>Salt: salt nutritional value, float</li><li>Net_weight: net weight of the product, float</li><li>Package_weight: gross weight of the product, float</li><li>Production_cost: cost of the production for a product, float</li><li>Inflation: price increase factor, float</li><li>Category_ID: category of the product, serial</li></ul> | ID |
|---|---|---|---|

| | | | |
|---|---|---|---|
| Item | Represents materials provided by suppliers from which the final products will be produced | • ID: ID of the item, int<br><br>• Description: description of the item, text<br><br>• Category_ID: ID of the category group, serial<br><br>• Quantity: quantity of the specific item, int | ID |
| Package | Represents packaging of finished products which are made up of boxes, tapes, and other packaging materials | • ID: ID of the package, serial<br><br>• Description: description o the package, text<br><br>• Weight: weight dimension of the package, int<br><br>• Height: height dimension of the package, int<br><br>• Width: width dimension of the package, int<br><br>• Depth: depth dimension of the package, int<br><br>• Package_Category_ID: ID of the package category, serial | ID |
| Product_Category | Represents the category of a product, an item or a package | • ID: ID of the product category, int<br><br>• Name: name of the product category, text<br><br>• Description: description of the product category, text | ID |

| Item_Category | Represents the category of a product, an item or a package | <ul><li>ID: ID of the item category, int</li><li>Name: name of the item category, text</li><li>Description: description of the item category, text</li></ul> | ID |
|---|---|---|---|
| Package_Category | Represents the category of a product, an item or a package | <ul><li>ID: ID of the package category, int</li><li>Name: name of the package category, text</li><li>Description: description of the package category, text</li></ul> | ID |

## Relationships Table

| Relationship | Description | Component Entities | Attributes |
|---|---|---|---|
| Have | Relates each employee to a role | <ul><li>Employee (1,1)</li><li>Role (0,N)</li></ul> | None |
| Work | Assigns each employee to a department | <ul><li>Employee (1,1)</li><li>Department (0,N)</li></ul> | None |
| Stipulate | Links the supplier with the company and the contract stipulated | <ul><li>Supplier (0,N)</li><li>Employee (0,N)</li><li>Contract (1,1)</li></ul> | None |

| Place | Links the order made by the employee | • Employee (0,N) <br><br> • Order (1,1) <br><br> • Customer (0,N) | None |
|---|---|---|---|
| Ships | Relates the employee shipping the order with the order itself and the shipment details | • Employee (0,N) <br><br> • Order (0,1) | Tracking number |
| Specify | Describes which items are provided by a contract | • Contract (1,N) <br><br> • Item (1,N) | None |
| Belongs to (1) | Links item to the category | • Item (1,1) <br><br> • Item_Category (0,N) | None |
| Belongs to (2) | Product item to the category | • Product (1,1) <br><br> • Product_Category (0,N) | None |
| Belongs to (3) | Package product to the category | • Package (1,1) <br><br> • Package_Category (0,N) | None |
| Made up of (1) | Describes which items are involved into creation to the product | • Item (0,N) <br><br> • Product (1,N) | None |
| Made up of (2) | Describes which items are involved into creation to the package | • Item (0,N) <br><br> • Package (1,N) | None |

| Stocked | Specifies the items and product stocked in the inventory | • Package (0,N)  • Product (0,N)  • Lot (1,N) | None |
|---|---|---|---|
| Draws from | Associates the order and the lot | • Order (1,N)  • Lot (0,1) | None |

## External Constraints

- Employees can only insert and modify transactions in their particular department and role (i.e workers assigned in raw materials cannot add finished products transactions)

- A Role is associated with multiple employees but an employee may only have one and specific role

- Customers coordinate with Salesmen thus, only Salesmen can create Customers' profile, orders and payment and track status

- Employees can only ship orders once payment of Customer has been confirmed

- Products and Items belong in a specific category and must be added correctly by the Employees

- Items must be issued following the First In, First Out Policy

- Unit of Measurement are in grams and centimetes

- Discount of 0 to 100 depends by the company's capability and Valued Added Tax (VAT) varies overtime

- Each lot can have 25, 50, 100 products. So, a customer can make an order of a quantity of products that can be obtained adding lots of these amounts. f.e A customer will not be able to buy 70 unit of a products, he must buy 75 units of a products (A lot of 25 and a lot of 50).

## Functional Requirements Satisfaction Check

The DBMS has to be able to:

- **store all the details of the employees, customers and suppliers in the organization:** Employee and Role entities store data related to the employees. Customer entity has details about the customers and Supplier entity has data related to the Supplier.

- **allow the employees to update their personal information:** Employee entity has some attributes as Email_address, Password or Phone_number that can be changed. Employees can access the system using their credentials and change this data.

- **store details of all on-hand products in the inventory such as item code, item description, quantity and expiration date:** The concept of inventory is implemented by the entities Item, Product and Lot, whose attributes show the data regarding the stock. The amount of each item is shown in attribute Quantity of entity Item. Assembled products are organized in lots whose information is contained in Lot entity in the attributes ID, Product_ID (the product that lot is composed of), Product_quantity (the amount of elements of product the lot is composed of), Expiration_date (common for all elements of the lot) and Order_ID (the order the lot is assigned to, if NULL determines a not-yet-assigned lot).

- **allow the employees to log into the system and enter the inbound items they received with information item code, item description, quantity, expiration date and supplier:** Employees can log in the database and insert this data in the entity Item, that could be inserting a new instance or updating an existing one.

- **show and generate the list of inbound and outbound transactions:** the inbound transactions can be derived from instances of the Contract entity, the outbound transactions can be derived from instances of the Order entity.

- **allow the employees to log into the system and enter the outbound transaction needed for the issuance of the products in the production and shipment to the customers:** Salesmen are responsible of entering outbound transactions, that are instances of the Order entity with proper values for the attributes. The salesman will check for not-yet-assigned lots that satisfy the requirements and initialize a relationship between lot and order by updating the Order attribute in Lot entity. Net_price is a derived attribute, computed automatically given the composition of the lot and the product cost.

- **inventory stocks will be automatically updated whenever there are inbound and outbound transactions:** the update is executed automatically when an inbound transaction happens by inspecting each ID_Item **x** and associated Quantity $Q_x$ in the relative Contract, then for each said **x** its Item_Quantity attribute in the entity Item is increased by $Q_x$. For outbound transactions, lots with NULL value in the order attribute are in the stock and unassigned, and lots associated to an order that is not in a Ship relationship are the ones that are waiting in the stock to be shipped. Therefore, these two sources of information allow to determine the lots in the stock at every time.

- **show and generate the current inventory balance or stock inquiries:** The Item entity has data related to the quantity of each item stored. The Lot entity has data about the quantity of products each lot contains, and which lots are in the stock or not-shipped yet. This latter information can be retrieved checking if there exist a relationship "Ship" that involves the order a lot is assigned to, since this relationship is created only once the order is actually shipped to the customer. An employee can access the instances of these entity and specifically the attributes Quantity of Item entity and Product_quantity of Lot entity (eventually, of all lots containing a certain product) to determine the exact current content of the stock.

- **receive and process the Customers order, specifying which products they want and respective quantity:** Salesmen are able to access the database and enter an instance of the Order entity reporting the desired list of products and the respective quantities. The Salesman will find and assign appropriate lots if they already exist, and if they do not exist it will notify the Production department which will notify back when they are ready. As soon as this happens, the order will be entered and will be uniquely determined by ID.

- **modification and cancellation of orders:** the salesman can change or cancel the order accessing it through the Attribute ID of Order entity.

- **allow users to view order and shipment status of finished products:** With the unique tracking number the attribute "Tracking number" of the Ship relationship, and the unique ID attribute of the Order entity the users can get information about the order and shipment.

- **generate invoice whenever payment has been made:** When a Salesman creates an order, it sets automatically to False a boolean attribute Order_paid that determines a not-payed-yet order. When the customer pays the order, the information about the payment is inserted as attributes of the Place relationship. The invoice document can be generated instantly extracting the information from the related instance of the Order entity. Final_Price is computed given the attributes Net_Price and Taxes.

- **permit transfer of items and products:** Entity Customer has an attribute Address which refers to where the ordered products are going to be shipped. When the customer pays the order and so the attribute Order_paid of the Order entity is set to True, the Order information is forwarded to the workers that inspect the related Address, set up the shipment and create a Ship relationship with Tracking number

- **grant Cycle Counting in order to validate the accuracy of inventory:** Cycle counting is meant as a periodic check by the Managers or by a Worker Supervisor on the items and products in the warehouse. After collecting the real quantities for each item and the list of lots in the stock, the Manager can update (if necessary) the values of such attributes in the database to the correct value.

- **re-ordering the previous orders is allowed:** The system allows salesmen to access past orders using the ID attribute and retrieve information about the lots, the products, their quantities and all the necessary data to set up a new order with the same content.

- **create tracking code for orders:** Attribute Tracking number of Ship relationship stores an identifier, provided by the third-party company responsible for the shipping, that uniquely identifies the shipment.

## Logical Design

### Transformation of the Entity-Relationship Schema

### Redundancy Analysis

### Choice of Principal Identifiers

### Analysis of Database Load

### Relational Schema

### Data Dictionary

| Relation | Attribute | Description | Domain | Constraints |
|----------|-----------|-------------|--------|-------------|

| Relation 1 | | | | |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| Relation 1 | | | | |
| | | | | |

## External Constraints

# Group Members Contributions

## Conceptual Design

- **Variations to the Requirement Analysis: Esposito**

- **Entity-Relationship Schema: Esposito,**

- **Entities Table: Esposito,**

- **Relationships Table:**

- **External Constraints:**

- **Functional Requirements Satisfaction Check:**

## Logical Design

- **Redundancy Analysis:**

- **Choice of Principal Identifiers:**

- **Analysis of Database Load:**

- **Relational Schema:**

- **Data Dictionary:**

- **External Constraints:**