By Irina Coman

Database Project 2024

Projection of a database for the management of a chain of cosmetic shops

This is an SQL project I created for managing a chain of cosmetic stores, developed by following these requirements:

Contents

[**1. Description of the real model, its utility, and the operational rules.** 2](#_Toc177055307)

[**3. Description of entities, including specifying the primary key.** 4](#_Toc177055308)

[**4. Description of relationships, including specifying their cardinality.** 6](#_Toc177055309)

[**5. Description of attributes, including data types, any constraints, default values, and possible attribute values.** 7](#_Toc177055310)

[**6. Creation of the entity-relationship diagram corresponding to the description from points 3-5.** 12](#_Toc177055311)

[**7. Creation of the conceptual diagram corresponding to the entity-relationship diagram designed in point 6. The resulting conceptual diagram must contain at least 7 tables (excluding subentities), including at least one associative table.** 13](#_Toc177055312)

[**8. Enumeration of the relational schemas corresponding to the conceptual diagram designed in point 7.** 14](#_Toc177055313)

[**9. Normalization up to the 3rd normal form (1NF-3NF).** 14](#_Toc177055314)

[ First Normal Form (1NF) 14](#_Toc177055315)

[ Second Normal Form (2NF) 15](#_Toc177055316)

[ Third Normal Form (3NF) 16](#_Toc177055317)

[**10. Creation of a sequence to be used for inserting records into tables (point 11).** 16](#_Toc177055318)

[**11. Creation of tables in SQL and insertion of coherent data into each of them (at least 5 records in each non-associative table; at least 10 records in associative tables; a maximum of 30 records in each table).** 18](#_Toc177055319)

[**12. Formulate in natural language and implement 5 complex SQL queries that, in their entirety, will utilize the following elements:** 45](#_Toc177055320)

[**Exercise 1** 46](#_Toc177055321)

[**Exercise 2** 47](#_Toc177055322)

[**Exercise 3** 48](#_Toc177055323)

[**Exercise 4** 49](#_Toc177055324)

[**Exercise 5** 51](#_Toc177055325)

[**13. Implementation of 3 operations for updating and deleting data using subqueries.** 52](#_Toc177055326)

[**Exercise 1** 52](#_Toc177055327)

[**Exercise 2** 53](#_Toc177055328)

[**Exercise 3** 53](#_Toc177055329)

[**14. Creation of a complex view. Provide an example of a DML operation allowed on that view and an example of a DML operation not allowed.** 54](#_Toc177055330)

[**Allowed DML Operation** 55](#_Toc177055331)

[**Not Allowed DML Operation** 55](#_Toc177055332)

[**15. Formulate in natural language and implement in SQL: a query that uses the outer join operation on at least 4 tables, a query that uses the division operation, and a query that implements top-n analysis.** 56](#_Toc177055333)

[**Outer join** 56](#_Toc177055334)

[**Division** 57](#_Toc177055335)

[**Top-N Analysis** 57](#_Toc177055336)

# **1. Description of the real model, its utility, and the operational rules.**

In this paragraph, I will present the data model for a chain of cosmetic stores, its utilities and operating rules.

This database contains information about the cosmetic stores within this chain, the cities where these stores are located, the managers who run them, the products sold, the employees working in these stores, the suppliers products are ordered from for stock maintenance, the orders with their details, placed to suppliers, the customers who purchase products from these stores and the sales done in these stores. The purpose of creating this database is to keep the evidence of inventory, orders to suppliers, employees and their distribution in various stores and sales realized in these stores, with products purchased by various customers.

Every store has a specific location and a functioning schedule and could have multiple employees for fulfilling different functions, such as managers and sales associates. Every store places orders to suppliers for renewing their stock of products. These orders contain details about the ordered products, their quantities and estimated delivery dates. Moreover, every store tracks the sales made, the details of transactions being taken into account, too.

Suppliers are responsible for delivering ordered products to stores. Products are classified into different categories, such as makeup, skincare, haircare etc. The relationship between orders and suppliers is managed through an associative table which allows tracking data about every order.

Employees work in these cosmetic stores, having a specific set of responsibilities, like direct interaction with customers and offering recommendation of products. Employees’ evidence includes personal information, position held, hiring date and the manager of the store they work in.

Managers have surveillance and coordination responsibilities in every store. They are responsible for the daily operation of their store, team management, monitoring sales and team’s performance and maintaining a high level of customer satisfaction. Data about managers includes, in addition to general information about employees, the specified store where they carry out their activity, contributing this way to operational organisation and eficiency.

The products include a wide variety of makeup, haircare, skincare and other selfcare articles. Every product is cataloged with essential details, like an ID, name, brand, type and price.

Orders represent product purchase requests placed by shops to suppliers for renewing their stock. Every order includes an ID, details about supplier, the date when it was placed, the estimated date of delivering and its total price. Monitoring orders is essential to ensure that stores have enough stock and to avoid product shortages.

Sales represent transactions realised between customers and sellers, in the cosmetic stores. Every sale differs from the others by the date when it was realised, total price, the products purchased and the seller who completed this transaction.

**2. Presentation of the constraints (restrictions, rules) imposed on the model.**

The data model respects some functioning restrictions:

* A store has one or more employees, and each employee works in only one store.
* A store has a single manager, and each manager manages only one store.
* A store can place multiple orders, and each order belongs to a single store.
* A city may or may not have stores, and each store is located in only one city.
* A manager supervises one or more employees, and each employee is under the supervision of only one manager.
* An employee conducts multiple sales to multiple customers, and a sale is made by a single employee to a single customer.
* A customer may be involved in a sale or not, meaning they may or may not purchase something from the store.
* A supplier receives one or more orders, and each order is placed with only one supplier.
* An order contains multiple products of the same type.
* A sale involves at least one product.

# **3. Description of entities, including specifying the primary key.**

For the data model related to the management of a cosmetic store chain, the entities are**: MAGAZIN (STORE), ANGAJAT (EMPLOYEE), MANAGER, ORAȘ (CITY), COMANDĂ (ORDER), FURNIZOR (SUPPLIER), PRODUS (PRODUCT), VÂNZARE (SALE)** and **CLIENT (CUSTOMER).**

In this section, these entities will be presented, providing a full description and mentioning the primary key for each.

All entities presented are independent, except for the dependent entities **RECEPȚIONEAZĂ (RECEIVES), LUCREAZĂ (WORKS)** and **VINDE (SELLS).**

MAGAZIN (STORE) = the physical location where makeup and personal care products can be purchased. Each store has employees, a manager and can place orders to suppliers to replenish its product stock. The primary key of this entity is **ID\_Magazin**.

ANGAJAT (EMPLOYEE) = the person who works in one of the cosmetic stores and is responsible for customer interaction and processing sales. The primary key of this entity is **ID\_Angajat**.

MANAGER = the person who supervises and coordinates one of the stores. The manager is responsible for the store's daily operation, team management, and monitoring sales and employee performance. The primary key of this entity is **ID\_Manager**.

ORAȘ (CITY) = the geographical location where the chain's cosmetic stores are located. Each city may host one or more stores. The primary key of this entity is **ID\_Oraș**.

COMANDĂ (ORDER) = a product purchase request placed by stores to suppliers. Each order includes details about the supplier, order date, estimated delivery date, total value and purchased products. Orders help stores maintain the necessary stock for sales. The primary key of this entity is **ID\_Comandă**.

FURNIZOR (SUPPLIER) = the entity that supplies cosmetic products to the stores. Each supplier has a set of products they offer and receives orders from stores to replenish stock. Supplier data includes contact information and details about the supplied products. The primary key of this entity is **ID\_Furnizor**.

PRODUS (PRODUCT) = a cosmetic item sold in stores, including makeup and personal care products. Each product is cataloged with a unique identifier, product name, brand, price and stock. Products are ordered by stores to maintain their stock. The primary key of this entity is **ID\_Produs**.

VÂNZARE (SALE) = a transaction made in cosmetic stores, where customers purchase products. Each sale includes details such as transaction date, total amount, sold products and the employee who processed the sale. This information is essential for sales analysis and inventory management. The primary key of this entity is **ID\_Vânzare**.

CLIENT (CUSTOMER) = the person who purchases products from the cosmetic stores. The primary key of this entity is **ID\_Client**.

RECEPȚIONEAZĂ (RECEIVES) = the process of receiving an order placed by a store from a supplier. The composite primary key consists of **ID\_Comandă** and **ID\_Furnizor**.

LUCREAZĂ (WORKS) = the activity performed by an employee in one of the stores of this chain. The composite primary key consists of **ID\_Angajat** and **ID\_Magazin**.

VINDE (SELLS) = a customer buys a product from a store, and this transaction is performed by one of the store's employees. The composite primary key consists of the primary keys **ID\_Angajat**, **ID\_Vânzare**, and **ID\_Client**.

# **4. Description of relationships, including specifying their cardinality.**

Next, the relationships between the entities listed above will be presented and described. For each relationship, both the minimum and maximum cardinalities will be specified.

**MANAGER\_gestionează\_MAGAZIN (MANAGER\_manages\_STORE)** = a relationship that links the MANAGER and MAGAZIN (STORE) entities, highlighting the connection between them (A store has one manager, and each manager manages one store). The relationship has a minimum and maximum cardinality of **1:1**.

**MAGAZIN\_dă\_COMANDĂ (STORE\_places\_ORDER)** = a relationship that links the MAGAZIN (STORE) and COMANDĂ (ORDER) entities, highlighting the connection between them (A store can place multiple orders, and each order belongs to one store). The relationship has a minimum cardinality of **1:0** and a maximum cardinality of **1:n**.

**MAGAZIN\_se\_află\_în\_ORAȘ (STORE\_is\_located\_in\_CITY)** = a relationship that links the MAGAZIN (STORE) and ORAȘ (CITY) entities, highlighting the connection between them (A store is located in one city, and a city may or may not have stores). The relationship has a minimum cardinality of **0:1** and a maximum cardinality of **n:1**.

**MANAGER\_conduce\_ANGAJAT (MANAGER\_supervises\_EMPLOYEE)** = a relationship that links the MANAGER and ANGAJAT (EMPLOYEE) entities, highlighting the connection between them (A manager supervises one or more employees, and each employee is supervised by only one manager). The relationship has a minimum cardinality of **1:1** and a maximum cardinality of **1:n**.

**COMANDĂ\_conține\_PRODUS (ORDER\_contains\_PRODUCT)** = a relationship that links the COMANDĂ(ORDER) and PRODUS(PRODUCT) entities, highlighting the connection between them (An order contains multiple products of the same type). The relationship has a minimum cardinality of **0:n** and a maximum cardinality of **1:n**.

**VÂNZARE\_implică\_PRODUS (SALE\_involves\_PRODUCT)** = a relationship that links the VÂNZARE (SALE) and PRODUS (PRODUCT) entities, highlighting the connection between them (A sale involves at least one product). The relationship has a minimum cardinality of **0:1** and a maximum cardinality of **1:n**.

**FURNIZOR\_primește\_COMANDĂ (SUPPLIER\_receives\_ORDER)** = a relationship that links the FURNIZOR (SUPPLIER) and COMANDĂ(ORDER) entities, highlighting the connection between them (A supplier receives one or more orders, and each order is placed with a single supplier). The relationship has a minimum cardinality of **1:1** and a maximum cardinality of **1:n**. The name of this relationship will be RECEPȚIONEAZĂ (RECEIVES).

**ANGAJAT\_lucrează\_în\_MAGAZIN (EMPLOYEE\_works\_in\_STORE)** = a relationship that links the ANGAJAT (EMPLOYEE) and MAGAZIN (STORE) entities, highlighting the connection between them (A store has one or more employees, and each employee works in one store). The relationship has a minimum cardinality of **1:1** and a maximum cardinality of **n:1**. The name of this relationship will be LUCREAZĂ (WORKS).

**ANGAJAT\_efectuează\_VÂNZARE\_unui\_CLIENT (EMPLOYEE\_makes\_SALE\_to\_CUSTOMER)** = a relationship that links the ANGAJAT (EMPLOYEE), VÂNZARE (SALE) and CLIENT (CUSTOMER) entities, highlighting the connection between them (An employee makes multiple sales, and a sale is performed by a single employee, and a customer may or may not make a purchase). The name of this relationship will be VINDE (SELLS).

# **5. Description of attributes, including data types, any constraints, default values, and possible attribute values.**

For each entity in the data model, its attributes and their descriptions, including data types and possible constraints, default values, or allowed values, will be presented.

**MAGAZIN (STORE)** Entity**:**

* **ID\_Magazin (ID\_Store)** = integer variable, maximum length of 5, representing the unique identifier of the store.
* **Nume\_Magazin (Store\_Name)** = character variable, maximum length of 100, representing the name of the store.
* **Adresă\_Magazin (Store\_Address)** = character variable, maximum length of 100, representing the address of the store.
* **ID\_Oraș (ID\_City)** = integer variable, maximum length of 4, representing the unique identifier of the city where the store is located. This attribute must correspond to a primary key value in the ORAȘ (CITY) table.

**ANGAJAT (EMPLOYEE)** Entity**:**

* **ID\_Angajat (ID\_Employee)** = integer variable, maximum length of 3, representing the unique identifier of the employee.
* **Nume\_Angajat (Employee\_Name)** = character variable, maximum length of 100, representing the name of the employee.
* **Prenume\_Angajat (Employee\_FirstName)** = character variable, maximum length of 100, representing the first name of the employee.
* **Email\_Angajat (Employee\_Email)** = character variable, maximum length of 100, representing the email address of the employee.
* **Dată\_Angajare\_A (HireDate\_E)** = date variable representing the hire date of the employee.
* **ID\_Manager** = integer variable, maximum length of 3, representing the unique identifier of the employee's manager. This attribute must correspond to a primary key value in the MANAGER table.

**MANAGER** Entity**:**

* **ID\_Manager** = integer variable, maximum length of 3, representing the unique identifier of the manager.
* **Nume\_Manager (Manager\_Name)** = character variable, maximum length of 100, representing the manager's last name.
* **Prenume\_Manager (Manager\_FirstName)** = character variable, maximum length of 100, representing the manager's first name.
* **Email\_Manager** = character variable, maximum length of 100, representing the email address of the manager.
* **Dată\_Angajare (HireDate)** = date variable representing the hire date of the manager.
* **ID\_Magazin (ID\_Store)** = integer variable, maximum length of 5, representing the unique identifier of the store managed by the manager. This attribute must correspond to a primary key value in the MAGAZIN (STORE) table.

**ORAȘ (CITY)** Entity**:**

* **ID\_Oraș (ID\_City)** = integer variable, maximum length of 4, representing the unique identifier of the city.
* **Nume\_Oraș (City\_Name)** = character variable, maximum length of 100, representing the name of the city.
* **Regiune (Region)** = character variable, maximum length of 100, representing the region in which the city is located.
* **Cod\_Poștal (Postal\_Code)** = character variable, maximum length of 10, representing the city's postal code.

**COMANDĂ (ORDER)** Entity**:**

* **ID\_Comandă (ID\_Order)** = integer variable, maximum length of 10, representing the unique identifier of the order.
* **Dată\_Comandă (Order\_Date)** = date variable representing the order date.
* **Dată\_Livrare (Delivery\_Date)** = date variable representing the estimated delivery date.
* **Valoare\_Totală (Total\_Value)** = numeric variable with 2 decimal places, maximum length of 10, representing the total value of the order.
* **ID\_Magazin (ID\_Store)** = integer variable, maximum length of 5, representing the unique identifier of the store that placed the order. This attribute must correspond to a primary key value in the MAGAZIN (STORE) table.
* **ID\_Produs (ID\_Product)** = integer variable, maximum length of 6, representing the unique identifier of the product in the order. This attribute must correspond to a primary key value in the PRODUS (PRODUCT) table.

**FURNIZOR (SUPPLIER)** Entity**:**

* **ID\_Furnizor (ID\_Supplier)** = integer variable, maximum length of 6, representing the unique identifier of the supplier.
* **Nume\_Furnizor (Supplier\_Name)** = character variable, maximum length of 100, representing the supplier's name.
* **Adresă\_Furnizor (Supplier\_Address)** = character variable, maximum length of 100, representing the supplier's address.
* **Email\_Furnizor (Supplier\_Email)** = character variable, maximum length of 100, representing the supplier's email address.
* **Telefon (Phone)** = character variable, maximum length of 15, representing the supplier's phone number.

**PRODUS (PRODUCT)** Entity**:**

* **ID\_Produs (ID\_Product)** = integer variable, maximum length of 5, representing the unique identifier of the product.
* **Nume\_Produs (Product\_Name)** = character variable, maximum length of 100, representing the product's name.
* **Brand** = character variable, maximum length of 100, representing the brand of the product.
* **Preț (Price)**= numeric variable with 2 decimal places, maximum length of 10, representing the price of the product.
* **Stoc (Stock)** = integer variable, maximum length of 5, representing the quantity available in stock.

**VÂNZARE (SALE)** Entity**:**

* **ID\_Vânzare (ID\_Sale)** = integer variable, maximum length of 8, representing the unique identifier of the sale.
* **Dată\_Vânzare (Sale\_Date)** = date variable representing the sale date.
* **Sumă\_Totală (Total\_Amount)** = numeric variable with 2 decimal places, maximum length of 10, representing the total amount of the sale.
* **ID\_Produs (ID\_Product)** = integer variable, maximum length of 5, representing the unique identifier of the product sold. This attribute must correspond to a primary key value in the PRODUS (PRODUCT) table.

**CLIENT (CUSTOMER)** Entity**:**

* **ID\_Client (ID\_Customer)** = integer variable, maximum length of 4, representing the unique identifier of the customer.
* **Nume\_Client (Customer\_Name)** = character variable, maximum length of 100, representing the customer's last name.
* **Prenume\_Client (Customer\_FirstName)** = character variable, maximum length of 100, representing the customer's first name.

**ANGAJAT\_lucrează\_în\_MAGAZIN (EMPLOYEE\_works\_in\_STORE)** Relationship **(LUCREAZĂ (WORKS)** Entity**):**

* **ID\_Angajat (ID\_Employee)** = integer variable, maximum length of 3, representing the unique identifier of the employee working in the store. This attribute must correspond to a primary key value in the ANGAJAT (EMPLOYEE) table.
* **ID\_Magazin (ID\_Store)** = integer variable, maximum length of 5, representing the unique identifier of the store where the employee works. This attribute must correspond to a primary key value in the MAGAZIN (STORE) table.
* **Poziție (Position)** = character variable, maximum length of 50, representing the employee's position in the store.

**ANGAJAT\_efectuează\_VÂNZARE\_unui\_CLIENT (EMPLOYEE\_makes\_SALE\_to\_CUSTOMER)** Relationship **(VINDE (SELLS)** Entity**):**

* **ID\_Angajat (ID\_Employee)** = integer variable, maximum length of 3, representing the unique identifier of the employee who makes the sale. This attribute must correspond to a primary key value in the ANGAJAT (EMPLOYEE) table.
* **ID\_Vânzare (ID\_Sale)** = integer variable, maximum length of 8, representing the unique identifier of the sale. This attribute must correspond to a primary key value in the VÂNZARE (SALE) table.
* **ID\_Client (ID\_Customer)** = integer variable, maximum length of 4, representing the unique identifier of the customer to whom the sale was made. This attribute must correspond to a primary key value in the CLIENT (CUSTOMER) table.

**FURNIZOR\_primește\_COMANDĂ (SUPPLIER\_receives\_ORDER)** Relationship **(RECEPȚIONEAZĂ (RECEIVES)** Entity**):**

* **ID\_Furnizor (ID\_Supplier)** = integer variable, maximum length of 6, representing the unique identifier of the supplier who receives the order. This attribute must correspond to a primary key value in the FURNIZOR (SUPPLIER) table.
* **ID\_Comandă (ID\_Order)** = integer variable, maximum length of 10, representing the unique identifier of the received order. This attribute must correspond to a primary key value in the COMANDĂ (ORDER) table.
* **Dată\_Primire (Receive\_Date)** = date variable representing the date on which the supplier received the order.
* **Stare\_Comandă (Order\_Status)** = character variable, maximum length of 50, representing the status of the order.
* **Cantitate (Quantity)** = integer variable, maximum length of 3, representing the ordered quantity.

# **6. Creation of the entity-relationship diagram corresponding to the description from points 3-5.**

**E/R Diagram:**

A diagram of a company

Description automatically generated

# **7. Creation of the conceptual diagram corresponding to the entity-relationship diagram designed in point 6. The resulting conceptual diagram must contain at least 7 tables (excluding subentities), including at least one associative table.**

A diagram of a company

Description automatically generated

# **8. Enumeration of the relational schemas corresponding to the conceptual diagram designed in point 7.**

The relational schemas corresponding to the conceptual diagram are as follows:

* **MAGAZIN** (ID\_Magazin#, Nume\_Magazin, Adresă\_Magazin, ID\_Oraș)
* **ANGAJAT** (ID\_Angajat#, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Dată\_Angajare\_A, ID\_Manager)
* **MANAGER** (ID\_Manager#, Nume\_Manager, Prenume\_Manager, Email\_Manager, Dată\_Angajare, ID\_Magazin)
* **ORAȘ** (ID\_Oraș#, Nume\_oraș, Regiune, Cod\_Poștal)
* **COMANDĂ** (ID\_Comandă#, Dată\_Comandă, Dată\_Livrare, Valoare\_Totală, ID\_Magazin, ID\_Produs)
* **FURNIZOR** (ID\_Furnizor#, Nume\_Furnizor, Adresă\_Furnizor, Email\_Furnizor, Telefon)
* **PRODUS** (ID\_Produs#, Nume\_Produs, Brand, Preț, Stoc)
* **VÂNZARE** (ID\_Vânzare#, Dată\_Vânzare, Sumă\_Totală, ID\_Produs)
* **CLIENT** (ID\_Client#, Nume\_Client, Prenume\_Client)
* **RECEPȚIONEAZĂ** (ID\_Furnizor#, ID\_Comandă#, Dată\_Primire, Stare\_Comandă, Cantitate)
* **LUCREAZĂ** (ID\_Angajat#, ID\_Magazin#, Poziție)
* **VINDE** (ID\_Angajat#, ID\_Vânzare#, ID\_Client#)

# **9. Normalization up to the 3rd normal form (1NF-3NF).**

## First Normal Form (1NF)

A relationship is in 1NF (First Normal Form) if each attribute it contains corresponds to an indivisible value. The First Normal Form also imposes that each record must be uniquely identified through a primary key.

An example of a non-1NF form could appear in the ORDER table, as follows:

|  |  |  |
| --- | --- | --- |
| **ID\_Comandă (ID\_Order)** | **Dată (Date)** | **Valoare\_Totală(Total\_Value)** |
| 6731983147 | 2022-01-15, 2022-01-20 | 1050.00 |
| 4729864671 | 2022-01-18, 2022-01-23 | 4000 |

It can be observed that, in the example above, the Dată (Date) column can be split into two distinct columns. The correct model in 1NF form, which will be implemented, is the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID\_Comandă (ID\_Order)** | **Dată\_Comandă (Order\_Date)** | **Dată\_Livrare (Delivery\_Date)** | **Valoare\_Totală (Total\_Value)** |
| 6731983147 | 2022-01-15 | 2022-01-20 | 1050.00 |
| 4729864671 | 2022-01-18 | 2022-01-23 | 4000 |

## Second Normal Form (2NF)

A relation is in Second Normal Form (2NF) if and only if it is already in 1NF, and every non-primary key attribute is fully dependent on the entire primary key. 2NF prohibits the existence of partial functional dependencies within the relation.

An example of a NON-2NF form could appear in the VINDE (SELLS) table, as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID\_Angajat (ID\_Employee)** | **ID\_Vânzare (ID\_Sale)** | **ID\_Client (ID\_Customer)** | **Dată\_Vânzare (Sale\_Date)** | **Preț (Price)** |
| 101 | 94696243 | 3576 | 2024-03-10 | 195.00 |
| 101 | 55315628 | 6326 | 2024-02-20 | 40.00 |
| 102 | 32153543 | 7983 | 2024-02-10 | 45.75 |

The relation is in 1NF – we have a unique identifier for all entries in the tables.

It can be observed that there is a functional dependency between the primary keys: ID\_Angajat, ID\_Vânzare, and ID\_Client. However, the attributes Data\_Vânzare and Preț are dependent only on ID\_Vânzare, which places the relation outside of 2NF.

The correct model in 2NF, to be implemented, is as follows:

|  |  |  |
| --- | --- | --- |
| **ID\_Angajat (ID\_Customer)** | **ID\_Vânzare (ID\_Sale)** | **ID\_Client (ID\_Customer)** |
| 101 | 94696243 | 3576 |
| 101 | 55315628 | 6326 |
| 102 | 32153543 | 7983 |

## Third Normal Form (3NF)

A relation is in the third normal form (3NF) if and only if it is in 2NF, and every non-key attribute is directly dependent on the primary key.

An example of a NON-3NF form could appear in the MAGAZIN (STORE) table as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID\_Magazin (ID\_Store)** | **Nume\_Magazin (Store\_Name)** | **Adresă\_Magazin (Store\_Address)** | **Nume\_Oraș (City\_Name)** | **Regiune (Region)** | **Cod\_Poștal (Postal\_Code)** |
| 17634 | Sevora Central | Str. Libertății nr. 12 | București | Muntenia | 302813 |
| 28637 | Sevora Nord | Str. Gării nr. 5 | Cluj-Napoca | Transilvania | 481462 |

It can be observed that the attributes Nume\_Oraș, Regiune, and Cod\_Poștal do not depend on the primary key ID\_Magazin. To bring the relation into 3NF, these attributes related to the city are separated from the MAGAZIN (STORE) table, creating a new ORAȘ (CITY) table. Thus, these attributes are replaced with the foreign key ID\_Oraș to more easily determine in which city a store is located.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID\_Magazin (ID\_Store)** | **Nume\_Magazin (Store\_Name)** | **Adresă\_Magazin (Store\_Address)** | **ID\_Oraș (ID\_City)** |
| 17634 | Sevora Central | Str. Libertății nr. 12 | 2793 |
| 28637 | Sevora Nord | Str. Gării nr. 5 | 3292 |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID\_Oraș (ID\_City)** | **Nume\_Oraș (City\_Name)** | **Regiune (Region)** | **Cod\_Poștal (Postal\_Code)** |
| 2793 | București | Muntenia | 302813 |
| 3292 | Cluj-Napoca | Transilvania | 481462 |

# **10. Creation of a sequence to be used for inserting records into tables (point 11).**

SQL code:

CREATE SEQUENCE ORAS\_SEQ START WITH 1;

CREATE SEQUENCE MAGAZIN\_SEQ START WITH 1;

CREATE SEQUENCE ANGAJAT\_SEQ START WITH 1;

CREATE SEQUENCE MANAGER\_SEQ START WITH 1;

CREATE SEQUENCE FURNIZOR\_SEQ START WITH 1;

CREATE SEQUENCE PRODUS\_SEQ START WITH 1;

CREATE SEQUENCE CLIENT\_SEQ START WITH 1;

CREATE SEQUENCE COMANDA\_SEQ START WITH 1;

CREATE SEQUENCE VANZARE\_SEQ START WITH 1;

CREATE SEQUENCE RECEPTIONEAZA\_SEQ START WITH 1;

CREATE SEQUENCE LUCREAZA\_SEQ START WITH 1;

CREATE SEQUENCE VINDE\_SEQ START WITH 1;

Code source:

A screenshot of a computer

Description automatically generated

The result:

A screenshot of a computer

Description automatically generated A screenshot of a computer screen

Description automatically generated

# **11. Creation of tables in SQL and insertion of coherent data into each of them (at least 5 records in each non-associative table; at least 10 records in associative tables; a maximum of 30 records in each table).**

* **ORAȘ (CITY)**

CREATE TABLE ORAS(

ID\_Oras NUMBER(5) CONSTRAINT PK\_ORAS PRIMARY KEY,

Nume\_Oras VARCHAR2(100) CONSTRAINT NN\_Nume\_Oras NOT NULL,

Regiune VARCHAR2(100),

Cod\_Postal VARCHAR2(10)

);

INSERT INTO ORAS (ID\_Oras, Nume\_Oras, Regiune, Cod\_Postal) VALUES (2793, 'Bucuresti', 'Muntenia', '302813');

INSERT INTO ORAS (ID\_Oras, Nume\_Oras, Regiune, Cod\_Postal) VALUES (3292, 'Cluj-Napoca', 'Transilvania', '481462');

INSERT INTO ORAS (ID\_Oras, Nume\_Oras, Regiune, Cod\_Postal) VALUES (1761, 'Timisoara', 'Banat', '379430');

INSERT INTO ORAS (ID\_Oras, Nume\_Oras, Regiune, Cod\_Postal) VALUES (3829, 'Iasi', 'Moldova', '705227');

INSERT INTO ORAS (ID\_Oras, Nume\_Oras, Regiune, Cod\_Postal) VALUES (5386, 'Constanta', 'Dobrogea', '947832');

select \* from ORAS;

savepoint save1;

A close-up of a white background

Description automatically generated

A screenshot of a computer

Description automatically generated

* **MAGAZIN (STORE)**

CREATE TABLE MAGAZIN(

ID\_Magazin NUMBER(5) CONSTRAINT PK\_MAGAZIN PRIMARY KEY,

Nume\_Magazin VARCHAR2(100) CONSTRAINT NN\_Nume\_Magazin NOT NULL,

Adresa\_Magazin VARCHAR2(100),

ID\_Oras NUMBER(5) CONSTRAINT FK\_MAGAZIN\_ORAS REFERENCES ORAS(ID\_Oras)

);

ALTER TABLE MAGAZIN

ADD CONSTRAINT UK\_Nume\_Magazin UNIQUE (Nume\_Magazin);

ALTER TABLE MAGAZIN

DROP CONSTRAINT FK\_MAGAZIN\_ORAS;

ALTER TABLE MAGAZIN

ADD CONSTRAINT FK\_MAGAZIN\_ORAS

FOREIGN KEY (ID\_Oras)

REFERENCES ORAS(ID\_Oras)

ON DELETE SET NULL;

INSERT INTO MAGAZIN (ID\_Magazin, Nume\_Magazin, Adresa\_Magazin, ID\_Oras) VALUES (17634, 'Sevora Central', 'Str. Libertatii nr. 12', 2793);

INSERT INTO MAGAZIN (ID\_Magazin, Nume\_Magazin, Adresa\_Magazin, ID\_Oras) VALUES (28637, 'Sevora Nord', 'Str. Garii nr. 5', 3292);

INSERT INTO MAGAZIN (ID\_Magazin, Nume\_Magazin, Adresa\_Magazin, ID\_Oras) VALUES (34902, 'Sevora Sud', 'Str. Florilor nr. 8', 2793);

INSERT INTO MAGAZIN (ID\_Magazin, Nume\_Magazin, Adresa\_Magazin, ID\_Oras) VALUES (47644, 'Sevora Est', 'Str. Mihai Viteazu nr. 20', 5386);

INSERT INTO MAGAZIN (ID\_Magazin, Nume\_Magazin, Adresa\_Magazin, ID\_Oras) VALUES (52831, 'Sevora Vest', 'Str. Victoriei nr. 15', 2793);

select \* from MAGAZIN;

A screenshot of a computer

Description automatically generated

A close-up of some words

Description automatically generated

A screenshot of a computer

Description automatically generated

* **MANAGER**

CREATE TABLE MANAGER(

ID\_Manager NUMBER(5) CONSTRAINT PK\_MANAGER PRIMARY KEY,

Nume\_Manager VARCHAR2(100) CONSTRAINT NN\_Nume\_Manager NOT NULL,

Prenume\_Manager VARCHAR2(100) CONSTRAINT NN\_Prenume\_Manager NOT NULL,

Email\_Manager VARCHAR2(100) CONSTRAINT NN\_Email\_Manager NOT NULL,

Data\_Angajare DATE CONSTRAINT NN\_Data\_Angajare NOT NULL,

ID\_Magazin NUMBER(5) CONSTRAINT FK\_MANAGER\_MAGAZIN REFERENCES MAGAZIN(ID\_Magazin)

);

ALTER TABLE MANAGER

DROP CONSTRAINT FK\_MANAGER\_MAGAZIN;

ALTER TABLE MANAGER

ADD CONSTRAINT FK\_MANAGER\_MAGAZIN

FOREIGN KEY (ID\_Magazin)

REFERENCES MAGAZIN(ID\_Magazin)

ON DELETE SET NULL;

INSERT INTO MANAGER (ID\_Manager, Nume\_Manager, Prenume\_Manager, Email\_Manager, Data\_Angajare, ID\_Magazin) VALUES (100, 'Popescu', 'Ion', 'ion.popescu@magazin.ro', DATE '2020-01-01', 47644);

INSERT INTO MANAGER (ID\_Manager, Nume\_Manager, Prenume\_Manager, Email\_Manager, Data\_Angajare, ID\_Magazin) VALUES (200, 'Ionescu', 'Maria', 'maria.ionescu@magazin.ro', DATE '2020-02-01', 28637);

INSERT INTO MANAGER (ID\_Manager, Nume\_Manager, Prenume\_Manager, Email\_Manager, Data\_Angajare, ID\_Magazin) VALUES (300, 'Georgescu', 'Andrei', 'andrei.georgescu@magazin.ro', DATE '2020-03-01', 52831);

INSERT INTO MANAGER (ID\_Manager, Nume\_Manager, Prenume\_Manager, Email\_Manager, Data\_Angajare, ID\_Magazin) VALUES (400, 'Vasilescu', 'Elena', 'elena.vasilescu@magazin.ro', DATE '2020-04-01', 17634);

INSERT INTO MANAGER (ID\_Manager, Nume\_Manager, Prenume\_Manager, Email\_Manager, Data\_Angajare, ID\_Magazin) VALUES (500, 'Dumitrescu', 'Cristina', 'cristina.dumitrescu@magazin.ro', DATE '2020-05-01', 34902);

select \* from MANAGER;

A screenshot of a computer code

Description automatically generated

A white background with black text

Description automatically generated

A screen shot of a computer

Description automatically generated

* **ANGAJAT (EMPLOYEE)**

CREATE TABLE ANGAJAT(

ID\_Angajat NUMBER(5) CONSTRAINT PK\_ANGAJAT PRIMARY KEY,

Nume\_Angajat VARCHAR2(100) CONSTRAINT NN\_Nume\_Angajat NOT NULL,

Prenume\_Angajat VARCHAR2(100) CONSTRAINT NN\_Prenume\_Angajat NOT NULL,

Email\_Angajat VARCHAR2(100) CONSTRAINT NN\_Email\_Angajat NOT NULL,

Data\_Angajare\_A DATE CONSTRAINT NN\_Data\_Angajare\_A NOT NULL,

ID\_Manager NUMBER(5) CONSTRAINT FK\_ANGAJAT\_MANAGER REFERENCES MANAGER(ID\_Manager)

);

ALTER TABLE ANGAJAT DROP CONSTRAINT FK\_ANGAJAT\_MANAGER;

ALTER TABLE ANGAJAT

ADD CONSTRAINT FK\_ANGAJAT\_MANAGER FOREIGN KEY (ID\_Manager)

REFERENCES MANAGER(ID\_Manager) ON DELETE SET NULL;

INSERT INTO ANGAJAT (ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Data\_Angajare\_A, ID\_Manager) VALUES

(101, 'Mihai', 'Alexandru', 'alexandru.mihai@magazin.ro', DATE '2021-01-14', 100);

INSERT INTO ANGAJAT (ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Data\_Angajare\_A, ID\_Manager) VALUES

(102, 'Popa', 'Roxana', 'roxana.popa@magazin.ro', DATE '2021-02-07', 200);

INSERT INTO ANGAJAT (ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Data\_Angajare\_A, ID\_Manager) VALUES

(103, 'Stan', 'Daniel', 'daniel.stan@magazin.ro', DATE '2021-03-21', 300);

INSERT INTO ANGAJAT (ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Data\_Angajare\_A, ID\_Manager) VALUES

(104, 'Iliescu', 'Ioana', 'ioana.iliescu@magazin.ro', DATE '2021-04-20', 400);

INSERT INTO ANGAJAT (ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Data\_Angajare\_A, ID\_Manager) VALUES

(105, 'Radu', 'Marius', 'marius.radu@magazin.ro', DATE '2021-05-10', 500);

INSERT INTO ANGAJAT (ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Data\_Angajare\_A, ID\_Manager) VALUES

(106, 'Neagu', 'Adriana', 'adriana.neagu@magazin.ro', DATE '2021-06-24', 300);

INSERT INTO ANGAJAT (ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Data\_Angajare\_A, ID\_Manager) VALUES

(107, 'Lazar', 'Cosmin', 'cosmin.lazar@magazin.ro', DATE '2021-07-19', 400);

INSERT INTO ANGAJAT (ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Data\_Angajare\_A, ID\_Manager) VALUES

(108, 'Voicu', 'Gabriel', 'gabriel.voicu@magazin.ro', DATE '2021-08-12', 400);

INSERT INTO ANGAJAT (ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Data\_Angajare\_A, ID\_Manager) VALUES

(109, 'Marin', 'Ana', 'ana.marin@magazin.ro', DATE '2021-09-21', 200);

INSERT INTO ANGAJAT (ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Email\_Angajat, Data\_Angajare\_A, ID\_Manager) VALUES

(110, 'Constantin', 'Lucian', 'lucian.constantin@magazin.ro', DATE '2021-10-25', 500);

select \* from ANGAJAT;

A screenshot of a computer

Description automatically generated

A white background with black text

Description automatically generated

A screen shot of a computer

Description automatically generated

* **PRODUS (PRODUCT)**

CREATE TABLE PRODUS(

ID\_Produs NUMBER(5) CONSTRAINT PK\_PRODUS PRIMARY KEY,

Nume\_Produs VARCHAR2(100) CONSTRAINT NN\_Nume\_Produs NOT NULL,

Brand VARCHAR2(100),

Pret NUMBER(10, 2) CONSTRAINT NN\_Pret NOT NULL,

Stoc NUMBER(5) CONSTRAINT NN\_Stoc NOT NULL

);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (15327, 'Fond de ten', 'Estee Lauder', 190.50, 100);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (78432, 'Ruj Color Sensational', 'Maybelline', 40.00, 200);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (31475, 'Mascara Lash Paradise', 'LOreal', 45.75, 150);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (42435, 'Pudra Translucenta', 'Clinique', 130.00, 300);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (53113, 'Paleta de Farduri', 'Urban Decay', 275.00, 250);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (63782, 'Sampon Elseve', 'LOreal', 22.00, 120);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (78245, 'Balsam de Par Hydra Source', 'Biolage', 62.50, 180);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (89213, 'Spray Fixativ Elnett', 'LOreal', 18.00, 140);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (93456, 'Fond de ten Double Wear', 'Estee Lauder', 195.00, 130);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (10234, 'Creion de Ochi Super Liner', 'LOreal', 17.00, 160);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (11324, 'Ruj Matte Ink', 'Maybelline', 35.00, 170);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (12456, 'Paleta de Contur', 'Anastasia Beverly Hills', 245.00, 90);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (13579, 'Ser pentru Par Frizz Ease', 'John Frieda', 125.50, 110);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (14789, 'Mascara Better Than Sex', 'Too Faced', 159.00, 100);

INSERT INTO PRODUS (ID\_Produs, Nume\_Produs, Brand, Pret, Stoc) VALUES (15987, 'Creion de Sprancene Brow Wiz', 'Anastasia Beverly Hills', 83.00, 130);

select \* from PRODUS;

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

* **COMANDĂ (ORDER)**

CREATE TABLE COMANDA(

ID\_Comanda NUMBER(10) CONSTRAINT PK\_COMANDA PRIMARY KEY,

Data\_Comanda DATE CONSTRAINT NN\_Data\_Comanda NOT NULL,

Data\_Livrare DATE,

Valoare\_Totala NUMBER(10, 2),

ID\_Magazin NUMBER(5) CONSTRAINT FK\_COMANDA\_MAGAZIN REFERENCES MAGAZIN(ID\_Magazin),

ID\_Produs NUMBER(5) CONSTRAINT FK\_COMANDA\_PRODUS REFERENCES PRODUS(ID\_Produs)

);

ALTER TABLE COMANDA

DROP CONSTRAINT FK\_COMANDA\_MAGAZIN;

ALTER TABLE COMANDA

ADD CONSTRAINT FK\_COMANDA\_MAGAZIN FOREIGN KEY (ID\_Magazin)

REFERENCES MAGAZIN(ID\_Magazin)ON DELETE SET NULL;

ALTER TABLE COMANDA

DROP CONSTRAINT FK\_COMANDA\_PRODUS;

ALTER TABLE COMANDA

ADD CONSTRAINT FK\_COMANDA\_PRODUS FOREIGN KEY (ID\_Produs)

REFERENCES PRODUS(ID\_Produs)ON DELETE SET NULL;

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (6731983147, DATE '2022-01-15', DATE '2022-01-20', 1050.00, 28637, 78245);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (4729864671, DATE '2022-01-18', DATE '2022-01-23', 4000.00, 34902, 53113);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (3287346409, DATE '2023-01-21', DATE '2023-01-26', 2362.50, 52831, 89213);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (4682289382, DATE '2023-01-25', DATE '2023-01-30', 9000.00, 52831, 42435);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (5643764201, DATE '2023-01-28', DATE '2023-02-02', 6250.00, 47644, 13579);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (6721983197, DATE '2023-02-01', DATE '2023-02-06', 3200.00, 17634, 10234);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (5732964683, DATE '2023-02-03', DATE '2023-02-08', 2750.00, 34902, 31475);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (4987346490, DATE '2023-02-05', DATE '2023-02-10', 3400.00, 17634, 78432);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (4182289352, DATE '2023-02-08', DATE '2023-02-13', 2850.00, 34902, 93456);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (5343764210, DATE '2023-02-10', DATE '2023-02-15', 1500.00, 28637, 11324);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (6021983147, DATE '2024-02-12', DATE '2024-02-17', 4700.00, 47644, 12456);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (5729864672, DATE '2024-02-14', DATE '2024-02-19', 5900.00, 28637, 15327);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (5287346410, DATE '2024-02-16', DATE '2024-02-21', 4100.00, 52831, 63782);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (4782289383, DATE '2024-02-18', DATE '2024-02-23', 3500.00, 17634, 14789);

INSERT INTO COMANDA (ID\_Comanda, Data\_Comanda, Data\_Livrare, Valoare\_Totala, ID\_Magazin, ID\_Produs) VALUES (5643764211, DATE '2024-02-20', DATE '2024-02-25', 4500.00, 17634, 15987);

select \* from COMANDA;

A screenshot of a computer

Description automatically generated

A group of black text

Description automatically generated

A screenshot of a computer

Description automatically generated

* **FURNIZOR (SUPPLIER)**

CREATE TABLE FURNIZOR(

ID\_Furnizor NUMBER(10) CONSTRAINT PK\_FURNIZOR PRIMARY KEY,

Nume\_Furnizor VARCHAR2(100) CONSTRAINT NN\_Nume\_Furnizor NOT NULL,

Adresa\_Furnizor VARCHAR2(200),

Email\_Furnizor VARCHAR2(100),

Telefon VARCHAR2(15)

);

INSERT INTO FURNIZOR (ID\_Furnizor, Nume\_Furnizor, Adresa\_Furnizor, Email\_Furnizor, Telefon) VALUES

(378261, 'Furnizor Makeup', 'Str. Progresului nr. 134', 'contact@furnizormakeup.ro', '0721378261');

INSERT INTO FURNIZOR (ID\_Furnizor, Nume\_Furnizor, Adresa\_Furnizor, Email\_Furnizor, Telefon) VALUES

(402864, 'Furnizor Creme Fata', 'Str. Tudor Vladimirescu nr. 29', 'contact@furnizorcremefata.ro', '0721402864');

INSERT INTO FURNIZOR (ID\_Furnizor, Nume\_Furnizor, Adresa\_Furnizor, Email\_Furnizor, Telefon) VALUES

(729462, 'Furnizor Creme Corp', 'Str. Capitan Aviator Alexandru Serbanescu nr. 50', 'contact@furnizorcremecorp.ro', '0721729462');

INSERT INTO FURNIZOR (ID\_Furnizor, Nume\_Furnizor, Adresa\_Furnizor, Email\_Furnizor, Telefon) VALUES

(382394, 'Furnizor Produse Par', 'Str. Nicolae Caramfil nr. 25', 'contact@furnizorprodusepar.ro', '0721382394');

INSERT INTO FURNIZOR (ID\_Furnizor, Nume\_Furnizor, Adresa\_Furnizor, Email\_Furnizor, Telefon) VALUES

(738621, 'Furnizor Produse Styling', 'Str. Academiei nr. 28', 'contact@furnizorprodusestyling.ro', '0721738621');

select \* from FURNIZOR;

A close up of a text

Description automatically generated

A close-up of a computer screen

Description automatically generated

* **VÂNZARE (SALE)**

CREATE TABLE VANZARE(

ID\_Vanzare NUMBER(10) CONSTRAINT PK\_VANZARE PRIMARY KEY,

Data\_Vanzare DATE CONSTRAINT NN\_Data\_Vanzare NOT NULL,

Suma\_Totala NUMBER(10, 2) CONSTRAINT NN\_Suma\_Totala NOT NULL,

ID\_Produs NUMBER(5) CONSTRAINT FK\_VANZARE\_PRODUS REFERENCES PRODUS(ID\_Produs)

);

ALTER TABLE VANZARE

DROP CONSTRAINT FK\_VANZARE\_PRODUS;

ALTER TABLE VANZARE

ADD CONSTRAINT FK\_VANZARE\_PRODUS FOREIGN KEY (ID\_Produs)

REFERENCES PRODUS(ID\_Produs)ON DELETE SET NULL;

INSERT INTO VANZARE (ID\_Vanzare, Data\_Vanzare, Suma\_Totala, ID\_Produs) VALUES (17854032, DATE '2024-02-01', 62.50, 78245);

INSERT INTO VANZARE (ID\_Vanzare, Data\_Vanzare, Suma\_Totala, ID\_Produs) VALUES (25684596, DATE '2024-02-05', 18.00, 89213);

INSERT INTO VANZARE (ID\_Vanzare, Data\_Vanzare, Suma\_Totala, ID\_Produs) VALUES (32153543, DATE '2024-02-10', 45.75, 31475);

INSERT INTO VANZARE (ID\_Vanzare, Data\_Vanzare, Suma\_Totala, ID\_Produs) VALUES (48046263, DATE '2024-02-15', 125.50, 13579);

INSERT INTO VANZARE (ID\_Vanzare, Data\_Vanzare, Suma\_Totala, ID\_Produs) VALUES (55315628, DATE '2024-02-20', 40.00, 78432);

INSERT INTO VANZARE (ID\_Vanzare, Data\_Vanzare, Suma\_Totala, ID\_Produs) VALUES (63415764, DATE '2024-02-25', 35.00, 11324);

INSERT INTO VANZARE (ID\_Vanzare, Data\_Vanzare, Suma\_Totala, ID\_Produs) VALUES (74523695, DATE '2024-03-01', 245.00, 12456);

INSERT INTO VANZARE (ID\_Vanzare, Data\_Vanzare, Suma\_Totala, ID\_Produs) VALUES (85780893, DATE '2024-03-05', 22.00, 63782);

INSERT INTO VANZARE (ID\_Vanzare, Data\_Vanzare, Suma\_Totala, ID\_Produs) VALUES (94696243, DATE '2024-03-10', 195.00, 93456);

INSERT INTO VANZARE (ID\_Vanzare, Data\_Vanzare, Suma\_Totala, ID\_Produs) VALUES (10826538, DATE '2024-03-15', 190.50, 15327);

select \* from VANZARE;

A screenshot of a computer

Description automatically generated

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

* **CLIENT (CUSTOMER)**

CREATE TABLE CLIENT(

ID\_Client NUMBER(5) CONSTRAINT PK\_CLIENT PRIMARY KEY,

Nume\_Client VARCHAR2(100) CONSTRAINT NN\_Nume\_Client NOT NULL,

Prenume\_Client VARCHAR2(100) CONSTRAINT NN\_Prenume\_Client NOT NULL

);

INSERT INTO CLIENT (ID\_Client, Nume\_Client, Prenume\_Client) VALUES (1965, 'Moldovan', 'Cristian');

INSERT INTO CLIENT (ID\_Client, Nume\_Client, Prenume\_Client) VALUES (2439, 'Popescu', 'Alina');

INSERT INTO CLIENT (ID\_Client, Nume\_Client, Prenume\_Client) VALUES (3576, 'Stan', 'Vasile');

INSERT INTO CLIENT (ID\_Client, Nume\_Client, Prenume\_Client) VALUES (4920, 'Ionescu', 'Mihaela');

INSERT INTO CLIENT (ID\_Client, Nume\_Client, Prenume\_Client) VALUES (5864, 'Dumitru', 'Andrei');

INSERT INTO CLIENT (ID\_Client, Nume\_Client, Prenume\_Client) VALUES (6326, 'Georgescu', 'Elena');

INSERT INTO CLIENT (ID\_Client, Nume\_Client, Prenume\_Client) VALUES (7983, 'Radu', 'Mihai');

INSERT INTO CLIENT (ID\_Client, Nume\_Client, Prenume\_Client) VALUES (8028, 'Marin', 'Ana');

INSERT INTO CLIENT (ID\_Client, Nume\_Client, Prenume\_Client) VALUES (9254, 'Pavel', 'Ioan');

INSERT INTO CLIENT (ID\_Client, Nume\_Client, Prenume\_Client) VALUES (1029, 'Barbu', 'Laura');

select \* from CLIENT;

A screenshot of a computer code

Description automatically generated

A screenshot of a table

Description automatically generated

* **RECEPȚIONEAZĂ (RECEIVES)**

CREATE TABLE RECEPTIONEAZA(

ID\_Furnizor NUMBER(10),

ID\_Comanda NUMBER(10),

Data\_Primire DATE CONSTRAINT NN\_Data\_Primire NOT NULL,

Stare\_Comanda VARCHAR2(50),

Cantitate NUMBER(5),

CONSTRAINT PK\_RECEPTIONEAZA PRIMARY KEY (ID\_Furnizor, ID\_Comanda),

CONSTRAINT FK\_RECEPTIONEAZA\_FURNIZOR FOREIGN KEY (ID\_Furnizor) REFERENCES FURNIZOR(ID\_Furnizor),

CONSTRAINT FK\_RECEPTIONEAZA\_COMANDA FOREIGN KEY (ID\_Comanda) REFERENCES COMANDA(ID\_Comanda)

);

ALTER TABLE RECEPTIONEAZA DROP CONSTRAINT FK\_RECEPTIONEAZA\_FURNIZOR;

ALTER TABLE RECEPTIONEAZA

ADD CONSTRAINT FK\_RECEPTIONEAZA\_FURNIZOR FOREIGN KEY (ID\_Furnizor)

REFERENCES FURNIZOR(ID\_Furnizor) ON DELETE SET NULL;

ALTER TABLE RECEPTIONEAZA DROP CONSTRAINT FK\_RECEPTIONEAZA\_COMANDA;

ALTER TABLE RECEPTIONEAZA

ADD CONSTRAINT FK\_RECEPTIONEAZA\_COMANDA FOREIGN KEY (ID\_Comanda)

REFERENCES COMANDA(ID\_Comanda) ON DELETE SET NULL;

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (729462, 6731983147, DATE '2022-01-15', 'Livrata', 100);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (378261, 4729864671, DATE '2022-01-18', 'Livrata', 200);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (382394, 3287346409, DATE '2023-01-21', 'Livrata', 150);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (402864, 4682289382, DATE '2023-01-25', 'Anulata', 300);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (738621, 5643764201, DATE '2023-01-28', 'Livrata', 250);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (378261, 6721983197, DATE '2023-02-01', 'Livrata', 100);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (729462, 5732964683, DATE '2023-02-03', 'Livrata', 200);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (382394, 4987346490, DATE '2023-02-05', 'Livrata', 150);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (402864, 4182289352, DATE '2023-02-08', 'Anulata', 300);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (738621, 5343764210, DATE '2023-02-10', 'Livrata', 250);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (378261, 6021983147, DATE '2024-02-12', 'Livrata', 100);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (729462, 5729864672, DATE '2024-02-14', 'Anulata', 200);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (402864, 5287346410, DATE '2024-02-16', 'Livrata', 150);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (382394, 4782289383, DATE '2024-02-18', 'Livrata', 250);

INSERT INTO RECEPTIONEAZA (ID\_Furnizor, ID\_Comanda, Data\_Primire, Stare\_Comanda, Cantitate) VALUES (738621, 5643764211, DATE '2024-02-20', 'Anulata', 150);

select \* from RECEPTIONEAZA;

A screenshot of a computer

Description automatically generated

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

* **LUCREAZĂ (WORKS)**

CREATE TABLE LUCREAZA(

ID\_Angajat NUMBER(5),

ID\_Magazin NUMBER(5),

Pozitie VARCHAR2(100),

CONSTRAINT PK\_LUCREAZA PRIMARY KEY (ID\_Angajat, ID\_Magazin),

CONSTRAINT FK\_LUCREAZA\_ANGAJAT FOREIGN KEY (ID\_Angajat) REFERENCES ANGAJAT(ID\_Angajat),

CONSTRAINT FK\_LUCREAZA\_MAGAZIN FOREIGN KEY (ID\_Magazin) REFERENCES MAGAZIN(ID\_Magazin)

);

ALTER TABLE LUCREAZA DROP CONSTRAINT FK\_LUCREAZA\_ANGAJAT;

ALTER TABLE LUCREAZA

ADD CONSTRAINT FK\_LUCREAZA\_ANGAJAT FOREIGN KEY (ID\_Angajat)

REFERENCES ANGAJAT(ID\_Angajat) ON DELETE SET NULL;

ALTER TABLE LUCREAZA DROP CONSTRAINT FK\_LUCREAZA\_MAGAZIN;

ALTER TABLE LUCREAZA

ADD CONSTRAINT FK\_LUCREAZA\_MAGAZIN FOREIGN KEY (ID\_Magazin)

REFERENCES MAGAZIN(ID\_Magazin) ON DELETE SET NULL;

INSERT INTO LUCREAZA (ID\_Angajat, ID\_Magazin, Pozitie) VALUES (101, 47644, 'Vanzator');

INSERT INTO LUCREAZA (ID\_Angajat, ID\_Magazin, Pozitie) VALUES (102, 28637, 'Casier');

INSERT INTO LUCREAZA (ID\_Angajat, ID\_Magazin, Pozitie) VALUES (103, 52831, 'Vanzator');

INSERT INTO LUCREAZA (ID\_Angajat, ID\_Magazin, Pozitie) VALUES (104, 17634, 'Casier');

INSERT INTO LUCREAZA (ID\_Angajat, ID\_Magazin, Pozitie) VALUES (105, 34902, 'Vanzator');

INSERT INTO LUCREAZA (ID\_Angajat, ID\_Magazin, Pozitie) VALUES (106, 52831, 'Casier');

INSERT INTO LUCREAZA (ID\_Angajat, ID\_Magazin, Pozitie) VALUES (107, 17634, 'Vanzator');

INSERT INTO LUCREAZA (ID\_Angajat, ID\_Magazin, Pozitie) VALUES (108, 17634, 'Casier');

INSERT INTO LUCREAZA (ID\_Angajat, ID\_Magazin, Pozitie) VALUES (109, 28637, 'Vanzator');

INSERT INTO LUCREAZA (ID\_Angajat, ID\_Magazin, Pozitie) VALUES (110, 34902, 'Casier');

select \* from LUCREAZA;

A screenshot of a computer program

Description automatically generated

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

* **VINDE (SELLS)**

CREATE TABLE VINDE(

ID\_Angajat NUMBER(5),

ID\_Vanzare NUMBER(10),

ID\_Client NUMBER(5),

CONSTRAINT PK\_VINDE PRIMARY KEY (ID\_Angajat, ID\_Vanzare, ID\_Client),

CONSTRAINT FK\_VINDE\_ANGAJAT FOREIGN KEY (ID\_Angajat) REFERENCES ANGAJAT(ID\_Angajat),

CONSTRAINT FK\_VINDE\_VANZARE FOREIGN KEY (ID\_Vanzare) REFERENCES VANZARE(ID\_Vanzare),

CONSTRAINT FK\_VINDE\_CLIENT FOREIGN KEY (ID\_Client) REFERENCES CLIENT(ID\_Client)

);

ALTER TABLE VINDE DROP CONSTRAINT FK\_VINDE\_ANGAJAT;

ALTER TABLE VINDE

ADD CONSTRAINT FK\_VINDE\_ANGAJAT FOREIGN KEY (ID\_Angajat)

REFERENCES ANGAJAT(ID\_Angajat) ON DELETE SET NULL;

ALTER TABLE VINDE DROP CONSTRAINT FK\_VINDE\_VANZARE;

ALTER TABLE VINDE

ADD CONSTRAINT FK\_VINDE\_VANZARE FOREIGN KEY (ID\_Vanzare)

REFERENCES VANZARE(ID\_Vanzare) ON DELETE SET NULL;

ALTER TABLE VINDE DROP CONSTRAINT FK\_VINDE\_CLIENT;

ALTER TABLE VINDE

ADD CONSTRAINT FK\_VINDE\_CLIENT FOREIGN KEY (ID\_Client)

REFERENCES CLIENT(ID\_Client) ON DELETE SET NULL;

INSERT INTO VINDE (ID\_Angajat, ID\_Vanzare, ID\_Client) VALUES (101, 94696243, 3576);

INSERT INTO VINDE (ID\_Angajat, ID\_Vanzare, ID\_Client) VALUES (101, 55315628, 6326);

INSERT INTO VINDE (ID\_Angajat, ID\_Vanzare, ID\_Client) VALUES (102, 32153543, 7983);

INSERT INTO VINDE (ID\_Angajat, ID\_Vanzare, ID\_Client) VALUES (101, 74523695, 1965);

INSERT INTO VINDE (ID\_Angajat, ID\_Vanzare, ID\_Client) VALUES (103, 10826538, 5864);

INSERT INTO VINDE (ID\_Angajat, ID\_Vanzare, ID\_Client) VALUES (109, 48046263, 1029);

INSERT INTO VINDE (ID\_Angajat, ID\_Vanzare, ID\_Client) VALUES (109, 85780893, 8028);

INSERT INTO VINDE (ID\_Angajat, ID\_Vanzare, ID\_Client) VALUES (104, 17854032, 4920);

INSERT INTO VINDE (ID\_Angajat, ID\_Vanzare, ID\_Client) VALUES (104, 25684596, 2439);

INSERT INTO VINDE (ID\_Angajat, ID\_Vanzare, ID\_Client) VALUES (105, 63415764, 9254);

select \* from VINDE;

A screenshot of a computer program

Description automatically generated

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

# **12. Formulate in natural language and implement 5 complex SQL queries that, in their entirety, will utilize the following elements:**

1. Synchronized subqueries involving at least 3 tables
2. Unsynchronized subqueries in the FROM clause
3. Grouping data, group functions, and filtering at the group level with unsynchronized subqueries (in the HAVING clause) involving at least 3 tables
4. Ordering and using NVL and DECODE functions in the same query
5. Using at least 2 string functions, 2 date functions, and one CASE expression
6. Using at least 1 query block (WITH clause)

**Exercise 1**: Select the ID, first name, and last name of the employee who made the most sales. Additionally, select the ID and name of the city where the store, in which this employee works, is located. If there are multiple employees with the same number of sales, select the first one in descending order by employee ID.

Solution:

with top\_angajati as (select ID\_Angajat, count(ID\_Vanzare) nr\_vanzari

from VINDE

group by ID\_Angajat

order by 2 desc, 1)

select ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, ID\_Oras, Nume\_Oras

from (select ta.ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, o.ID\_Oras, o.Nume\_Oras, nr\_vanzari

from top\_angajati ta join ANGAJAT a on ta.ID\_Angajat = a.ID\_Angajat

join LUCREAZA l on a.ID\_Angajat = l.ID\_Angajat

join MAGAZIN m on l.ID\_Magazin = m.ID\_Magazin

join ORAS o on m.ID\_Oras = o.ID\_Oras

order by nr\_vanzari desc, ID\_Angajat desc) top\_angajat

where rownum = 1;

A white paper with black text

Description automatically generated



This exercise meets the following requirements:

b) Uncorrelated subqueries in the FROM clause (from (...) top\_angajat).

c) Data grouping (group by ID\_Angajat) and group functions (e.g., count).

d) Ordering (order by 2 desc, 1).

f) Use of at least one query block (with top\_angajati as (...)).

**Exercise 2**: Select the code, full name, and a custom message for employees who made sales up to February 15, 2024, and whose first name starts with 'A', and select the code, full name, and a custom message for clients who made sales after February 15, 2024. For employees, the custom message should display the total sale amount for those whose last name ends in 'n', the product sold for those whose last name ends in 'u', and -1 in all other cases. For clients, the custom message should display the total sale amount for clients whose first name ends in 'a', the product sold for clients whose first name ends in 'n', and -1 in all other cases. The data should be displayed in ascending order by the person's name.

Solution:

select distinct a.ID\_Angajat, concat(concat(Nume\_Angajat, ' '), Prenume\_Angajat) Nume\_Complet, decode(substr(Nume\_Angajat, -1), 'n', Suma\_totala, 'u', ID\_Produs, nvl(null, -1)) mesaj\_particular

from ANGAJAT a join VINDE v on a.ID\_Angajat = v.ID\_Angajat

join VANZARE va on v.ID\_Vanzare = va.ID\_Vanzare

where va.Data\_Vanzare <= to\_date('15-FEB-2024', 'dd-mon-yyyy')

and upper(Prenume\_Angajat) like 'A%'

union

select distinct c.ID\_Client, concat(concat(Nume\_Client, ' '), Prenume\_Client) Nume\_Complet, decode(substr(Prenume\_Client, -1), 'a', Suma\_totala, 'n', ID\_Produs, nvl(null, -1)) mesaj\_particular

from CLIENT c join VINDE v on c.ID\_Client = v.ID\_Client

join VANZARE va on v.ID\_Vanzare = va.ID\_Vanzare

where to\_char(va.Data\_Vanzare, 'dd-MON-yyyy') >= to\_date('15-FEB-2024', 'dd-mon-yyyy')

order by 2;

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

This exercise satisfies the following requirements:

d) ordering (order by 2) and the use of NVL function (nvl(null, -1)) and DECODE function (decode(substr(Nume\_Angajat, -1), 'n', Suma\_totala, 'u', ID\_Produs, nvl(null, -1)) as mesaj\_particular) within the same query.

e) the use of at least two string functions (concat, upper) and one date function (to\_date('15-FEB-2024', 'dd-mon-yyyy')).

**Exercise 3:** Select the names of employees and the total number of sales they have made, for employees who have made total sales with a value greater than the average sales value across all cities. The results should be ordered by the total number of sales in descending order. Additionally, label the employees based on their performance as follows:  
• 'Ridicată' ('High') for those who have made more than 2 sales  
• 'Medie' (‘Medium') for those with 2 sales  
• 'Scăzută' ('Low') for those with less than 2 sales.

Solution:

select a.Nume\_Angajat, a.Prenume\_Angajat, count(v.ID\_Vanzare) as Total\_Vanzari, case when count(v.ID\_Vanzare) > 2 then 'Ridicata' when count(v.ID\_Vanzare) = 2 then 'Medie' else 'Scazuta' end as Performanta

from ANGAJAT a join VINDE v on a.ID\_Angajat = v.ID\_Angajat

join VANZARE va on v.ID\_Vanzare = va.ID\_Vanzare

group by a.Nume\_Angajat, a.Prenume\_Angajat

having sum(va.Suma\_Totala) > (select avg(t.Valoare\_Totala)

from (select sum(va1.Suma\_Totala) as Valoare\_Totala

from VINDE v1 join VANZARE va1 on v1.ID\_Vanzare = va1.ID\_Vanzare

join ANGAJAT a1 on v1.ID\_Angajat = a1.ID\_Angajat

group by v1.ID\_Angajat) t)

order by Total\_Vanzari desc;

A white background with black text

Description automatically generated

A close up of a computer screen

Description automatically generated

This exercise meets the following requirements:  
 b) Unsynchronized subqueries in the FROM clause  
 c) Data grouping, group functions, group-level filtering with unsynchronized subqueries (in the HAVING clause) involving at least 3 tables (within the same query)  
 d) Ordering (order by Total\_Vanzari desc) ………e) Use of a CASE expression

**Exercise 4:** Select the code, total value, status, and a particular message for orders that were placed at least 6 months ago and were placed by a store located in a city with at least 2 stores. For delivered orders, the particular message will display the name of the store to which the order was delivered, and for canceled orders, it will display the name of the supplier to whom the order was placed.

Solution:

select c.ID\_Comanda, c.Valoare\_Totala, r.Stare\_Comanda, case when r.Stare\_Comanda = 'Livrata' then m.Nume\_Magazin when r.Stare\_Comanda = 'Anulata' then f.Nume\_furnizor end as Nume\_magazin\_furnizor

from COMANDA c join RECEPTIONEAZA r on r.ID\_Comanda = c.ID\_Comanda

join FURNIZOR f on f.ID\_Furnizor = r.ID\_Furnizor

join MAGAZIN m on c.ID\_Magazin = m.ID\_Magazin

join ORAS o on m.ID\_Oras = o.ID\_Oras

where c.Data\_Comanda <= add\_months(sysdate, -6)

and o.ID\_Oras in (select ID\_Oras

from MAGAZIN

group by ID\_Oras

having count(ID\_Magazin) >= 2);

A white background with black text

Description automatically generated

A screen shot of a computer

Description automatically generated

This exercise meets the following requirements:

c) Data grouping, aggregate functions

e) Use of a date function (add\_months), a CASE expression

**Exercise 5:** Select the code, delivery date, and total value of orders placed by stores that have at least 2 employees and a manager who was hired before March 15, 2020, and whose last name contains the letter 'o'.

Solution:

select c.ID\_Comanda, c.Data\_Livrare, c.Valoare\_Totala

from COMANDA c join MAGAZIN m on c.ID\_Magazin = m.ID\_Magazin

where m.ID\_Magazin in (select l.ID\_Magazin

from LUCREAZA l join ANGAJAT a on l.ID\_Angajat = a.ID\_Angajat

join MANAGER mg on a.ID\_Manager = mg.ID\_Manager

where (select count(\*)

from LUCREAZA l2

where l2.ID\_Magazin = l.ID\_Magazin) >= 2

and mg.ID\_Magazin = m.ID\_Magazin

and mg.Data\_Angajare < to\_date('15-03-2020', 'DD-MM-YYYY')

and mg.Nume\_Manager like '%o%')

order by c.Data\_Livrare;

A white screen with black text

Description automatically generated

A screenshot of a calendar

Description automatically generated

This exercise meets the following requirements:

a) Synchronized subqueries involving at least 3 tables

e) Usage of a date function (to\_date)

# **13. Implementation of 3 operations for updating and deleting data using subqueries.**

**Exercise 1:** Update the position of employees in the LUCREAZĂ (WORKS) table to ‘Angajatul lunii’ ('Employee of the month') for employees whose total sales amount is greater than the average sales of all employees.

Solution:

update LUCREAZA

set Pozitie = 'Angajatul lunii'

where ID\_Angajat in (select ID\_Angajat

from (select ID\_Angajat, sum(Suma\_Totala) as total\_vanzari

from VINDE v join VANZARE va on v.ID\_Vanzare = va.ID\_Vanzare

group by ID\_Angajat) totaluri

where total\_vanzari > (select avg(total\_vanzari)

from (select sum(Suma\_Totala) as total\_vanzari

from VINDE v join VANZARE va on v.ID\_Vanzare = va.ID\_Vanzare

group by ID\_Angajat) medie\_totala));

A screenshot of a computer

Description automatically generated

A close-up of a text

Description automatically generated

**Exercise 2**: Update the email addresses of managers to the new format LastName\_Manager + ID\_Manager + @magazin + first digit of ID\_Manager + .ro for managers who oversee stores with at least 2 employees.

Solution:

update MANAGER

set Email\_Manager = concat(concat(concat(concat(lower(Nume\_Manager), ID\_Manager), '@magazin'), substr(ID\_Manager, 1, 1)), '.ro')

where ID\_Manager in (select m.ID\_Manager

from MANAGER m join MAGAZIN mg on m.ID\_Magazin = mg.ID\_Magazin

join LUCREAZA l on mg.ID\_Magazin = l.ID\_Magazin

group by m.ID\_Manager

having count(l.ID\_Angajat) >= 2);

A white background with black text

Description automatically generated

A black text on a white background

Description automatically generated

**Exercise 3:** Delete cities where there are no stores.

Solution:

delete from ORAS

where ID\_Oras not in (select ID\_Oras

from MAGAZIN);

rollback to save1;

A screenshot of a computer code

Description automatically generated

A black text on a white background

Description automatically generated

# **14. Creation of a complex view. Provide an example of a DML operation allowed on that view and an example of a DML operation not allowed.**

Let's assume we want to create a view that displays information about employees, their managers, and the stores where they work, for stores located in the city of Bucharest and for employees who have made at least one sale.

SQL code:

create view ViewMagazin as

select a.ID\_Angajat, a.Nume\_Angajat, a.Prenume\_Angajat, a.Data\_Angajare\_A, a.Email\_Angajat, m.ID\_Manager, m.Nume\_Manager, m.Prenume\_Manager, m.Data\_Angajare, mg.ID\_Magazin, mg.Nume\_Magazin

from ANGAJAT a join LUCREAZA l on a.ID\_Angajat = l.ID\_Angajat

join MAGAZIN mg on l.ID\_Magazin = mg.ID\_Magazin

join ORAS o on mg.ID\_Oras = o.ID\_Oras

join MANAGER m on mg.ID\_Magazin = m.ID\_Magazin

left outer join VINDE v on a.ID\_Angajat = v.ID\_Angajat

join VANZARE va on v.ID\_Vanzare = va.ID\_Vanzare

where lower(o.Nume\_Oras) = 'bucuresti'

group by a.ID\_Angajat, a.Nume\_Angajat, a.Prenume\_Angajat, a.Data\_Angajare\_A, a.Email\_Angajat, m.ID\_Manager, m.Nume\_Manager, m.Prenume\_Manager, m.Data\_Angajare, mg.ID\_Magazin, mg.Nume\_Magazin

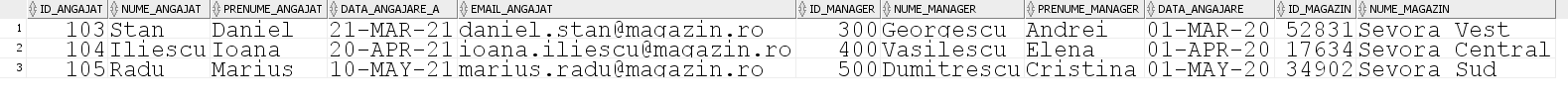
having count(va.ID\_Vanzare) >= 1;

A white background with black text

Description automatically generated

A black text on a white background

Description automatically generated



**Allowed DML Operation**  
 The SELECT operation below is permitted because it refers to a column that does not involve aggregation or a complex relationship.

SQL code:

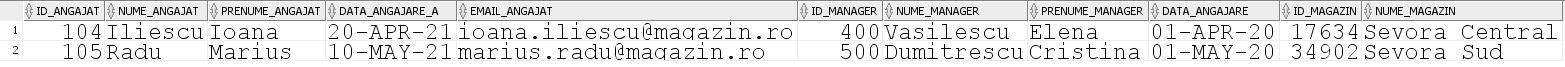
select \*

from ViewMagazin

where lower(Nume\_Angajat) like '%u%';

A close-up of a white background

Description automatically generated



The operation in the image displays all information from ViewMagazin for employees who have the letter 'u' in their last name.

## **Not Allowed DML Operation**

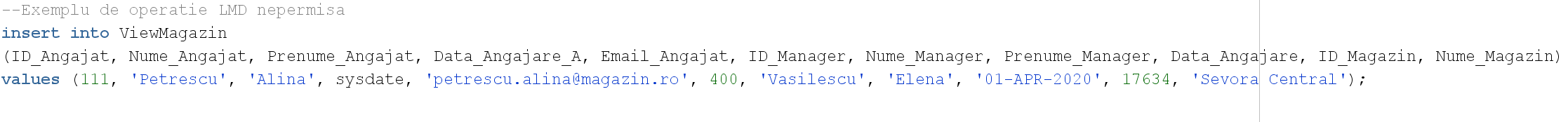
The INSERT operation below is not permitted because the view includes multiple tables and complex relationships.

SQL code:

insert into ViewMagazin

(ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Data\_Angajare\_A, Email\_Angajat, ID\_Manager, Nume\_Manager, Prenume\_Manager, Data\_Angajare, ID\_Magazin, Nume\_Magazin)

values (111, 'Petrescu', 'Alina', sysdate, 'petrescu.alina@magazin.ro', 400, 'Vasilescu', 'Elena', '01-APR-2020', 17634, 'Sevora Central');



A close-up of a letter

Description automatically generated

This INSERT is not permitted because the view includes columns from multiple tables (ANGAJAT, MAGAZIN, MANAGER), and the database cannot determine how to distribute these values among the base tables.

# **15. Formulate in natural language and implement in SQL: a query that uses the outer join operation on at least 4 tables, a query that uses the division operation, and a query that implements top-n analysis.**

**Outer join:** Select all stores, managers, employees, and orders, even if some links are missing, to have a complete view of all data.

Solution:

select nvl(nvl(nvl(mg.ID\_Magazin, m.ID\_Magazin), l.ID\_Magazin), c.ID\_Magazin) as ID\_Magazin, mg.Nume\_Magazin,

m.ID\_Manager, m.Nume\_Manager, m.Prenume\_Manager, a.ID\_Angajat, a.Nume\_Angajat, a.Prenume\_Angajat, c.ID\_Comanda, c.Valoare\_Totala

from MAGAZIN mg full outer join MANAGER m on mg.ID\_Magazin = m.ID\_Magazin

full outer join LUCREAZA l on mg.ID\_Magazin = l.ID\_Magazin or m.ID\_Magazin = l.ID\_Magazin

full outer join ANGAJAT a on l.ID\_Angajat = a.ID\_Angajat

full outer join COMANDA c on mg.ID\_Magazin = c.ID\_Magazin;

A close up of text

Description automatically generated

A screenshot of a computer program

Description automatically generated

**Division**: Select the employees who have made sales and work in stores located in the city with ID 2793.

Solution:

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

**Top-N Analysis**: Select the top 5 employees who have achieved the highest sales based on the total value of orders.

Solution:

select ID\_Angajat, Nume\_Angajat, Prenume\_Angajat, Total\_Vanzari

from (select a.ID\_Angajat, a.Nume\_Angajat, a.Prenume\_Angajat, sum(va.Suma\_Totala) as Total\_Vanzari

from ANGAJAT a join VINDE v ON a.ID\_Angajat = v.ID\_Angajat

join VANZARE va on v.ID\_Vanzare = va.ID\_Vanzare

group by a.ID\_Angajat, a.Nume\_Angajat, a.Prenume\_Angajat

order by Total\_Vanzari desc) top

WHERE rownum <= 5;

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated