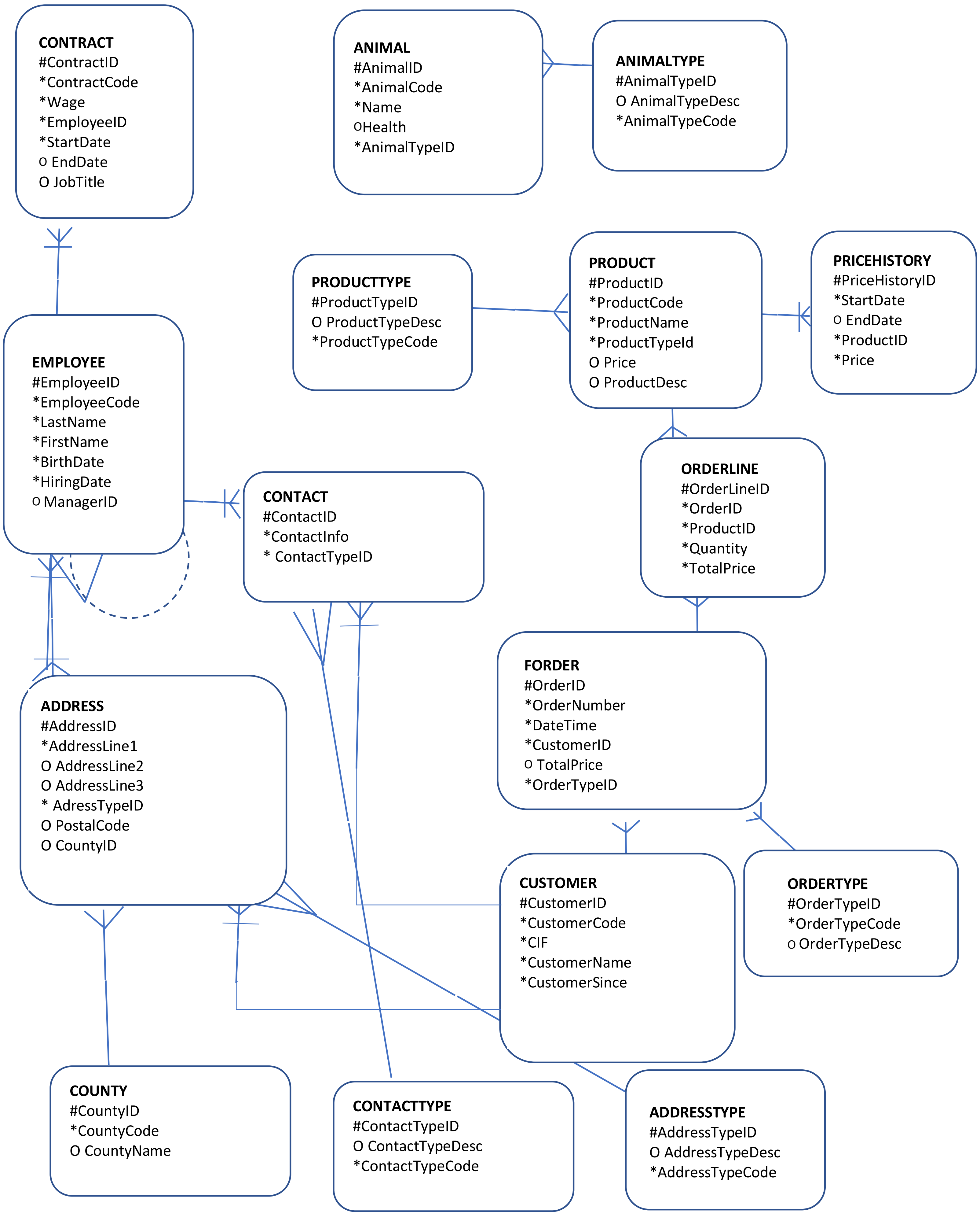
Proiect Ferma

Student: Florea Darius George

Specializare: IE ID, an 2, gr 2

In pagina urmatoare am create diagrama ERD pentru acest proiect. Ideea din spatele alegerilor facute este ca ferma sa poata fi usor de extins / restrains, oricand sa se poata adauga functionalitati noi in aplicatia din fata acestei baza de date, sa se adauge tipuri noi de contracte, animale, produse, comenzi. In aceeasi maniera am gandit si sistemul de “person details”, informatiile despre persoane fiind stocate in tabelele “Contact” si “Address”. Avem mai multe tipuri de addrese si contacte, deoarece oamenii vor sa fie contactati in mod diferit, avand mai multe adrese care servesc unor scopuri diferite. Daca vreodata se va dori sa se faca marketing, sa se trimita promotii catre clienti sau potentiali clienti, se vor alege adresele de tip „MAIL” pentru pliante spre exemplu.

Putem avea diferite tipuri de comenzi (ex. „ORDER” „ENQUIRY” „DISCARD” dupa cum se prezentase la curs), o comanda avand mai multe „ORDERLINE”, putand suporta mai multe produse, chiar diferite ca si tip. Avand acest tip de implementare ferma poate vinde atat carne, cat si blanuri, alte produse provenite de la animale (ex. Oua) sau chiar animale.


CONTRACT
#ContractID
*ContractCode
*Wage
*EmployeeID
*StartDate
o EndDate
O JobTitle
ANIMAL
#AnimalID
*AnimalCode
*Name
oHealth
*AnimalTypeID
ANIMALTYPE
#AnimalTypeID
O AnimalTypeDesc
*AnimalTypeCode
PRODUCTTYPE
#ProductTypeID
O ProductTypeDesc
*ProductTypeCode
PRODUCT
#ProductID
*ProductCode
*ProductName
*ProductTypeId
O Price
O ProductDesc
PRICEHISTORY
#PriceHistoryID
*StartDate
o EndDate
*ProductID
*Price
EMPLOYEE
#EmployeeID
*EmployeeCode
*LastName
*FirstName
*BirthDate
*HiringDate
o ManagerID
CONTACT
#ContactID
*ContactInfo
* ContactTypeID
ORDERLINE
#OrderLineID
*OrderID
*ProductID
*Quantity
*TotalPrice
FORDER
#OrderID
*OrderNumber
*DateTime
*CustomerID
o TotalPrice
*OrderTypeID
ADDRESS
#AddressID
*AddressLine1
O AddressLine2
O AddressLine3
* AdressTypeID
O PostalCode
O CountyID
CUSTOMER
#CustomerID
*CustomerCode
*CIF
*CustomerName
*CustomerSince
ORDERTYPE
#OrderTypeID
*OrderTypeCode
o OrderTypeDesc
COUNTY
#CountyID
*CountyCode
O CountyName
CONTACTTYPE
#ContactTypeID
O ContactTypeDesc
*ContactTypeCode
ADDRESSTYPE
#AddressTypeID
O AddressTypeDesc
*AddressTypeCode

Tabele de mapare

Maparea tabelelor este cat se poate de generica, vom folosi urmatoarele tabele:

CustomerAddress – va mapa tabelele Customer si Address, pentru moment vom considera aceasta relatie ca fiind “one to many”, un Customer poate avea mai multe adrese insa o adresa nu poate tine mai multi customeri, presupunand ca majoritatea clientilor vor fi firme, nu persoane fizice. Oricand se va dori schimbarea de la OTM la MTM se va putea face cu usurinta

EmployeeAddress – ca si CustomerAddress, doar ca aceasta relatie o vom considera “many to many”, intreaga familie poate sa lucreze la ferma noastra, avand aceeasi adresa, totodata o persoana poate avea mai multe adrese.

CustomerContact – va mapa tabelele Customer si Contact, o relatie OTM, un client poate avea mai multe contacte, dar un contact nu poate fi folosit de mai multi client

EmployeeContact – va functiona in aceeasi maniera in care functioneaza Customer Contact

Nu vom avea tabela de mapare intre Customer si Order, vom considera ca un order nu poate exista fara un customer iar acel field va fi Mandatory pentru order.

Structura acestor tabele va fi una simpla, vom avea un primary key (ex. CustomerAddressID) si 2 foregin keys, reprezentand primary key-urile celor doua tabele pe care dorim sa le mapam (ex. CustomerID AddressID).

Normalizarea bazei de date

Baza de date a fost gandita initial normalizata in forma 3NF (O relatie este in 3NF daca este in 2NF si nu exista dependente functionale tranzitive fata de cheia primara (nu exista dependente functionale intre atributele non-chei).

Definitia formei 2NF: O relatie este in 2NF daca este in 1NF si oricare dintre atributele care nu apartin cheii primare depinde complet (nu partial) de cheia primara

Definitia formei 1NF: Atributele unei relatii trebuie sa fie domenii de valori atomice adica sa nu mai poata fi descompuse. Nu trebuie sa existe grupuri repetitive.

Nu este necesara trecerea de la o forma la alta.

Acestea fiind spuse, diagrama de mai sus(impreuna cu tabelele de mapare, care nu sunt reprezentate in diagrama ERD) reprezinta o baza de date normalizata forma 3NF.

<u>Legaturile dintre tabele</u>

Vom omite legaturile deja descrise in sectiunea „Tabele de mapare”

Employee – Contract, o relatie one or many, ne vom folosi de un field(foreign key) EmployeeID prezent in tabela Contract, servind ca si field de legatura.

Address – County, o relatie one to many, bineinteles un judet poate avea mai multe adrese, dar o adresa un singur judet.

Contact – ContactType, o relatie one to many, un contact poate sa fie de un singur tip, insa un tip poate descrie mai multe contacte, ne vom folosi de fieldul (FK) ContactTypeID prezent in tabela Contact. Similar avem Animal – AnimalType, Product – ProductType, Address – AddressType, Order - OrderType.

Order – Orderline, o relatie one to many, un Order poate avea mai multe Orderlines, insa un Orderline nu poate apartine mai multor Orders, vom implementa un field OrderID(FK), in tabela OrderLine

Product – PriceHistory, o relatie one or many, un produs poate avea unul sau mai multe preturi in istoric, ne vom folosi de field-ul (FK) ProductID prezent in tabela PriceHistory pentru a evidentia acest lucru

Putem mentiona faptul ca la inceput am precizat ca aceasta ferma poate vinde si animale. Pentru a realiza acest lucru vom avea un ProductType „Animal” care va fi legat direct de unul sau mai multe produse. Produsele de tip „Animal” vor stoca in field-ul „ProductCode” valorile prezente in „AnimalCode”, astfel se va face legatura intre aceste doua tabele.

Product-OrderLine, o relatie one to many, un produs poate fi legat de mai multe OrderLines, insa un OrderLine nu poate avea mai mult de un singur produs. Daca se vor comanda mai multe produse simultan, vom avea un Order cu n OrderLine, n reprezentand numarul de produse dorite.

Explicatii legate de campuri, atribute

Pentru o colaborare cat mai usoara, in general, atat denumirile campurilor si a tabelelor trebuie sa fie simple dar totodata descriptive. Astfel ca spre exemplu avem tabela „Employee” care reprezinta un angajat, field-urile „FirstName” tradus in romana „prenumele” si „LastName” tradus in romana „numele de familie” sau „BirthDate” reprezentand data nasterii a angajatului. Similar au fost create si celelalte field-uri, in asa fel incat scopul si continutul lor poate fi dedus din numele atribuit.

Avem insa field-uri de tip „cod” care sunt obligatorii, insa nu sunt chei primare(ex. ProductCode). Scopul lor este de a oferi userilor aplicatiei un mod de a distinge inregistrarile multiple. Aceste field-uri actioneaza precum cheia primara, ele fiind unice, totodata avand un format user friendly. Spre exemplu avem un „Product” care are ProductId „2344523456”, un numar greu de retinut, iar „ProductCode” are valoarea „PIG_1122”, un cod mai simplu, usor de retinut, descriptiv. Totodata ca si best practice se incearca evitarea afisarii cheilor primare pe partea de frontend a aplicatiei, cu alte cuvinte userii nu ar trebui sa aiba access la ele, unul dintre motivele pentru care se face asta ar fi securitatea datelor, a aplicatiei.

Implementarea bazei de date si Constrangeri legate de campuri

Employee:

```
CREATE TABLE employee(EmployeeID NUMBER PRIMARY KEY,EmployeeCode VARCHAR(30)
NOT NULL UNIQUE,

FirstName VARCHAR(15) NOT NULL,

LastName VARCHAR(15) NOT NULL,

BirthDate DATE,

HiringDate DATE NOT NULL,

ManagerID NUMBER);
```

Pentru tabela de mai sus avem constrangeri de tip NOT NULL pe field-urile FirstName, LastName, HiringDate.

EmployeeCode avand de asemenea o strangere de UNIQUE value, deoarece nu pot fi recorduri cu acelasi cod.

AnimalType:

CREATE TABLE AnimalType(AnimalTypeID NUMBER PRIMARY KEY, AnimalTypeDesc VARCHAR(30), AnimalTypeCode VARCHAR(15) NOT NULL UNIQUE);

Vom avea un field optional, AnimalTypeDesc (ex de valori: porc rasa x, vitel pentru carne, oaie pentru lapte), intrucat este necesar sa avem codul NOT NULL si UNIQUE pentru buna functionare a aplicatiei.

ProductType:

CREATE TABLE ProductType(ProductTypeID NUMBER PRIMARY KEY, ProductTypeDesc VARCHAR(30), ProductTypeCode VARCHAR(15) NOT NULL UNIQUE);

Logica din spatele acestui tabel este similara cu cea descrisa la AnimalType.

ContactType:

CREATE TABLE ContactType(ContactTypeID NUMBER PRIMARY KEY, ContactTypeDesc VARCHAR(30), ContactTypeCode VARCHAR(15) NOT NULL UNIQUE);

Logica din spatele acestui tabel este similara cu cea descrisa la AnimalType.

AddressType:

CREATE TABLE AddressType(AddressTypeID NUMBER PRIMARY KEY, AddressTypeDesc VARCHAR(30), AddressTypeCode VARCHAR(15) NOT NULL UNIQUE);

Logica din spatele acestui tabel este similara cu cea descrisa la AnimalType.

OrderType:

CREATE TABLE OrderType(OrderTypeID NUMBER PRIMARY KEY, OrderTypeDesc VARCHAR(30), OrderTypeCode VARCHAR(15) NOT NULL UNIQUE);

Logica din spatele acestui tabel este similara cu cea descrisa la AnimalType.

County:

CREATE TABLE County (CountyID NUMBER PRIMARY KEY, CountyCode VARCHAR(5) NOT NULL UNIQUE, CountyName VARCHAR(15));

Nu vom constrange numele judetului cu „not null” deoarece codul va fi suficient pentru buna functionare a aplicatiei. Pentru ca aplicatia sa suporte si alte tari, eventual s-ar putea adauga un field „CountryID”, ca si foreign key acesta fiind NOT NULL, bineinteles.

Animal:

CREATE TABLE Animal(AnimalID NUMBER PRIMARY KEY, AnimalCode VARCHAR(15) NOT NULL UNIQUE, Name VARCHAR(15) NOT NULL, Health VARCHAR(35), AnimalTypeID NUMBER REFERENCES AnimalType(AnimalTypeID));

Vom introduce un foreign key „AnimalTypeID” care va indica tipul animalului, preluat din tabela AnimalType. Acest lucru s-a realizat folosind comanda AnimalTypeID NUMBER REFERENCES AnimalType(AnimalTypeID).

Contact:

CREATE TABLE Contact(ContactID NUMBER PRIMARY KEY, ContactInfo VARCHAR(30) NOT NULL UNIQUE, ContactTypeID NUMBER REFERENCES ContactType(ContactTypeID));

Similar cu Animal, s-a creat un foreign key ContactTypeID care sa descrie tipul contactului.(PHONE, EMAIL, MOBILE,FACEBOOK,LINKEDIN)

Address:

CREATE TABLE Address(AddressID NUMBER PRIMARY KEY, AddressLine1 VARCHAR(35) NOT NULL UNIQUE, AddressLine2 VARCHAR(35), AddressLine3 VARCHAR(35), AddressTypeID NUMBER REFERENCES AddressType(AddressTypeID), PostalCode CHAR(6), CountyID NUMBER REFERENCES County(CountyID));

AddressLine1 serveste adresei propriu zise, in unele tari adresa se scrie pe mai multe randuri(in mai multe campuri, ex UK), unele sisteme trimit adresa in 3 campuri, noi astfel avem optiunea fie sa le concatenam in addressline1 sau sa le stocam exact asa cum vin. Este o alegere care sa ne ofere flexibilitate.

In principiu codul postal are 6 caractere, daca ulterior va fi nevoie sa facem o modificare, se va putea extinde.

Avem doua foreign keys, una catre judet, iar cealalta catre tipul adresei.

Contract:

CREATE TABLE Contract(ContractID NUMBER PRIMARY KEY, ContractCode VARCHAR(25) NOT NULL UNIQUE, Wage NUMBER NOT NULL, EmployeeID NUMBER REFERENCES Employee(EmployeeID), StartDate DATE NOT NULL, EndDate DATE, JobTitle VARCHAR(15) NOT NULL);

EndDate este optional, deoarece unele contracte pot fi pe o perioada nederminata.

Avem un foreign key catre employee, fiecare contract este creat pentru un angajat. Alternativ se puteau crea angajatii un baza unor contracte deja existente.

Product:

CREATE TABLE Product(ProductID NUMBER PRIMARY KEY, ProductCode VARCHAR(15) NOT NULL UNIQUE, ProductName VARCHAR(30) NOT NULL, ProductTypeID REFERENCES ProductType(ProductTypeID), Price NUMBER, ProductDesc VARCHAR(30));

Avem un foreign key catre product type, de asemenea avem un cod produs mandatory, unic, acesta va fi folosit de catre userii aplicatiei pentru a identifica produsele.

PriceHistory:

CREATE TABLE PriceHistory(PriceHistoryID NUMBER PRIMARY KEY, StartDate DATE NOT NULL, EndDate DATE, Price NUMBER, ProductID NUMBER REFERENCES Product(ProductID));

Aceasta tabela tine istoricul preturilor unui produs, deci avem un FK catre product, startdate este mandatory, iar enddate nu, deoarece produsul poate are inca valabil pretul prezent in istoric, aceasta data se va updata la momentul schimbarii pretului produsului.

Customer:

CREATE TABLE Customer(CustomerID NUMBER PRIMARY KEY, CustomerCode VARCHAR(15) NOT NULL UNIQUE, Cif VARCHAR(20) NOT NULL, CustomerName VARCHAR(30), CustomerSince DATE NOT NULL, Notes VARCHAR(50));

In cazul persoanelor fizice, CIF va fi reprezentat de catre CNP. Notes este un field pentru eventualele notite despre un anumit client (ex. Bun platnic, obisnuieste sa comande x, etc) pentru a ajuta angajatii sa retina informatii utile despre clienti.

CustomerSince se va asigna automat la crearea unui record.

FOrder:

CREATE TABLE FOrder(OrderID NUMBER PRIMARY KEY, OrderNumber VARCHAR(30) NOT NULL UNIQUE, DateTime TIMESTAMP NOT NULL, CustomerID REFERENCES Customer(CustomerID), TotalPrice NUMBER, OrderTypeID NUMBER REFERENCES OrderType(OrderTypeID));

Din moment ce „Order” este un cuvant rezervat, vom denumi tabela Forder.

TotalPrice este un field calculat, practic fiind suma tuturor orderline-urilor ale unui order.

DateTime este data si ora exacta la care s-a realizat comanda.

OrderNumber este un field de tip caracter, similar cu ProductCode, se va folosi pentru a identifica orderele.

Doua foreign keys, una care ne indica pentru ce client cream un order, iar cealalta ce tip de order am creat.

OrderLine:

CREATE TABLE OrderLine(OrderLineID NUMBER PRIMARY KEY, OrderID NUMBER REFERENCES FOrder(OrderID), ProductID NUMBER REFERENCES Product(ProductID), Quantity NUMBER NOT NULL, TotalPrice NUMBER NOT NULL, CreatedDT TIMESTAMP NOT NULL);

Acest tabel reprezinta o linie dintr-un order, are ca foreign key un order si un produs. Este necesara completarea cantitatii comandate, pretul total, momentul crearii acestei linii.

Urmeaza crearea tabelelor de mapare:

CustomerAddress:

CREATE TABLE CustomerAddress(CustomerAddressID NUMBER PRIMARY KEY, AddressID NUMBER REFERENCES Address(AddressID), CustomerID NUMBER REFERENCES Customer(CustomerID), AddressTypeID NUMBER REFERENCES AddressType(AddressTypeID));

Am ales sa folosim si foreign key pentru AddressType, deoarece asta va imbunatati performantii cautarii adreselor ulterior.

Similar vom realiza si:

EmployeeAddress:

```
CREATE TABLE EmployeeAddress(EmployeeAddressID NUMBER PRIMARY KEY, AddressID
NUMBER REFERENCES Address(AddressID), EmployeeID NUMBER REFERENCES
Employee(EmployeeID), AddressTypeID NUMBER REFERENCES AddressType(AddressTypeID));
```

CustomerContact:

```
CREATE TABLE CustomerContact(CustomerContactID NUMBER PRIMARY KEY, ContactID
NUMBER REFERENCES Contact(ContactID), CustomerID NUMBER REFERENCES
Customer(CustomerID), ContactTypeID NUMBER REFERENCES ContactType(ContactTypeID));
```

EmployeeContact:

```
CREATE TABLE EmployeeContact(EmployeeContactID NUMBER PRIMARY KEY, ContactID
NUMBER REFERENCES Contact(ContactID), EmployeeID NUMBER REFERENCES
Employee(EmployeeID), ContactTypeID NUMBER REFERENCES ContactType(ContactTypeID));
```

<u>Popularea tabelelor</u>

County:

(id, cod, nume)

```
INSERT INTO County VALUES(1, 'AB', 'Alba');

INSERT INTO County VALUES(1, 'AB', 'Alba');

INSERT INTO County VALUES(2, 'CJ', 'Cluj');

INSERT INTO County VALUES(3, 'BN', 'Bistrita-Nasaud');

INSERT INTO County VALUES(4, 'HD', 'Hunedoara');

INSERT INTO County VALUES(5, 'BV', 'Brasov');

INSERT INTO County VALUES(6, 'MM', 'Maramures');
```

AnimalType:

(id, desc, cod)

```
INSERT INTO AnimalType VALUES (1, 'Porc calitate 1', 'PORC');
INSERT INTO AnimalType VALUES (2, 'Pui calitate 1', 'PUI');
INSERT INTO AnimalType VALUES (3, '', 'RATA');
INSERT INTO AnimalType VALUES (4, '', 'IEPURE');
INSERT INTO AnimalType VALUES (5, 'Curcan calitate 2', 'CURCAN');
INSERT INTO AnimalType VALUES (6, 'Peste', 'PESTE');
```

ContactType:

(id, desc, cod)

```
INSERT INTO ContactType VALUES(1, '', 'TELEFON');
INSERT INTO ContactType VALUES(2, '', 'FACEBOOK');
INSERT INTO ContactType VALUES(3, '', 'LINKEDIN');
INSERT INTO ContactType VALUES(4, '', 'EMAIL');
INSERT INTO ContactType VALUES(5, '', 'MOBIL');
INSERT INTO ContactType VALUES(6, '', 'TWITTER');
```

AddressType:

(id, desc, cod)

```
INSERT INTO AddressType VALUES(1, 'Main mailing address', 'MAILING1');
INSERT INTO AddressType VALUES(2, 'Secondary mailing address', 'MAILING2');
INSERT INTO AddressType VALUES(3, 'Third mailing address', 'MAILING3');
INSERT INTO AddressType VALUES(4, 'Bills only address', 'BILL');
INSERT INTO AddressType VALUES(5, 'Main delivery address', 'DELIVERY1');
INSERT INTO AddressType VALUES(6, 'Secondary delivery address', 'DELIVERY2');
```

ProductType:

(ProductTypeID, ProductTypeDesc,ProductTypeCode)

```
INSERT INTO ProductType VALUES(1, '', 'ANIMAL');
```

INSERT INTO ProductType VALUES(2, '', 'LACTAT');

INSERT INTO ProductType VALUES(3, '', 'BLANA');

INSERT INTO ProductType VALUES(4, '', 'REST');

INSERT INTO ProductType VALUES(5, '', 'TEST_PRODUCT');

INSERT INTO ProductType VALUES(6, '', 'TEST_PRODUCT2');

OrderType:

(id, desc, cod)

INSERT INTO OrderType VALUES(1, '', 'ORDER');

INSERT INTO OrderType VALUES(2, '', 'CERERE');

INSERT INTO OrderType VALUES(3, '', 'DEBARASARE');

INSERT INTO OrderType VALUES(4, '', 'PROMOTIE_ORDER');

INSERT INTO OrderType VALUES(5, '', 'TEST_ORDER');

INSERT INTO OrderType VALUES(6, '', 'TEST_ORDER2');

Employee:

(id, cod, prenume, nume, data nasterii, data angajarii, id manager)

INSERT INTO Employee VALUES(1, 'MRC123', 'Marcel', 'Aluminiu', '12/Sep/1967', '01/Jan/2021', 0);

INSERT INTO Employee VALUES(2, 'ABL222', 'Abel', 'Bunul', '15/Jan/1991', '21/Jan/2021', 1);

INSERT INTO Employee VALUES(3, 'ANA111', 'Ana', 'Bunea', '15/Jan/1991', '21/Jan/2021', 1);

INSERT INTO Employee VALUES(4, 'MAR111', 'Maria', 'Bunea', '15/Jan/1991', '21/Jan/2021', 1);

INSERT INTO Employee VALUES(5, 'MIH333', 'Mihai', 'Nedumeritul', '15/OCt/1989', '21/Jan/2021', 1);

INSERT INTO Employee VALUES(6, 'PET333', 'Petre', 'Namidee', '22/Dec/1978', '21/Feb/2021', 1);

In acest exemplu Angajatii il au ca manager pe Marcel Aluminiu(employee id 1).

Animal:

(id, cod, nume, stare sanatate, cod tip animal)

```
INSERT INTO Animal VALUES(1, 'PORC1', 'Porc aleator', 'Fara probleme', 1);
INSERT INTO Animal VALUES(2, 'PORC2', 'Porc aleator', '', 1);
INSERT INTO Animal VALUES(3, 'IEP1', 'Iepure aleator', '', 4);
INSERT INTO Animal VALUES(4, 'IEP2', 'Iepure aleator', '', 4);
INSERT INTO Animal VALUES(5, 'CURC1', 'Curcan generic', '', 5);
INSERT INTO Animal VALUES(6, 'CURC2', 'Curcan generic', '', 5);
```

Contact:

(id, info, contact type id)

```
INSERT INTO Contact VALUES(1, '+40755123456', 5);
INSERT INTO Contact VALUES(2, '+40755133456', 5);
INSERT INTO Contact VALUES(3, 'www.facebook.com/marcel123', 2);
INSERT INTO Contact VALUES(4, 'www.facebook.com/anabanana', 2);
INSERT INTO Contact VALUES(5, 'maria.mar@gmail.com', 4);
INSERT INTO Contact VALUES(6, 'abel.bunul@gmail.com', 4);
```

Address:

(id, addressline1, addressline2, addressline3, addresstype id, cod postal, id judet)

```
INSERT INTO Address VALUES (1, 'Str pietrelor nr 1', 'bl 4 sc 1 et 4 ap 29', '', 1, '422220', 2);
INSERT INTO Address VALUES (2, 'Str pietrelor nr 2', 'bl 5 sc 1 et 4 ap 29', '', 1, '411567', 1);
INSERT INTO Address VALUES (3, 'Str pietrelor nr 3', 'bl 5 sc 1 et 4 ap 29', '', 1, '411567', 6);
INSERT INTO Address VALUES (4, 'Str pietrelor nr 4', 'bl 1 sc 1 et 1 ap 1', '', 1, '411567', 5);
INSERT INTO Address VALUES (5, 'Str pietrelor nr 5', '', 'test add3', 1, '344999', 3);
INSERT INTO Address VALUES (6, 'Str pietrelor nr 6', '', 'test add3', 1, '344999', 3);
```

Contract:

(id, cod, salariu, id employee, start date, end date, titlu pozitie)

```
INSERT INTO Contract VALUES (1, '9000000', 5000, 1, '01/Jan/2021', '', 'CEO');
INSERT INTO Contract VALUES (2, '9000001', 2000, 2, '21/Jan/2021', '', 'MUNCITOR');
INSERT INTO Contract VALUES (3, '9000002', 2000, 3, '21/Jan/2021', '', 'MUNCITOR');
INSERT INTO Contract VALUES (4, '9000003', 2000, 4, '21/Jan/2021', '', 'MUNCITOR');
INSERT INTO Contract VALUES (5, '9000004', 2000, 5, '21/Jan/2021', '', 'MUNCITOR');
INSERT INTO Contract VALUES (6, '9000005', 1750, 5, '21/FEB/2021', '21/MAY/2021',
'MUNCITOR_PROBA');
```

Product:

(id, cod, nume, producttypeid, pret, desc)

```
INSERT INTO Product VALUES (1, 'X0000001','Porc', 1, 200, '');
INSERT INTO Product VALUES (2, 'L0000001','Iaurt 1.5', 2, 8, '');
INSERT INTO Product VALUES (3, 'B0000001','Blana urs maro', 3, 999, '');
INSERT INTO Product VALUES (4, 'T10000001','Test', 3, 0, '');
INSERT INTO Product VALUES (5, 'T10000002','Test', 3, 0, '');
INSERT INTO Product VALUES (6, 'T10000003','Test', 5, 11111, '');
```

PriceHistory:

(id, startdate, enddate, pret, productid)

```
INSERT INTO PriceHistory VALUES (1, '05/Jan/2021','27/Feb/2021', 598, 1);
INSERT INTO PriceHistory VALUES (2, '27/Feb/2021','27/Jun/2021', 667, 1);
INSERT INTO PriceHistory VALUES (3, '27/Jun/2021','', 200, 1);
INSERT INTO PriceHistory VALUES (4, '27/Jun/2021','', 8, 2);
INSERT INTO PriceHistory VALUES (5, '27/Jun/2021','', 999, 3);
INSERT INTO PriceHistory VALUES (6, '27/Jan/2021','', 1111, 6);
```

Customer:

(id, cod, cif, nume, customersince, notes)

```
INSERT INTO customer VALUES (1, 'C00001', '245674347', 'Tester SRL', '27/Feb/2021','');
INSERT INTO customer VALUES (2, 'C00002', '245674323247', 'Tester SRL 2', '22/Feb/2021','');
INSERT INTO customer VALUES (3, 'C00003', '243247', 'Tester SRL 3', '22/Mar/2021','');
INSERT INTO customer VALUES (4, 'C00004', '24111113247', 'Tester SRL 4', '22/Apr/2021','');
INSERT INTO customer VALUES (5, 'C00005', '2453247', 'Tester SRL 5', '21/Jan/2021','');
INSERT INTO customer VALUES (6, 'C00006', '2453247', 'Old SRL', '01/Jan/2021','oldest');
```

FOrder:

(id, cod, datetime, customerid, price, ordertypeid)

```
INSERT INTO FOrder VALUES (1, 'O000001', '29/Jun/2021', 1, 0, 1);
INSERT INTO FOrder VALUES (2, 'O000002', '29/Jun/2021', 2, 0, 2);
INSERT INTO FOrder VALUES (3, 'O000003', '29/May/2021', 3, 0, 3);
INSERT INTO FOrder VALUES (4, 'O000004', '29/May/2021', 4, 0, 4);
INSERT INTO FOrder VALUES (5, 'O000005', '29/May/2021', 5, 0, 5);
INSERT INTO FOrder VALUES (6, 'O000006', '29/May/2021', 1, 0, 1);
```

OrderLine:

(orderlineid, orderid, productid, quantity, totalprice, createddt)

```
INSERT INTO OrderLine VALUES (1, 1, 1, 15, 0, '29/Jun/2021');
INSERT INTO OrderLine VALUES (2, 2, 2, 15, 0, '29/Jun/2021');
INSERT INTO OrderLine VALUES (3, 3, 3, 15, 0, '29/Jun/2021');
INSERT INTO OrderLine VALUES (4, 4, 4, 15, 0, '29/Jun/2021');
INSERT INTO OrderLine VALUES (5, 1, 5, 15, 0, '29/Jun/2021');
INSERT INTO OrderLine VALUES (6, 2, 2, 15, 0, '29/Jun/2021');
```

CustomerAddress:

(customeraddressid, customerid, addressid, addresstypeid)

```
INSERT INTO CustomerAddress VALUES (1,1,1,1);

INSERT INTO CustomerAddress VALUES (2,2,2,1);

INSERT INTO CustomerAddress VALUES (3,3,3,1);

INSERT INTO CustomerAddress VALUES (4,4,4,1);

INSERT INTO CustomerAddress VALUES (5,5,5,1);

INSERT INTO CustomerAddress VALUES (6,6,6,6);
```

EmployeeAddress:

(employeeaddressid, employeeid, addressid, addresstypeid):

```
INSERT INTO EmployeeAddress VALUES (6,6,6,6);

INSERT INTO EmployeeAddress VALUES (1,1,1,1);

INSERT INTO EmployeeAddress VALUES (2,2,2,1);

INSERT INTO EmployeeAddress VALUES (3,3,3,1);

INSERT INTO EmployeeAddress VALUES (4,4,4,1);

INSERT INTO EmployeeAddress VALUES (5,5,5,1);
```

CustomerContact:

(customercontactid, customerid, contactid, contacttypeid):

```
INSERT INTO CustomerContact VALUES (5,5,5,1);

INSERT INTO CustomerContact VALUES (6,6,6,1);

INSERT INTO CustomerContact VALUES (1,1,1,1);

INSERT INTO CustomerContact VALUES (2,2,2,1);

INSERT INTO CustomerContact VALUES (3,3,3,4);

INSERT INTO CustomerContact VALUES (4,4,4,1);
```

EmployeeContact:

(employeecontactid, employeeid, contactid, contactypeid)

INSERT INTO EmployeeContact VALUES (1,1,3,1);

INSERT INTO EmployeeContact VALUES (2,3,4,1);

INSERT INTO EmployeeContact VALUES (3,4,5,1);

INSERT INTO EmployeeContact VALUES (4,2,6,1);

INSERT INTO EmployeeContact VALUES (5,5,1,1);

INSERT INTO EmployeeContact VALUES (6,6,2,1);

Modificari structura

Vom renunta la constraint-ul de unicitate all field-ului AddressLine1 din tabela Address:

ALTER TABLE Address DROP CONSTRAINT SYS_C006429317;

Vom creste numarul de caractere maxim admise pentru field-ul „Name” din tabela „Animal”

ALTER TABLE Animal

MODIFY Name VARCHAR(50);

Vom adauga un constraint prin care vom verifica daca field-ul „Quantity” al tabelei „OrderLine” are o valoare pozitiva, mai mare decat 0

ALTER TABLE OrderLine

ADD CONSTRAINT chk_quantity CHECK (Quantity > 0);

Actualizari de continut

Vom face un update al ContactTypeId in tabelele de mapare:

```
UPDATE CustomerContact

  SET (ContactTypeID) = (SELECT Contact.ContactTypeID FROM Contact WHERE
Contact.ContactID = CustomerContact.ContactID)
```

Similar vom face si in cazul EmployeeContact:

```
UPDATE EmployeeContact

  SET (ContactTypeID) = (SELECT Contact.ContactTypeID FROM Contact WHERE
Contact.ContactID = EmployeeContact.ContactID)
```

EmployeeAddress:

```
UPDATE EmployeeAddress

  SET (AddressTypeID) = (SELECT Address.AddressTypeID FROM Address WHERE
Address.AddressID = EmployeeAddress.AddressID)
```

CustomerAddress:

```
UPDATE CustomerAddress

  SET (AddressTypeID) = (SELECT Address.AddressTypeID FROM Address WHERE
Address.AddressID = CustomerAddress.AddressID)
```

Vom face un update pe TotalPrice pentru tabelele FOrder si OrderLine:

```
UPDATE OrderLine

  SET (TotalPrice) = (SELECT Product.Price FROM Product WHERE Product.ProductID =
OrderLine.ProductID) * OrderLine.Quantity
```

Aceasta instructiune va face un calcul inmultind pretul produsului de pe un OrderLine cu cantitatea descrisa in OrderLine.Quantity.

UPDATE FOrder

SET (TotalPrice) = (SELECT SUM(OrderLine.TotalPrice) FROM OrderLine WHERE OrderLine.OrderID = FOrder.OrderID)

Functia SUM folosita mai sus va insuma pretul de pe toate orderline-urile care apartin unui Forder

View-uri

Vom crea un view „Employee_Details" care ne va afisa informatiile de baza ale angajatilor:

Numele, prenumele, adresa, tipul adresei, contactul, tipul contactului

CREATE VIEW Employee_Details AS

SELECT Employee.FirstName, Employee.LastName, Employee.BirthDate, Contact.ContactInfo, ContactType.ContactTypeCode, Address.AddressLine1, AddressType.AddressTypeCode

FROM Employee, Contact, ContactType, Address, AddressType, EmployeeContact, EmployeeAddress

WHERE

Employee.EmployeeID = EmployeeContact.EmployeeID AND

Employee.EmployeeID = EmployeeAddress.EmployeeID AND

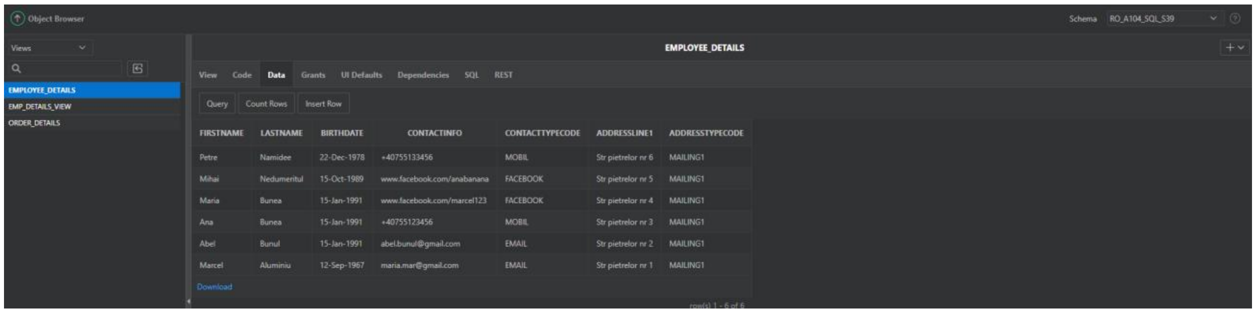
EmployeeAddress.AddressID = Address.AddressID AND

EmployeeContact.ContactID = Contact.ContactID AND

Address.AddressTypeID = AddressType.AddressTypeID AND

Contact.ContactTypeID = ContactType.ContactTypeID

Rezultat:



FIRSTNAME	LASTNAME	BIRTHDATE	CONTACTINFO	CONTACTTYPECODE	ADDRESSLINE1	ADDRESSTYPECODE
Petre	Namidee	22-Dec-1978	+40755133456	MOBIL	Str pietrelor nr 6	MAILING1
Mihai	Nedumeritul	15-Oct-1989	www.facebook.com/anabanana	FACEBOOK	Str pietrelor nr 5	MAILING1
Maria	Bunea	15-Jan-1991	www.facebook.com/marcel123	FACEBOOK	Str pietrelor nr 4	MAILING1
Ana	Bunea	15-Jan-1991	+40755123456	MOBIL	Str pietrelor nr 3	MAILING1
Abel	Bunul	15-Jan-1991	abel.bunul@gmail.com	EMAIL	Str pietrelor nr 2	MAILING1
Marcel	Aluminiu	12-Sep-1967	maria.mar@gmail.com	EMAIL	Str pietrelor nr 1	MAILING1

Vom crea un view „Order_Details” care ne va afisa informatii despre Ordere:

ProductCode, OrderLine.Quantity, Forder.TotalPrice, CustomerCode, CustomerName, customerSince, ContactInfo, ContactType

CREATE VIEW Order_Details AS

SELECT Product.ProductCode, FOrder.TotalPrice, Customer.CustomerCode, Customer.CustomerName, Customer.CustomerSince, Contact.ContactInfo, ContactType.ContactTypeCode

FROM FOrder, Customer, Contact, ContactType, CustomerContact, Product, OrderLine

WHERE

FOrder.CustomerID = Customer.CustomerID AND

CustomerContact.CustomerID = Customer.CustomerID AND

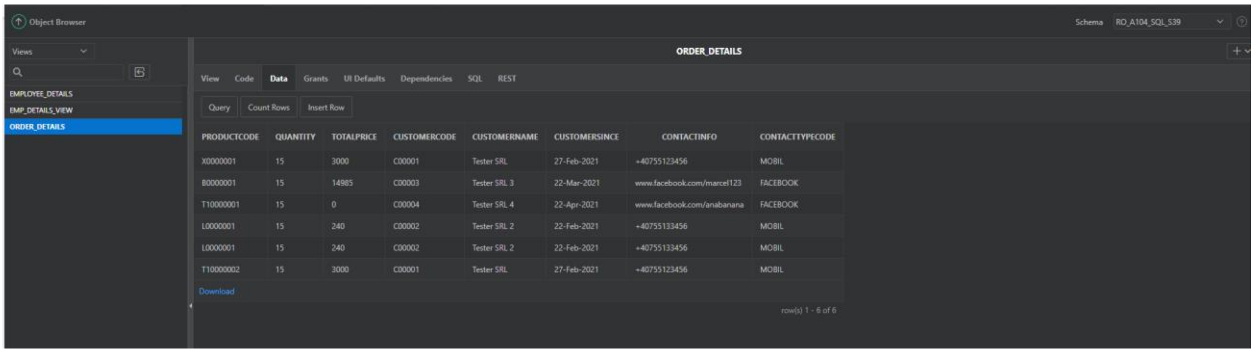
Contact.ContactID = CustomerContact.ContactID AND

ContactType.ContactTypeID = CustomerContact.ContactTypeID AND

OrderLine.OrderID = FOrder.OrderID AND

Product.ProductID = OrderLine.ProductID

Rezultat:



ORDER_DETAILS

PRODUCTCODE	QUANTITY	TOTALPRICE	CUSTOMERCODE	CUSTOMERNAME	CUSTOMERSINCE	CONTACTINFO	CONTACTTYPECODE
X0000001	15	3000	C00001	Tester SRL	27-Feb-2021	+40755123456	MOBIL
B0000001	15	14985	C00003	Tester SRL 3	22-Mar-2021	www.facebook.com/marcel123	FACEBOOK
T10000001	15	0	C00004	Tester SRL 4	22-Apr-2021	www.facebook.com/anabanana	FACEBOOK
L0000001	15	240	C00002	Tester SRL 2	22-Feb-2021	+40755133456	MOBIL
L0000001	15	240	C00002	Tester SRL 2	22-Feb-2021	+40755133456	MOBIL
T10000002	15	3000	C00001	Tester SRL	27-Feb-2021	+40755123456	MOBIL

row(s) 1 - 6 of 6

Interogari

1.

SELECT Product.ProductCode, FOrder.TotalPrice, Customer.CustomerCode, Customer.CustomerName, Customer.CustomerSince, Contact.ContactInfo, ContactType.ContactTypeCode

FROM FOrder, Customer, Contact, ContactType, CustomerContact, Product, OrderLine

WHERE

FOrder.CustomerID = Customer.CustomerID AND

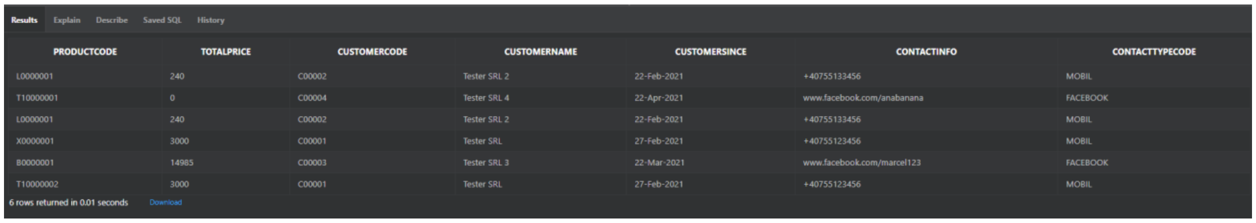
CustomerContact.CustomerID = Customer.CustomerID AND

Contact.ContactID = CustomerContact.ContactID AND

ContactType.ContactTypeID = CustomerContact.ContactTypeID AND

OrderLine.OrderID = FOrder.OrderID AND

Product.ProductID = OrderLine.ProductID



PRODUCTCODE	TOTALPRICE	CUSTOMERCODE	CUSTOMERNAME	CUSTOMERSINCE	CONTACTINFO	CONTACTTYPECODE
L0000001	240	C00002	Tester SRL 2	22-Feb-2021	+40755133456	MOBIL
T10000001	0	C00004	Tester SRL 4	22-Apr-2021	www.facebook.com/anabanana	FACEBOOK
L0000001	240	C00002	Tester SRL 2	22-Feb-2021	+40755133456	MOBIL
X0000001	3000	C00001	Tester SRL	27-Feb-2021	+40755123456	MOBIL
B0000001	14985	C00003	Tester SRL 3	22-Mar-2021	www.facebook.com/marcel123	FACEBOOK
T10000002	3000	C00001	Tester SRL	27-Feb-2021	+40755123456	MOBIL

6 rows returned in 0.01 seconds

Interogarea de mai sus ne va prezenta un set de informatii generale despre orderele pe care le avem.

2.

SELECT Employee.FirstName, Employee.LastName, Employee.BirthDate, Contact.ContactInfo, ContactType.ContactTypeCode, Address.AddressLine1, AddressType.AddressTypeCode

FROM Employee, Contact, ContactType, Address, AddressType, EmployeeContact, EmployeeAddress

WHERE

Employee.EmployeeID = EmployeeContact.EmployeeID AND

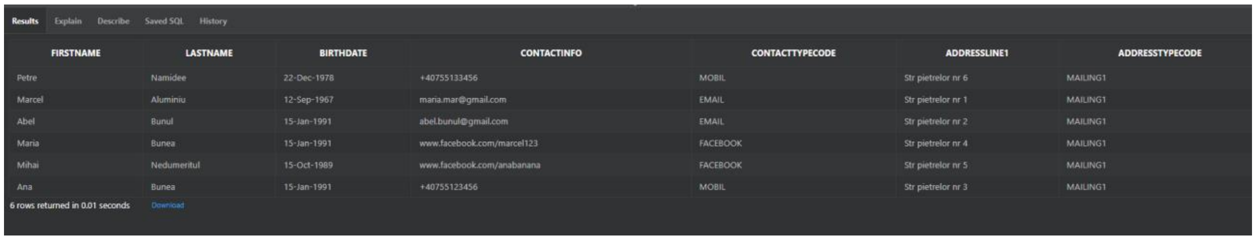
Employee.EmployeeID = EmployeeAddress.EmployeeID AND

EmployeeAddress.AddressID = Address.AddressID AND

EmployeeContact.ContactID = Contact.ContactID AND

Address.AddressTypeID = AddressType.AddressTypeID AND

Contact.ContactTypeID = ContactType.ContactTypeID



FIRSTNAME	LASTNAME	BIRTHDATE	CONTACTINFO	CONTACTTYPECODE	ADDRESSLINE1	ADDRESSTYPECODE
Petre	Namidee	22-Dec-1978	+40755133456	MOBIL	Str pietrelor nr 6	MAILING1
Marcel	Aluminiu	12-Sep-1967	maria.mar@gmail.com	EMAIL	Str pietrelor nr 1	MAILING1
Abel	Bunul	15-Jan-1991	abel.bunul@gmail.com	EMAIL	Str pietrelor nr 2	MAILING1
Maria	Bunea	15-Jan-1991	www.facebook.com/marcel123	FACEBOOK	Str pietrelor nr 4	MAILING1
Mihai	Nedumeritul	15-Oct-1989	www.facebook.com/anabanana	FACEBOOK	Str pietrelor nr 5	MAILING1
Ana	Bunea	15-Jan-1991	+40755123456	MOBIL	Str pietrelor nr 3	MAILING1

6 rows returned in 0.01 seconds

Interogarea de mai sus ne va prezenta un set de informatii generale despre angajatii pe care ii avem.

3.

SELECT Customer.CustomerCode, Customer.CustomerName, FOrder.TotalPrice, Product.ProductName

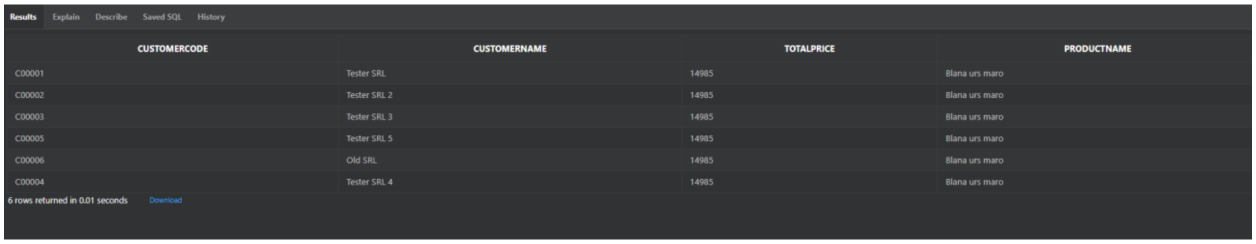
FROM Customer, FOrder, Product, OrderLine

WHERE FOrder.TotalPrice > 10000 AND

OrderLine.OrderID = FOrder.OrderID AND

Product.ProductID = OrderLine.ProductID

Interogarea de mai sus ne va prezenta numele, codul si ce au comandat clientii care au avut comenzi mai mari de 10000 de unitati monetare



CUSTOMERCODE	CUSTOMERNAME	TOTALPRICE	PRODUCTNAME
C00001	Tester SRL	14985	Blana urs maro
C00002	Tester SRL 2	14985	Blana urs maro
C00003	Tester SRL 3	14985	Blana urs maro
C00005	Tester SRL 5	14985	Blana urs maro
C00006	Old SRL	14985	Blana urs maro
C00004	Tester SRL 4	14985	Blana urs maro

6 rows returned in 0.01 seconds

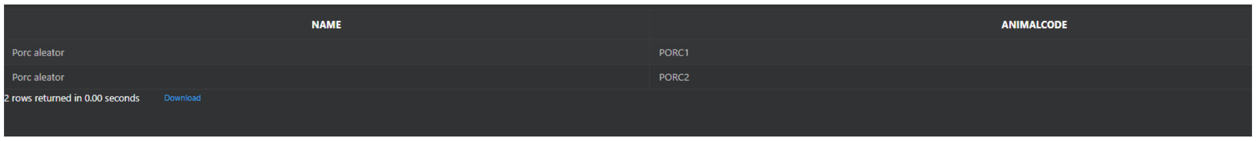
4.

SELECT Animal.Name, AnimalCode

FROM Animal

WHERE (SELECT AnimalType.AnimalTypeCode FROM AnimalType WHERE AnimalType.AnimalTypeID = Animal.AnimalTypeID) = 'PORC'

Aici vom avea ca rezultat doar animalele de tip „PORC”



NAME	ANIMALCODE
Porc aleator	PORC1
Porc aleator	PORC2

2 rows returned in 0.00 seconds Download

5.

SELECT Product.ProductName, AVG(OrderLine.TotalPrice)

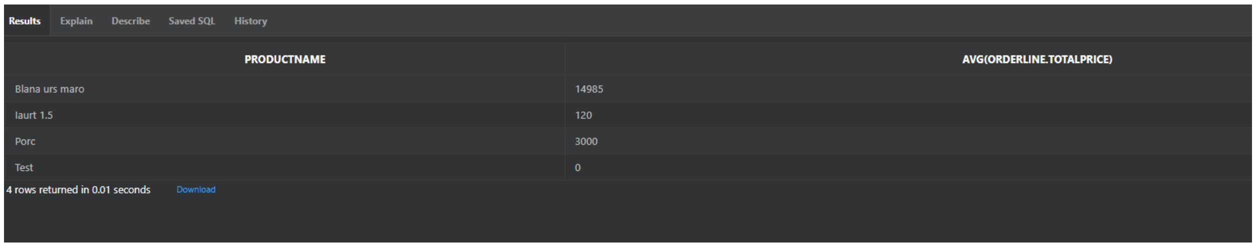
FROM Product

INNER JOIN OrderLine USING(ProductID)

GROUP BY Product.ProductName

ORDER BY Product.ProductName;

Va afisa media unitatilor monetare cheltuite per produs.



PRODUCTNAME	AVG(ORDERLINE.TOTALPRICE)
Blana urs maro	14985
Iaurt 1.5	120
Porc	3000
Test	0

4 rows returned in 0.01 seconds Download

6.

SELECT Customer.CustomerName, Address.AddressLine1, Address.AddressLine2, Address.AddressLine3, AddressType.AddressTypeCode

FROM Customer, Address, AddressType, CustomerAddress

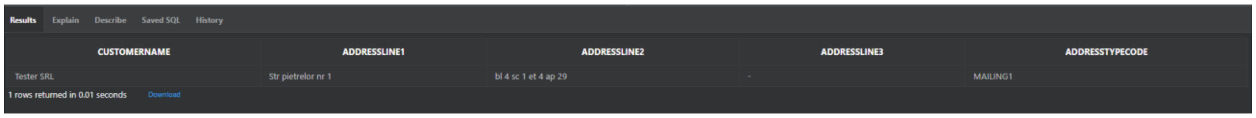
WHERE Customer.CustomerID = CustomerAddress.CustomerID AND

Address.AddressID = CustomerAddress.AddressID AND

AddressType.AddressTypeID = CustomerAddress.AddressTypeID AND

Address.CountyID = (SELECT County.CountyID FROM County WHERE County.CountyName LIKE 'C%')

Va afisa clientii si adresele lor in cazul in care acestia se afla intr-un judet al carui nume incepe cu litera ‚C’.



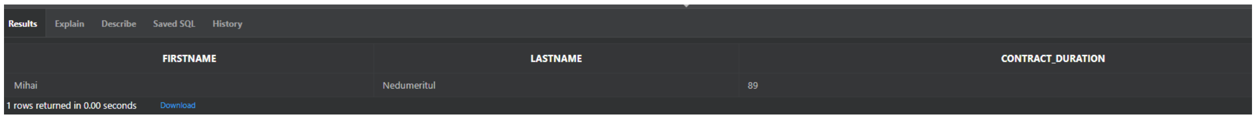
CUSTOMERNAME	ADDRESSLINE1	ADDRESSLINE2	ADDRESSLINE3	ADDRESSTYPECODE
Tester SRL	Str pietrelor nr 1	bl 4 sc 1 et 4 ap 29	-	MAILING1

1 rows returned in 0.01 seconds

7.

```
SELECT Employee.FirstName, Employee.LastName, (Contract.EndDate - Contract.StartDate) AS Contract_duration

FROM Employee

INNER JOIN Contract USING (EmployeeID)

WHERE (Contract.EndDate - Contract.StartDate) <= 90
```

Va afisa angajatii care au contractul pe o perioada mai mica sau egala cu 90 de zile precum si durata contractului.



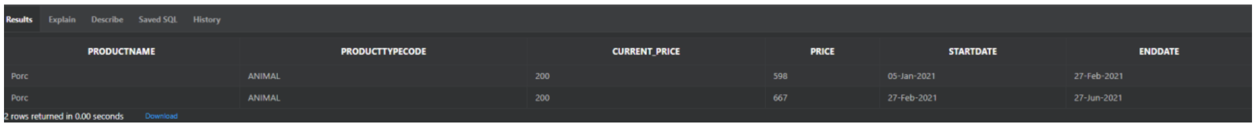
FIRSTNAME	LASTNAME	CONTRACT_DURATION
Mihai	Nedumeritul	89

1 rows returned in 0.00 seconds

8.

```
SELECT Product.ProductName, ProductType.ProductTypeCode, Product.Price AS Current_Price, PriceHistory.Price, PriceHistory.StartDate, PriceHistory.EndDate

FROM Product

INNER JOIN PriceHistory USING (ProductID)

INNER JOIN ProductType USING (ProductTypeID)

WHERE PriceHistory.EndDate IS NOT NULL

ORDER BY Product.ProductName;
```

Va afisa istoricul preturilor per produs, ordonate dupa numele produsului doar pentru produsele care si-au schimbat pretul, precum si pretul lor actual. Produsele care si-au pastrat pretul nu vor fi afisate



PRODUCTNAME	PRODUCTTYPECODE	CURRENT_PRICE	PRICE	STARTDATE	ENDDATE
Porc	ANIMAL	200	598	05-Jan-2021	27-Feb-2021
Porc	ANIMAL	200	667	27-Feb-2021	27-Jun-2021

2 rows returned in 0.00 seconds