**COURSEWORK SUBMISSION FORM**

|  |  |  |  |
| --- | --- | --- | --- |
| **STUDENT USE** | | **STAFF USE** | |
| Module Name | Database Systems Development | First Marker’s  (acts as signature) |  |
| Module Code | 5BUIS009C-n | Second Marker’s  (acts as signature) |  |
| Lecturer Name | Dmitriy Pochitaev | Agreed Mark |  |
| UoW Student IDs |  | **For Registrar’s office use only (hard copy submission)** | |
| WIUT Student IDs | 12219 |
| Deadline date | December 2, 2022 |
| Assignment Type | Individual |

**SUBMISSION INSTRUCTIONS**

**COURSEWORKS *must* be submitted in *both* HARD COPY (to the Registrar’s Office) *and* ELECTRONIC unless instructed otherwise.**

For hardcopy submission instructions refer to: [http://intranet.wiut.uz/Shared%20Documents/Forms/AllItems.aspx](http://intranet.wiut.uz/Shared Documents/Forms/AllItems.aspx) - Coursework hard copy submission instructions.doc

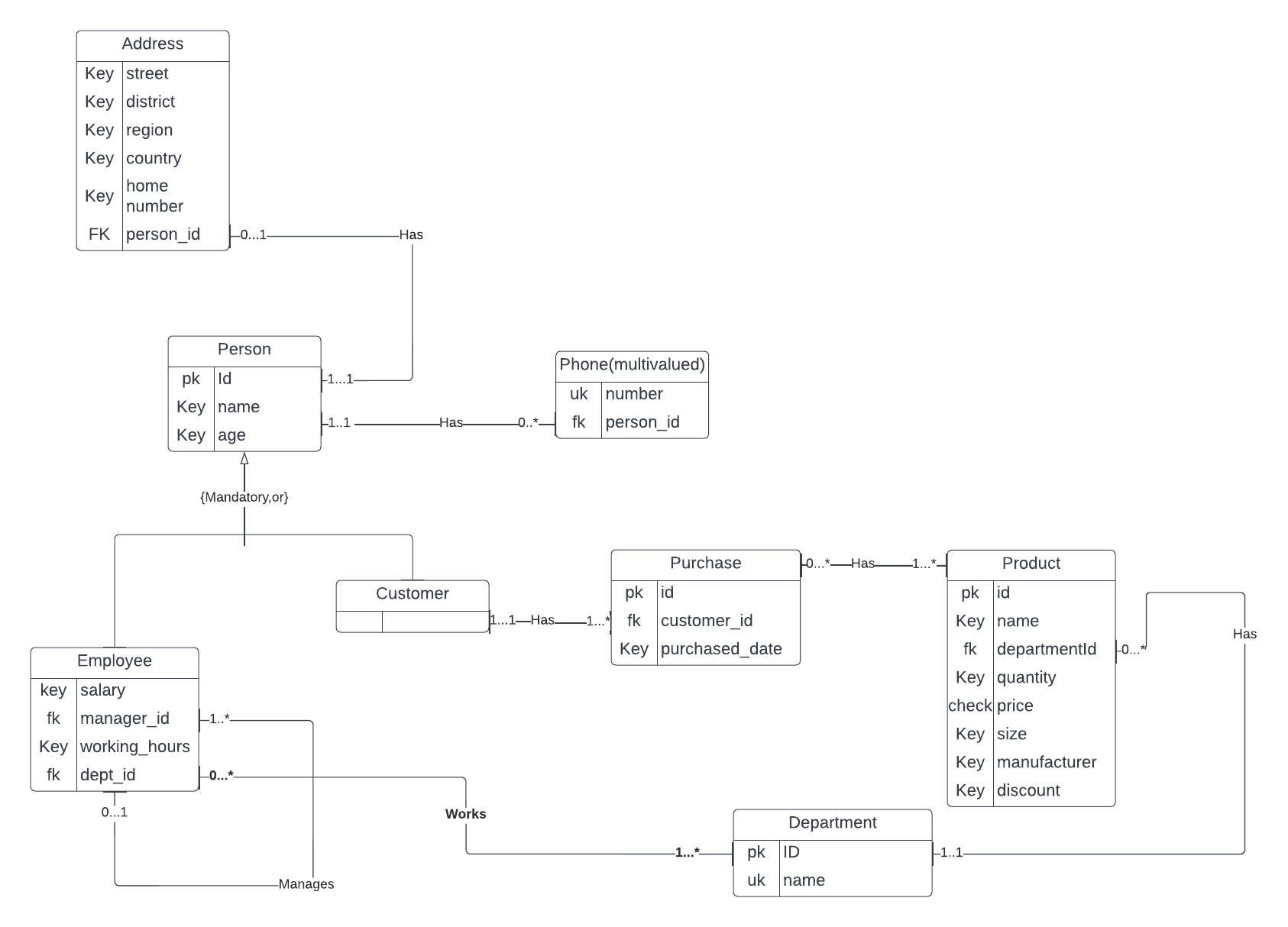
For online submission instructions refer to: [http://intranet.wiut.uz/Shared%20Documents/Forms/AllItems.aspx](http://intranet.wiut.uz/Shared Documents/Forms/AllItems.aspx) - Coursework online submission instructions.doc

|  |
| --- |
| **MARKERS FEEDBACK (Continued on the next page)** |
|  |

1. **Report**

Shop named “Terra Pro” decided to store information about everything or every type of person that is directly related to their shops. I personally do a lot of shopping from this shop ranging from shoes to every type of clothing from time to time. There are couple of franchises of this brand across the Tashkent city. Two of the most important types of people they care about are their employees and their customers. They want to store names, ages, phone numbers, salaries, managers of each employee, working hours, and the department that they are working in. As for customers, their names, ages, phone numbers are to be stored in the database. Both of them can have phone numbers ranging from 0 to many. Likewise with address, one person can share no address or one address. But one address can be only assigned to only one person, which can be either customer or employee of the shop. As for employees each employee can have either no manager or only one manager who is controlling them. If that employee is the manager himself, then he has no manager. But one manager can control as many as employees. Another entity shop owners want to know about is the department. They want to know only the name of the department. As for the relationship with the employees, in one department, many workers may work or even no employee may work in one department. But one employee must work in at least 1 department until as many departments. Another entity, the one of the most important one is product. Products are also associated with specific departments. One product can be in one and only one department in any case. But one department can have products from 0 number to many products. Another data they want to know about is the purchases that customers are making. Purchases are related with product as well as customers. In terms of relationship between customers and purchases, from the side of customer, one customer can have from 1 to many purchases. But one purchase can only done by one and only one customer. In terms of relationship between product and purchase. In one purchase customer can buy from one product to many products. But product can be 0 to many related with purchases.

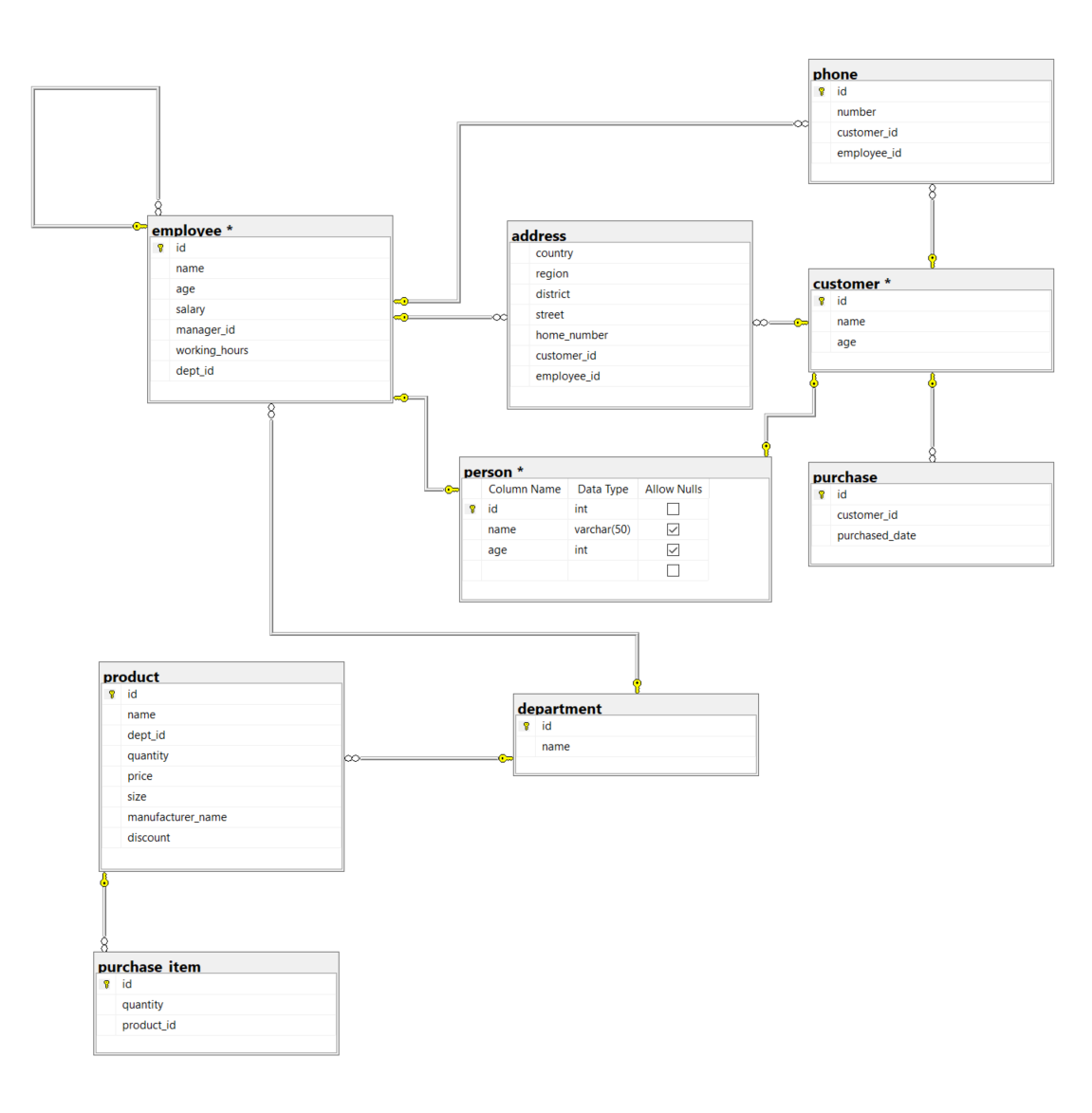
Enhanced ER diagram of Terra Pro shop.



Here it can be seen that Person class is the superclass for both Employee and the customer classes. Both of their common attributes has been stored in Person class.

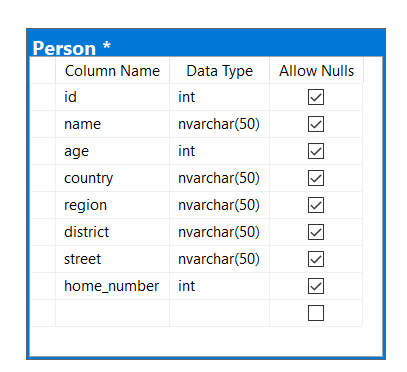
In terms of **multivalued attributes** the **phone number** can be considered as the mutivalued attribute since the one person can have more than one phone number by the requirement. As for **composite attribute,** the **address** can be seen as a composite attribute since the address table has been separated from person table.

As for relationships, Person can have one or no address, no phone number or as many phone numbers as he wants. Customer can have from 1 to many purchases. One purchase can have from 1 to many products. One department can have from 0 to many products. Employee can have either 0 or 1 manager. But manager can control as many sellers including 0.

Mapping of EER diagram to relational model using MS Sql Server Management Studio.

Here it can be seen that ***both customer and employee are the subclasses of the person table. Three attributes are inherited from parent person class to child classes. Another thing is that a lot of association have been used among tables to build relationship between them. Examples are between customer and phone, employee and phone, employee and address, customer and address, product and product item, customer and purchase. In terms of multivalued attributes, phone number has been used as a multivalued attribute. This was supposed to be a field of employee table and customer table but since these people c***an have more than one phone number, multivalued attribute had been used in this case scenario by creating a separate table for phone.

Regarding the normalization according to the 3NF. No normal keys should depend on the non-primary key fields. In my RM diagram all the tables have been normalized already. Because all the fields are directly depend on the primary key field, mostly id. I used to have the normalization problem while the address fields were situated inside of Person table like this.



But this would not be a correct way of doing this. Functional Dependency would be like this here:

{id->name, id-> name, id-> age, id-> country, country-> region, region-> district, district-> street, street->home\_number}. And this would violate the 3NF, because as stated above all the fields should depend only on the primary key(either super key or candidate key) in the table.

I have mostly tried to use the candidate key to follow the rule of third normal form.

One problem that I also tackled while building the EER is that person table used to have the phone number. But person can have more than 1 phone number. In this case there would be redundancy in the person table. That is why I have decided to make separate table of phone number table.

1. **Practical Part**

use master

GO

IF EXISTS (SELECT name FROM master.dbo.sysdatabases WHERE name = N'TerraPro')

BEGIN

ALTER DATABASE [TerraPro] SET OFFLINE WITH ROLLBACK IMMEDIATE;

ALTER DATABASE [TerraPro] SET ONLINE;

DROP TABLE IF EXISTS [TerraPro].[dbo].[phone];

DROP TABLE IF EXISTS [TerraPro].[dbo].[address];

DROP TABLE IF EXISTS [TerraPro].[dbo].[purchase\_item];

DROP TABLE IF EXISTS [TerraPro].[dbo].[purchase];

DROP TABLE IF EXISTS [TerraPro].[dbo].[product];

DROP TABLE IF EXISTS [TerraPro].[dbo].[customer];

DROP TABLE IF EXISTS [TerraPro].[dbo].[employee];

DROP TABLE IF EXISTS [TerraPro].[dbo].[department];

DROP DATABASE [TerraPro];

END

GO

create database [TerraPro] --Create new database with name TerraPro

GO

use TerraPro

GO

create table department (

id int identity(1,1) not null,

name varchar(20) not null,

constraint pk\_department\_id primary key(id),

constraint uk\_department\_name unique(name));

GO

insert into department(name) values('Man');

insert into department(name) values('Woman');

insert into department(name) values('Kid');

GO

create table employee(

id int identity (1,1) not null,

name varchar(30) not null,

age int not null,

salary money not null,

manager\_id int constraint fk\_employee\_manager references employee(id),

working\_hours int not null,

dept\_id int constraint fk\_employee\_dept references department(id),

constraint pk\_employee\_id primary key(id),

constraint ck\_employee\_age check(age>=18))

GO

insert into employee(name, age, salary, manager\_id, working\_hours, dept\_id) values ('Kolya', 19, 10000, null, 3, 1)

insert into employee(name, age, salary, manager\_id, working\_hours, dept\_id) values ('John', 20, 20000, 1, 4, 3)

insert into employee(name, age, salary, manager\_id, working\_hours, dept\_id) values ('Chandler', 24, 40000, 1, 8, 3)

go

create table customer(

id int identity(1,1) not null ,

name varchar(30) not null,

age int not null,

constraint pk\_customer\_id primary key(id))

go

insert into customer(name, age) values('Mike', 12)

insert into customer(name, age) values('Joe', 14)

insert into customer(name, age) values('Johanna', 66)

go

create table phone(

id int identity not null,

number varchar(9) not null,

customer\_id int constraint fk\_phone\_customer references customer(id),

employee\_id int constraint fk\_phone\_employee references employee(id),

constraint ck\_phone\_one\_is\_not\_null check (coalesce(customer\_id, employee\_id) is not null),

constraint ck\_phone\_only\_employee\_null check((employee\_id is null and customer\_id is not null) or (employee\_id is not null and customer\_id is null)),

constraint ck\_phone\_number check (number not like '%[^0-9]%'), --checks if only digit is written not the letters or other symbols

constraint ck\_phone\_length CHECK(DATALENGTH(number)=9), --checks for the exactly 9-digit number

constraint pk\_phone\_id primary key(id))

go

insert into phone(number, customer\_id, employee\_id) values ('222222221', 1, null)

insert into phone(number, customer\_id, employee\_id) values ('111111111', null, 1)

insert into phone(number, customer\_id, employee\_id) values ('111111111', null, 1)

-- below code does not run because both of the customer\_id and employee\_id cannot have the value at the same time.

-- insert into phone(number, customer\_id, employee\_id) values ('11111111', 1, 1)

go

create table address(

country varchar(35) not null ,

region varchar(35) not null,

district varchar(35) not null,

street varchar(35) not null,

home\_number int not null,

customer\_id int constraint fk\_address\_customer references customer(id),

employee\_id int constraint fk\_address\_employee references employee(id),

constraint ck\_address\_one\_is\_not\_null check (coalesce(customer\_id, employee\_id) is not null),

constraint uk\_address\_count\_reg\_dis\_str\_hn\_ci\_ei unique(country, region, district, street, home\_number, customer\_id, employee\_id)

)

go

insert into address values('Uzbekistan', 'Samarkand', 'Jom', 'Kazirabad', 14, 1, 1)

insert into address values('Uzbekistan', 'Toshkent', 'Mirzo Ulugbek', 'Sayram', 15, 1, 1)

insert into address values('Uzbekistan', 'Qashqadaryo', 'Qarshi', 'Guzor', 16, 1, 1)

go

create table product(

id int identity(1,1) not null, --id is set automatically

name varchar(30) not null,

dept\_id int constraint fk\_product\_dept references department(id) not null,

quantity int not null,

price money not null,

size int not null,

manufacturer\_name varchar(30),

discount int default 0,

constraint pk\_product\_id primary key(id))

go

insert into product(name, dept\_id, quantity, price, size, manufacturer\_name) values ('Dress', 1, 100, 1000, 32, 'Crocodile')

insert into product(name, dept\_id, quantity, price, size, manufacturer\_name) values ('Dunk', 2, 200, 2000, 34, 'Adidas')

insert into product(name, dept\_id, quantity, price, size, manufacturer\_name) values ('Jogger', 3, 300, 3000, 35, 'Nike')

go

create table purchase(

id int identity(1,1) not null, --id is set automatically

customer\_id int constraint fk\_purchase\_customer references customer(id) not null,

purchased\_date datetime default getDate(),

constraint pk\_purchase\_id primary key(id))

go

insert into purchase(customer\_id) values (1);

insert into purchase(customer\_id) values(3);

insert into purchase(customer\_id) values(2);

go

create table purchase\_item(

id int identity(1,1) not null, --id is set automatically

quantity int not null,

product\_id int constraint fk\_purchase\_item\_product references product(id) not null,

purchase\_id int constraint fk\_purchase\_item\_purchase references purchase(id) not null,

constraint pk\_purchase\_item\_id primary key(id))

go

insert into purchase\_item(quantity, product\_id, purchase\_id) values (10, 1, 1);

insert into purchase\_item(quantity, product\_id, purchase\_id) values (20, 2, 1);

insert into purchase\_item(quantity, product\_id, purchase\_id) values (30, 3, 1);

References

<https://lucid.app/lucidchart/5c6f4bdc-66b1-4379-9291-c3eaccb4b1d4/edit?invitationId=inv_71ec7e4c-602b-441b-8270-5cf881f488ca>