Phase 2: Data Modeling

Entity are an object with physical existence or with a conceptual existence

The entities in the diagram are PERSON,BRANCH,STOCK,EMPLOYEE.

Relation In any business processing one object may be associated with another object due to some event.

The Relationship in this diagram is WORKES WITH, DONATION, WORKES WITH, DELIVERS.

Attribute the items of information which characterize and describe these entities.

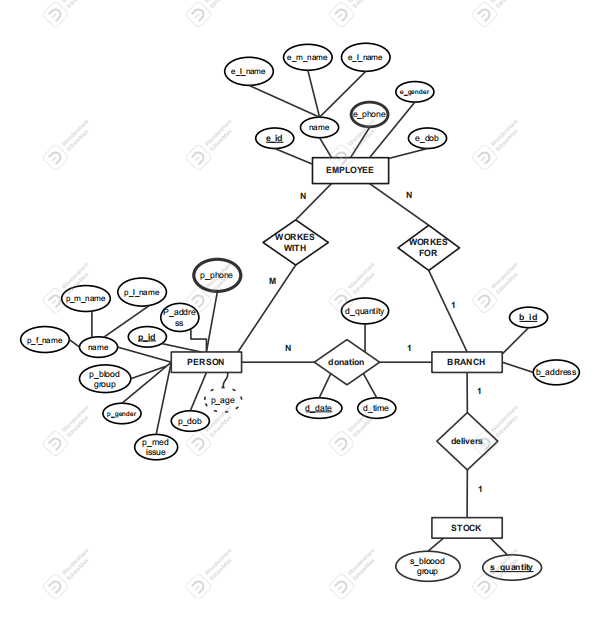
The Attributes in the entity PERSON is p\_id, p\_name ,p\_ gender,p\_ blood group, p\_date of birth,p\_ phone no , p\_address, p\_medical issue.

The Attributes in the entity EMPLOYEE is e\_id, name ,e\_ gender,e\_date of birth,e\_ phone no , e\_address,b\_id.

The Attributes in the entity BRANCH is b\_id, m\_id, b\_id.

The Attributes in the entity STOCK is s\_blood\_group, s\_quantity, b\_id.

**THE ER DIAGRAM**



Phase 3: Logical Design

The Relational schema is

Person(**p\_id**,name, p\_gender,p\_age,p\_dob,p\_address,p\_b\_group,p\_med\_issues**,b\_id** )

Person\_phone(**p\_id**, **p\_phone**)

Employee(**e\_id**,name,e\_gender,e\_phone,e\_dob,**b\_id**)

Branch(**b\_id**,b\_adress)

Stock(**s\_blood\_group**, s\_quantity,**b\_id**)

Table person

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field name | Data type | Field size | Constraint | Validation |
| p\_id | int | 4 bytes | PK | NOT NULL |
| p\_f\_name | varchar | 8 bytes |  | NOT NULL |
| p\_m\_name | varchar | 8 bytes |  | NOT NULL |
| p\_l\_name | varchar | 8 bytes |  | NOT NULL |
| p\_gender | char | 1 bytes |  | NOT NULL |
| p\_age | int | 4 bytes |  | NOT NULL |
| p\_phone | int | 4 bytes |  | NOT NULL |
| p\_dob | date | 7 bytes |  | NOT NULL |
| p\_address | varchar | 8 bytes |  | DEFAULT NULL |
| p\_b\_group | varchar | 8 bytes |  | NOT NULL |
| p\_med\_issues | varchar | 8 bytes |  | DEFAULT NULL |

Table employee

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field name | Data type | Field size | Constraint | Validation |
| e\_id | int | 4 bytes | PK | NOT NULL |
| e\_f\_name | varchar | 8 bytes |  | NOT NULL |
| e\_m\_name | varchar | 8 bytes |  | NOT NULL |
| e\_l\_name | varchar | 8 bytes |  | NOT NULL |
| e\_gender | char | 1 bytes |  | NOT NULL |
| e\_phone | int | 4 bytes |  | NOT NULL |
| e\_dob | date | 7 bytes |  | NOT NULL |
| b\_id | int | 4 bytes | FK |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field name | Data type | Field size | Constraint | Validation |
| p\_id | int | 4 bytes | PK,FK |  |
| d\_date | date | 7 bytes |  | NOT NULL |
| d\_time | time | 3 bytes |  | NOT NULL |
| d\_quantity | int | 4 bytes |  | NOT NULL |
| b\_id | int | 4 bytes | PK,FK |  |

Table donation

Table branch

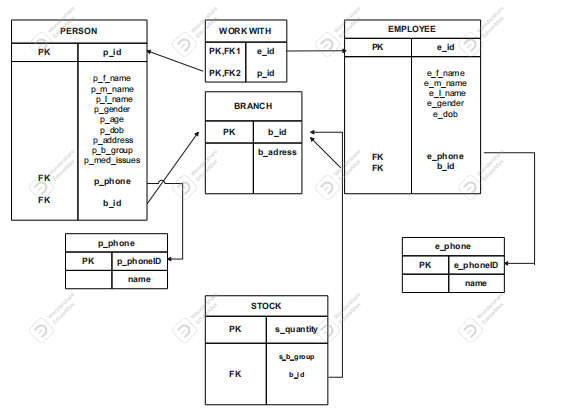
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field name | Data type | Field size | Constraint | Validation |
| b\_id | int | 4 bytes | PK | NOT NULL |
| b\_address | varchar | 8 bytes |  | DEFAULTNULL |

Table stock

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field name | Data type | Field size | Constraint | Validation |
| s\_blood\_group | varchar | 8 bytes | PK | NOT NULL |
| s\_quantity | int | 4 bytes |  | NOT NULL |

**The Relational Model**

**Diagram**



**The Code**

CREATE DATABASE blood\_doner;

USE blood\_doner;

CREATE TABLE employee (

e\_id int NOT NULL,

e\_f\_name varchar(25) NOT NULL,

e\_m\_name varchar(25) NOT NULL,

e\_l\_name varchar(25) NOT NULL,

e\_gender char(1) NOT NULL,

e\_phone char(10) NOT NULL,

e\_dob date NOT NULL,

b\_id int,

PRIMARY KEY(e\_id)

);

alter table employee

add foreign key (b\_id) references branch(b\_id) on delete set null;

CREATE TABLE branch (

b\_id int PRIMARY KEY NOT NULL,

b\_adress varchar(40) NOT NULL

);

CREATE TABLE person (

p\_id int NOT NULL,

p\_f\_name varchar(25) NOT NULL,

p\_m\_name varchar(25) NOT NULL,

p\_l\_name varchar(25) NOT NULL,

p\_gender char(1) NOT NULL,

p\_age int NOT NULL,

p\_phone char(10) NOT NULL,

p\_dob date NOT NULL,

p\_address varchar(100) DEFAULT NULL,

p\_b\_group varchar(3) NOT NULL,

p\_med\_issues varchar(100) DEFAULT NULL,

PRIMARY KEY(p\_id)

);

CREATE TABLE donation (

p\_id int ,

d\_date date NOT NULL,

d\_time time NOT NULL,

d\_quantity int NOT NULL,

b\_id int

);

ALTER TABLE donation

ADD PRIMARY KEY (p\_id,b\_id),

ADD FOREIGN KEY (p\_id) REFERENCES person (p\_id) on delete cascade,

ADD FOREIGN KEY (b\_id) REFERENCES branch (b\_id) on delete cascade;

CREATE TABLE stock (

s\_blood\_group varchar(3) PRIMARY KEY NOT NULL,

s\_quantity int NOT NULL,

b\_id int

);

alter table stock

add foreign key (b\_id) references branch(b\_id) on delete set null;

select \* from stock;

INSERT INTO stock(s\_blood\_group,s\_quantity) VALUES

('A+', 200),('A-', 100),('AB+', 100),('AB-', 0),('B+', 0),('B-', 0),('O+', 100),('O-', 100);

select \* from employee;

insert into employee value(100,'samuel','alebachew','kebede','M','0911111211','1981-5-3',1),

(101,'gebre','biru','alemu','M','0911111411','1990-2-13',3),

(102,'mahlet','alemu','kebede','F','091211111','1975-2-3',3),

(103,'habetamu','alemu','kebede','M','0913111111','1980-7-3',2),

(104,'tesfaye','alemu','kebede','M','0914111111','1986-12-3',1),

(105,'yordanos','alelegn','kebede','F','0915111111','1974-2-15',2),

(106,'mekedes','fekadu','alemu','F','0916111111','1984-12-13',3),

(107,'endris','alemu','kebede','M','0916111111','1980-4-6',2),

(108,'helen','zewedu','gemechu','F','0916111111','1985-2-3',1),

(109,'elias','gezaw','kebede','M','0917111111','1989-10-3',2);

select \* from branch;

insert into branch value(1,'kera branch');

insert into branch value(2,'gerji branch');

insert into branch value(3,'semit branch');

select \* from person;

insert into person value(1,'abebe','alemu','kebede','M',34,'0911111111','1980-2-3','kilinto','A+','nothing'),

(2,'tadios','geremew','akalu','M',24,'0912111111','1990-7-21','bole','O+','nothing'),

(3,'tinsia','getachew','kebede','M',19,'0911111113','1995-2-3','semit','A-','nothing'),

(4,'dagim','habetamu','kebede','M',28,'0911111141','1986-2-3','gerji','A+','nothing'),

(5,'ruth','tadesse','kebede','F',20,'0911115111','1994-2-3','tulu dimtu','B+','heart diesase'),

(6,'abdela','wasihun','kebede','M',40,'0911111161','1974-2-3','4 kilo','B-','Hematuria'),

(7,'tigist','','kebede','F',47,'0911171111','1968-2-3','mexico','O-','Hepatitis A'),

(8,'elsa','alemu','kebede','F',50,'0911181111','1964-2-3','megenagna','AB+','nothing'),

(9,'getahun','alemu','kebede','M',24,'0911119111','1990-5-30','ayat','AB-','HIV'),

(10,'rahel','alemu','kebede','F',54,'0911110111','1960-2-3','goro','O-','nothing');

select \*from donation;

insert into donation value(1,'2022-7-4',':15:3',100,2),

(2,'2021-4-6',':00:2',100,1),

(3,'2021-5-8',':30:9',100,3),

(4,'2022-1-5',':15:4',100,3),

(5,'2022-10-19',':30:6',0,2),

(6,'2021-3-15',':30:3',0,1),

(7,'2020-5-3',':30:2',0,3),

(8,'2020-1-5',':30:7',100,2),

(9,'2022-4-4',':30:1',0,1),

(10,'2021-1-1',':30:5',100,1);

**-- QUERIES**

select \* from person

where p\_med\_issues ="nothing";

select s\_blood\_group,s\_quantity from stock

where s\_quantity >0;

select count(p\_id)from person

where p\_med\_issues ="nothing";

select \* from stock

order by s\_quantity desc;

select sum(s\_quantity)

from stock;

select count(p\_gender),p\_gender

from person

where p\_med\_issues='nothing'

group by p\_gender;

-- **TRIGGER**

create table trig\_view(

message varchar(200)

);

delimiter $$

create trigger trig\_ins before insert

on person

for each row begin

insert into trig\_view value

("added new potential doner ");

end $$

delimiter ;

delimiter $$

create trigger trig\_ins2 before insert

on person

for each row begin

insert into trig\_view value(new.p\_id,new.p\_f\_name,new.p\_age);

end $$

delimiter ;

delimiter $$

create trigger trig\_upd before update

on person

for each row begin

insert into trig\_view value("updated potential doners info");

end $$

delimiter ;

delimiter $$

create trigger trig\_del before delete

on person

for each row begin

insert into trig\_view value("deleted potential doner");

end $$

delimiter ;

select \* from trig\_view;

**Data Security and Back up Plan**

Data security is important to protect the data in an organisation from unauthorised access, malicious attacks, or phishing activities. The term database security includes the methods taken to protect the security breach from the data level, user-level, and system level.

A database could be open to threats like DDoS attacks, unauthorised database access, SQL injection attacks, and more. Organisations must take measures to maintain the data security of the database, ensure fast recovery, and always have a backup available.

Ensuring the security of your database backups is equally important as the security for the database itself. Reliable backups are an important part of recovery. This blog outlines common database backup security considerations which should be implemented to maintain the data security and integrity of these files.

Database Backup Security

Like the database, backups should also be secured on three levels namely; data level, user-level, and system level. There should be more than one copy of a backup, this ensures that if the backup has been damaged or altered by unauthorised access, there would still be a clean copy for recovery purposes.

User-Level Database Backup Security

However well the data is encrypted, if it is easily available to anyone, encryption would be pointless. The place where backups are stored must have the same user management controls as your database access does.

It is also good to have multiple-factor authentication, not only for the database but for the database backups as well. A good combination of user login (username and strong password) combined with security codes from mobiles is a good way to verify the access as a means to data protection. This is a good method for any off-premise backup access (such as backups on the cloud).

System-Level Database Backup Security

This is a security measure at the hardware, network and communication lines level which can also pose a threat to the database backups. Like a database, the database backup files can also be accessed via these channels. To avoid this, there should be limited access and protection (such as a firewall) on the servers wherever the backups are stored. This minimizes the threat to backup servers or backup hosting via system-level threat

Conclusion

Having a good database backup plan and policy is important in any organisation. The plan or policy should include enough measures to secure the backup from various types of malicious attacks.

**Future Enhancements**

Exciting new enhancements to the database include adding password protection to the system. we will be give them individual passwords that will allow them access to certain parts of the database, thus allowing them to view the records of the person donated, and let them enter new donor of the students directly into the database