PET STORE MANAGEMENT SYSTEM

A Project Report
Submitted by

D. HUSSAIN (19331A1216)

in partial fulfillment for the Award of the Degree of
BACHELOR OF TECHNOLOGY

IN INFORMATION TECHNOLOGY

MVGR COLLEGE OF ENGINEERING

2019-2023

Description:

The pet stores industry comprises of stores that sell a range of pets, such as dogs, cats, fish and birds and also, they sell pet foods and pet supplies, such as collars, leashes, health and beauty aids, shampoos, medication, toys, pet containers, dog kennels and cat furniture et al. Some stores also offer pet services, such as grooming and training.

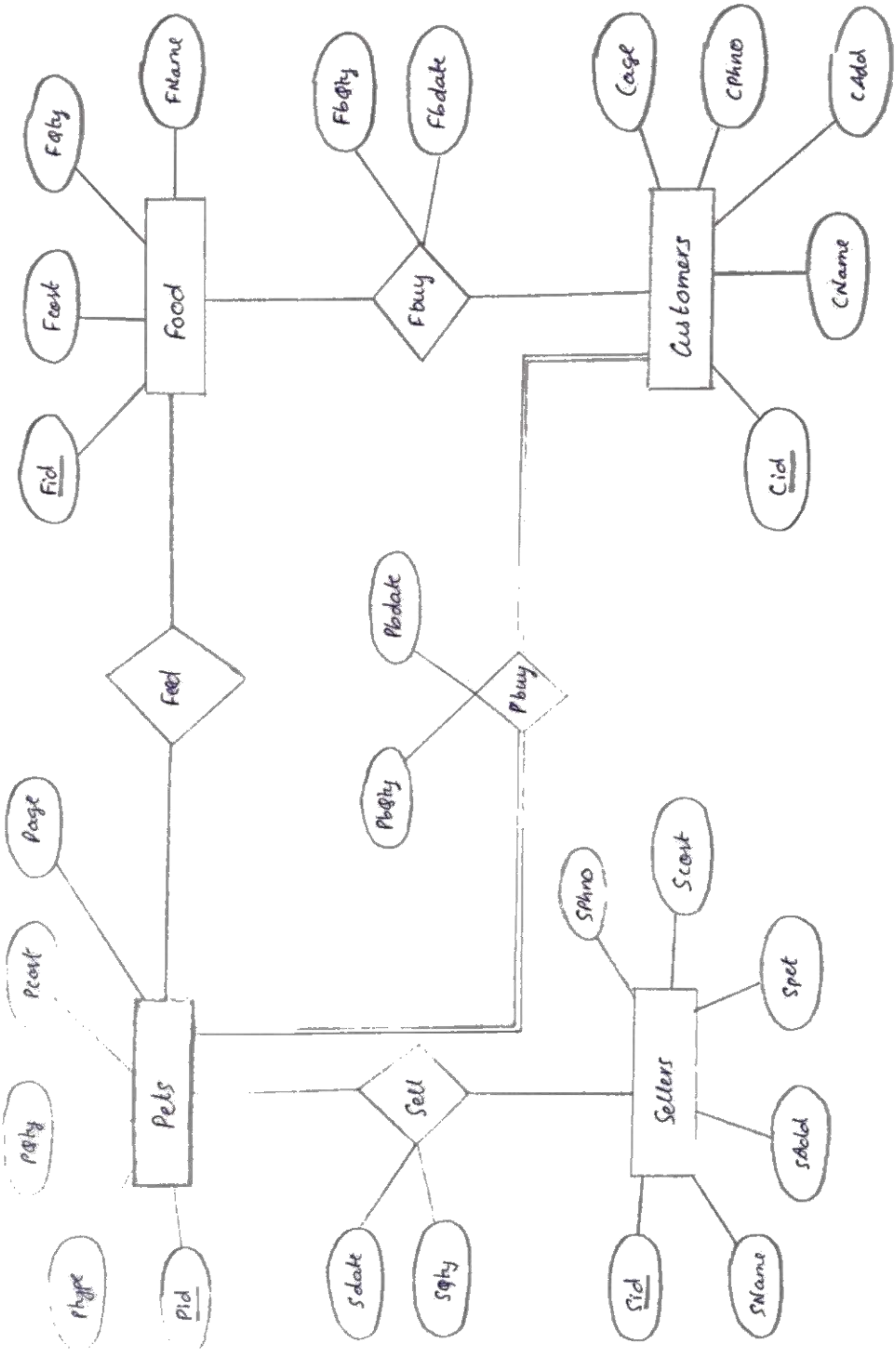
The Pet Stores industry has steadily remained relevant over the years as cats, dogs, fish and birds have remained common companions in many households. Pet parents or pet owner who treats their pet friends as family member; have grown increasingly hence the increase in demand for pet stores products and services.

So, this project is highly useful for the pet store industry to manage its data in easy-to-work and efficient manner.

Software Requirement Specification (SRS):

1. Pets have an ID, a type, quantity of pet, their cost and their age duration.
2. Foods given to them are based on pet type and has Food ID, food name, quantity, cost spent on food and food sold to customers.
3. Food is fed to pets and each type of pet has its own type of food. One or more type of food can be fed to each pet.
4. Customer has an ID, address, name, phone number, their age who can buy pets as well as food from the pet store.
5. Customer can buy pets and their food. It can either be single or customer can buy randomly, pets and food, food alone, anyway possible.
6. Customer can be termed as buyers too.
7. There are customers and sellers handled by this store which do not have direct connection between customers and sellers. Pet store is an intermediatory between the customers and sellers who are interested to sell pets.
8. Sellers have an ID, address, a name, pet they intend to sell, cost at which they sell and their phone numbers.
9. When a customer buys pet, there must be quantity of pet that customer intends to buy and their date of purchase is noted in a relation.
10. Similarly, sellers can sell their pets to the store and their date of selling is recorded in a relation.
11. The same goes with food. Customer can buy food for their pet whenever needed and the quantity of food and date of purchase is noted.

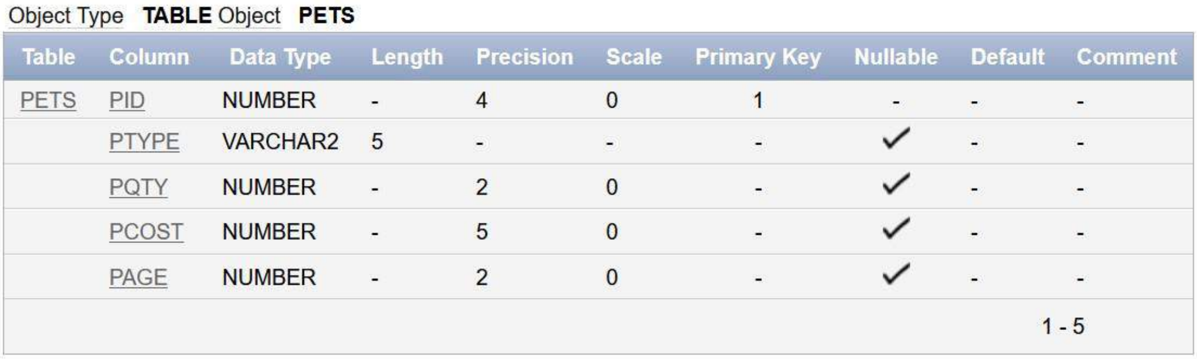
ER Diagram:


Food
FName
Fqty
Fcost
Fid
Fbuy
Fbqty
Fbdate
Customers
Cage
CPhno
CAdd
CName
Cid
Feed
Pbuy
Pbdate
Pbqty
Pets
Page
Pcost
Pqty
Ptype
Pid
Sell
Sdate
Sqty
Sellers
SPhno
Scost
Spet
SAdd
Sid
SName

Tables creation:

1. **Pets**

```
create table Pets(
        pid number(4),
        ptype varchar2(5),
        pqty number(2),
        pcost number(5),
        page number(2),
        constraint z1 primary key(pid)
        );
```



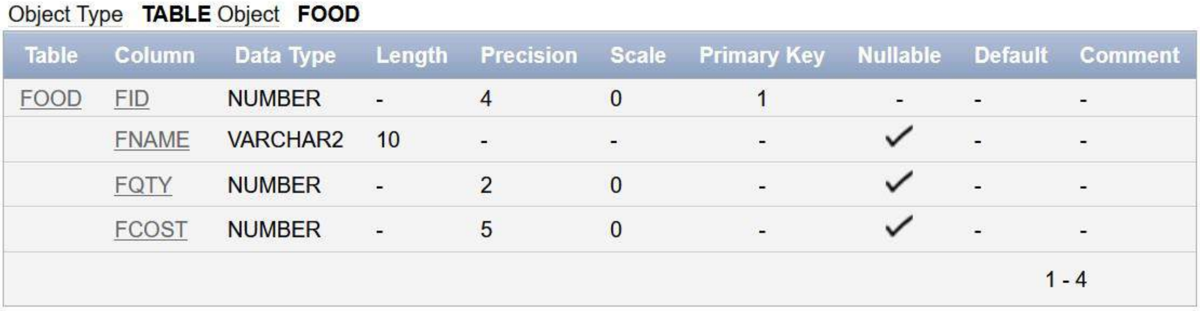
Object Type **TABLE** Object **PETS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PETS	PID	NUMBER	-	4	0	1	-	-	-
	PTYPE	VARCHAR2	5	-	-	-	✓	-	-
	PQTY	NUMBER	-	2	0	-	✓	-	-
	PCOST	NUMBER	-	5	0	-	✓	-	-
	PAGE	NUMBER	-	2	0	-	✓	-	-

1 - 5

2. **Food**

```
create table Food(
        fid number(4),
        fname varchar(10),
        fqty number(2),
        fcost number(5),
        constraint z2 primary key(fid)
        );
```



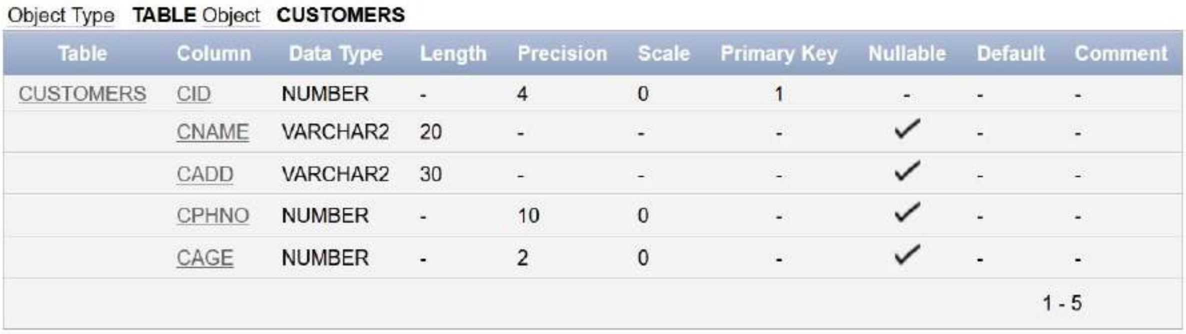
Object Type **TABLE** Object **FOOD**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
FOOD	FID	NUMBER	-	4	0	1	-	-	-
	FNAME	VARCHAR2	10	-	-	-	✓	-	-
	FQTY	NUMBER	-	2	0	-	✓	-	-
	FCOST	NUMBER	-	5	0	-	✓	-	-

1 - 4

3. **Customers**

```
create table Customers(
        cid number(4),
        cname varchar(20),
        cadd varchar(30),
        cphno number(10),
        cage number(2),
        constraint z3 primary key(cid)
        );
```



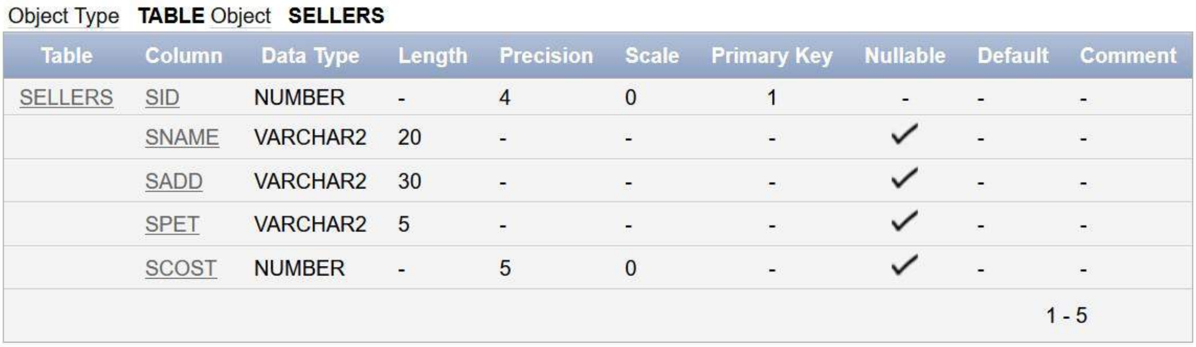
Object Type **TABLE** Object **CUSTOMERS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CUSTOMERS	CID	NUMBER	-	4	0	1	-	-	-
	CNAME	VARCHAR2	20	-	-	-	✓	-	-
	CADD	VARCHAR2	30	-	-	-	✓	-	-
	CPHNO	NUMBER	-	10	0	-	✓	-	-
	CAGE	NUMBER	-	2	0	-	✓	-	-

1 - 5

4. Sellers

```
create table Sellers(
        sid number(4),
        sname varchar(20),
        sadd varchar(30),
        spet varchar(5),
        scost number(5),
        constraint z4 primary key(sid)
        );
```



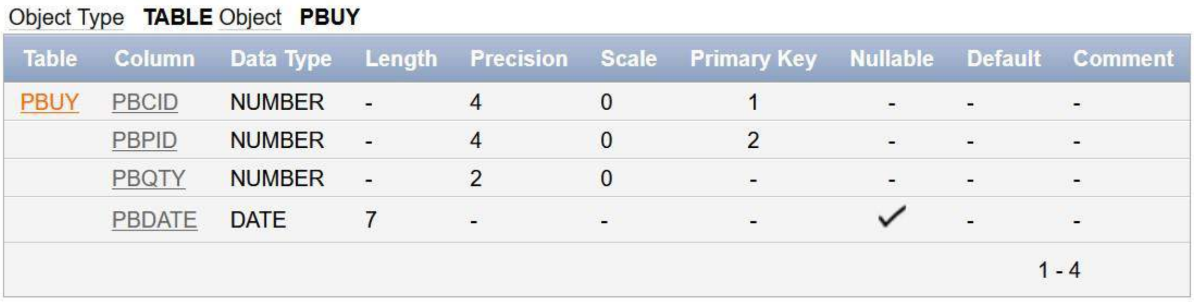
Object Type **TABLE** Object **SELLERS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SELLERS	SID	NUMBER	-	4	0	1	-	-	-
	SNAME	VARCHAR2	20	-	-	-	✓	-	-
	SADD	VARCHAR2	30	-	-	-	✓	-	-
	SPET	VARCHAR2	5	-	-	-	✓	-	-
	SCOST	NUMBER	-	5	0	-	✓	-	-

1 - 5

5. Pbuy(relation)

```
create table Pbuy(
        pbcid number(4),
        pbpid number(4),
        pbqty number(2),
        pbdate date,
        constraint z5 primary key(pbcid, pbpid),
        constraint zz5 foreign key(pbcid) references customers(cid),
        constraint zzz5 foreign key(pbpid) references pets(pid));
```



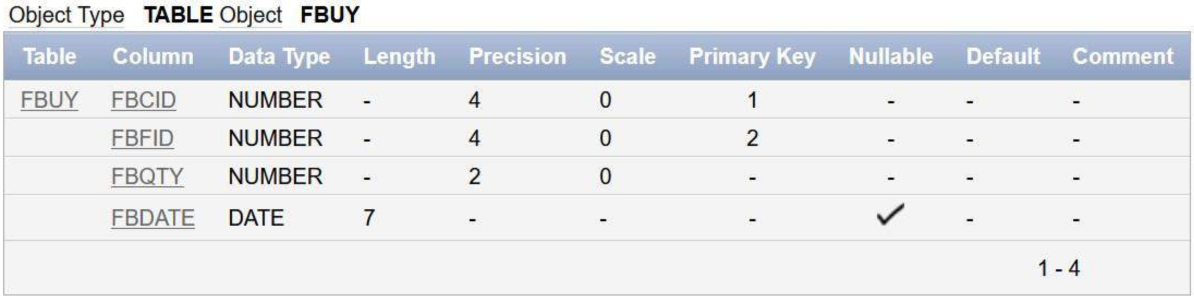
Object Type **TABLE** Object **PBUY**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PBUY	PBCID	NUMBER	-	4	0	1	-	-	-
	PBPID	NUMBER	-	4	0	2	-	-	-
	PBQTY	NUMBER	-	2	0	-	-	-	-
	PBDATE	DATE	7	-	-	-	✓	-	-

1 - 4

6. Fbuy(relation)

```
create table Fbuy(
            fbcid number(4),
            fbfid number(4),
            fbqty number(2),
            fbdate date,
            constraint z6 primary key(fbcid,fbfid),
            constraint zz6 foreign key(fbcid) references customers(cid),
            constraint zzz6 foreign key(fbfid) references food(fid)
            );
```



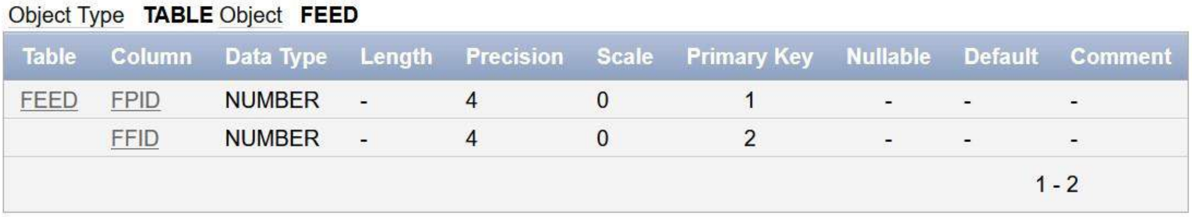
Object Type **TABLE** Object **FBUY**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
FBUY	FBCID	NUMBER	-	4	0	1	-	-	-
	FBFID	NUMBER	-	4	0	2	-	-	-
	FBQTY	NUMBER	-	2	0	-	-	-	-
	FBDATE	DATE	7	-	-	-	✓	-	-

1 - 4

7. Feed(relation)

```
create table Feed(
            fpid number(4),
            ffid number(4),
            constraint z7 primary key(fpid,ffid),
            constraint zz7 foreign key(fpid) references pets(pid),
            constraint zzz7 foreign key(ffid) references food(fid)
            );
```



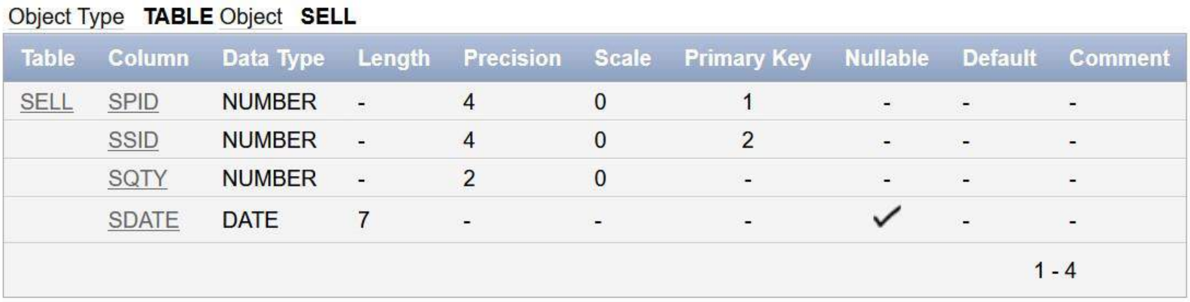
Object Type **TABLE** Object **FEED**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
FEED	FPID	NUMBER	-	4	0	1	-	-	-
	FFID	NUMBER	-	4	0	2	-	-	-

1 - 2

8. **Sell(relation)**

```
create table Sell(
        spid number(4),
        ssid number(4),
        sqty number(2),
        sdate date,
        constraint z8 primary key(spid,ssid),
        constraint zz8 foreign key(spid) references pets(pid),
        constraint zzz8 foreign key(ssid) references sellers(sid)
        );
```



Object Type **TABLE** Object **SELL**

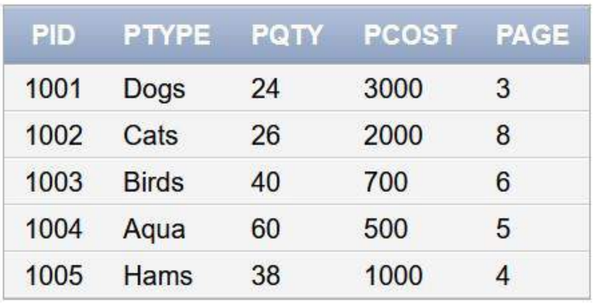
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SELL	SPID	NUMBER	-	4	0	1	-	-	-
	SSID	NUMBER	-	4	0	2	-	-	-
	SQTY	NUMBER	-	2	0	-	-	-	-
	SDATE	DATE	7	-	-	-	✓	-	-

1 - 4

Insertion:

Pets:

```
insert into pets values(1001,'Dogs',24,3000,3);

insert into pets values(1002,'Cats',26,2000,8);

insert into pets values(1003,'Birds',40,700,6);

insert into pets values(1004,'Aqua',60,500,5);

insert into pets values(1005,'Hams',38,1000,4);
```



PID	PTYPE	PQTY	PCOST	PAGE
1001	Dogs	24	3000	3
1002	Cats	26	2000	8
1003	Birds	40	700	6
1004	Aqua	60	500	5
1005	Hams	38	1000	4

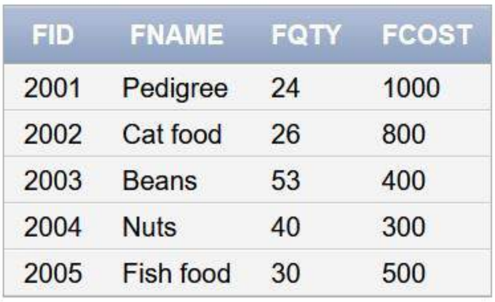
Food:

```
insert into food values(2001,'Pedigree',24,1000);

insert into food values(2002,'Cat food',26,800);
```

insert into food values(2003,'Beans',53,400);

insert into food values(1004,'Nuts',40,300);

insert into food values(1005,'Fish food',30,500);



FID	FNAME	FQTY	FCOST
2001	Pedigree	24	1000
2002	Cat food	26	800
2003	Beans	53	400
2004	Nuts	40	300
2005	Fish food	30	500

Customers:

insert into customers values(3001,'Zei Jhun','1-24-63,Ram Street,Vein',8932164378,23);

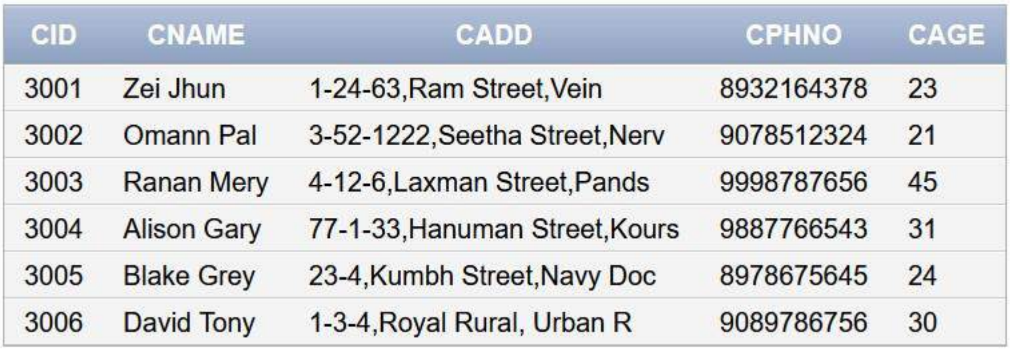
insert into customers values(3002,'Omann Pal','3-52-1222,Seetha Street,Nerv', 9078512324, 21);

insert into customers values(3003,'Ranan Mery','4-12-6,Laxman Street,Pands', 9998787656, 45);

insert into customers values(3004,'Alison Gary','77-1-33,Hanuman Street,Kours', 9887766543, 31);

insert into customers values(3005,'Blake Grey','23-4,Kumbh Street,Navy Doc', 8978675645, 24);

insert into customers values(3006,'David Tony','1-3-4,Royal Rural, Urban R', 9089786756, 30);



CID	CNAME	CADD	CPHNO	CAGE
3001	Zei Jhun	1-24-63,Ram Street,Vein	8932164378	23
3002	Omann Pal	3-52-1222,Seetha Street,Nerv	9078512324	21
3003	Ranan Mery	4-12-6,Laxman Street,Pands	9998787656	45
3004	Alison Gary	77-1-33,Hanuman Street,Kours	9887766543	31
3005	Blake Grey	23-4,Kumbh Street,Navy Doc	8978675645	24
3006	David Tony	1-3-4,Royal Rural, Urban R	9089786756	30

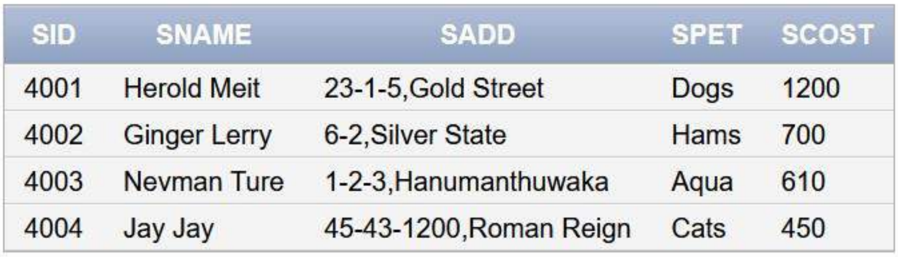
Sellers:

insert into sellers values(4001,'Herold Meit','23-1-5,Gold Street','Dogs',1200);

insert into sellers values(4002,'Ginger Lerry','6-2,Silver State','Hams',700);

insert into sellers values(4003,'Nevman Ture','1-2-3,Hanumanthuwaka','Aqua',610);

insert into sellers values(4004,'Jay Jay','45-43-1200,Roman Reign','Cats',450);



SID	SNAME	SADD	SPET	SCOST
4001	Herold Meit	23-1-5,Gold Street	Dogs	1200
4002	Ginger Lerry	6-2,Silver State	Hams	700
4003	Nevman Ture	1-2-3,Hanumanthuwaka	Aqua	610
4004	Jay Jay	45-43-1200,Roman Reign	Cats	450

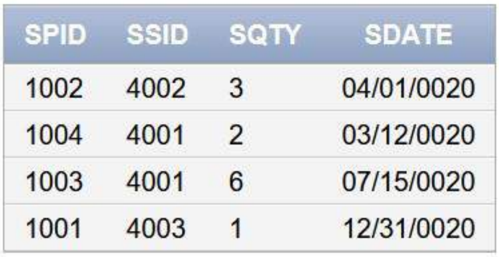
Sell:

insert into sell values(1004,4001,2,'MAR-12-20');

insert into sell values(1002,4002,3,'APR-1-20');

insert into sell values(1003,4001,6,'JUL-15-20');

insert into sell values(1001,4003,1,'DEC-31-20');



SPID	SSID	SQTY	SDATE
1002	4002	3	04/01/0020
1004	4001	2	03/12/0020
1003	4001	6	07/15/0020
1001	4003	1	12/31/0020

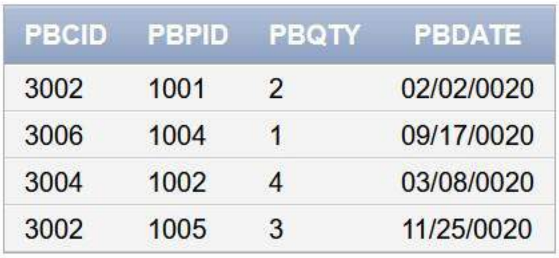
Pbuy

insert into pbuy values(3002,1001,2,'FEB-2-20');

insert into pbuy values(3006,1004,1,'SEP-17-20');

insert into pbuy values(3004,1002,4,'MAR-8-20');

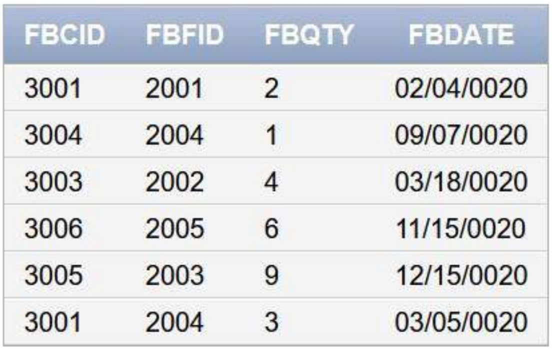
insert into pbuy values(3002,1005,3,'NOV-25-20');



PBCID	PBPID	PBQTY	PBDATE
3002	1001	2	02/02/0020
3006	1004	1	09/17/0020
3004	1002	4	03/08/0020
3002	1005	3	11/25/0020

Fbuy

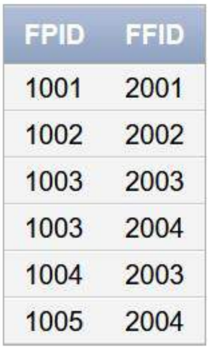
```
insert into fbuy values(3001,2001,2,'FEB-4-20');

insert into fbuy values(3004,2004,1,'SEP-7-20');

insert into fbuy values(3003,2002,4,'MAR-18-20');

insert into fbuy values(3006,2005,6,'NOV-15-20');

insert into fbuy values(3005,2003,9,'DEC-15-20');

insert into fbuy values(3001,2004,3,'MAR-5-20');
```



FBCID	FBFID	FBQTY	FBDATE
3001	2001	2	02/04/0020
3004	2004	1	09/07/0020
3003	2002	4	03/18/0020
3006	2005	6	11/15/0020
3005	2003	9	12/15/0020
3001	2004	3	03/05/0020

Feed

```
insert into feed values(1004,2003);

insert into feed values(1002,2002);

insert into feed values(1005,2004);

insert into feed values(1001,2001);

insert into feed values(1003,2003);

insert into feed values(1003,2004);
```



FPID	FFID
1001	2001
1002	2002
1003	2003
1003	2004
1004	2003
1005	2004

Normalization:

Normalization is the process of minimizing redundancy from a relation or set of relations. Redundancy in relation may cause insertion, deletion and update anomalies. So, it helps to minimize the redundancy in relations. Normal forms are used to eliminate or reduce redundancy in database tables.

There are various types of normal forms, some of them are still under construction.

1. First Normal Form (1NF)
2. Second Normal Form (2NF)
3. Third Normal Form (3NF)
4. Boyce-Codd Normal Form (BCNF)

First Normal Form (1NF):
If a relation contains a composite or multi-valued attribute, it violates the first normal form.

A table is in 1 NF if and only if:

1. There are only Single Valued Attributes.
2. Attribute Domain does not change.
3. There is a unique name for every Attribute/Column.
4. The order in which data is stored does not matter.

Second Normal Form (2NF):
Second Normal Form (2NF) is based on the concept of full functional dependency. Second Normal Form applies to relations with composite keys, that is, relations with a primary key composed of two or more attributes. A relation with a single-attribute primary key is automatically in at least 2NF. A relation that is not in 2NF may suffer from the update anomalies.

To be in second normal form, a relation must be in first normal form and relation must not contain any partial dependency. A relation is in 2NF if it has No Partial Dependency, i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.

Third Normal Form (3NF):
A relation is in third normal form, if there is no transitive dependency for non-prime attributes as well as it is in second normal form.

A relation is in 3NF if at least one of the following condition holds in every non-trivial function dependency X –> Y:

1. X is a super key.
2. Y is a prime attribute (each element of Y is part of some candidate key).

Normalizing the tables:

Pets: Funtional depend of this table is,

pid → ptype, pqty, pcost, page

where pid is the primary key. As all the non-prime attributes are solely dependent on the primary key or candidate key(if here) and there are no partial and transitive dependencies, it is in 3rd normal form.

Food: Funtional depend of this table is,

fid → fname, fqty, fcost

where fid is the primary key. As all the non-prime attributes are solely dependent on the primary key or candidate key(if here) and there are no partial and transitive dependencies, it is in 3rd normal form.

Customers: Funtional depend of this table is,

cid → cname, cadd, cphno, cage

where cid is the primary key. As all the non-prime attributes are solely dependent on the primary key or candidate key(if here) and there are no partial and transitive dependencies, it is in 3rd normal form.

Normalizing the relations:

Pbuy: Funtional depend of this table is,

(pbcid, pbpid) → pbqty, pbdate

where pbcid and pbpid together form primary key, which means it's the candidate key. Here, no non-prime attribute is dependent on the non-prime attribute by transitivity and there isn't any partial dependency. Hence, it is in 3rd normal form.

Fbuy: Funtional depend of this table is,

(fbcid, fbfid) → fbqty, fbdate

where fbcid and fbfid together form primary key, which means it's the candidate key. Here, no non-prime attribute is dependent on the non-prime attribute by transitivity and there isn't any partial dependency. Hence, it is in 3rd normal form.

Feed: Funtional depend of this table is,

(fpid, ffid) → fpid, ffid

There is no other attribute in this table, indicating that it is in higher normal form than the others, it is in Boyce-Codd Normal Form (BCNF).

Conclusion:

Pet Store Management System is a mini database management system, which is developed to help small scale pet shop owner to keep track of the available pet and its products along with the sales details of his shop.

This project is helpful for the stores handling the pets to be sold and bought. It makes the store manager to easily maintain the data of purchase and sales even with the data of customers, sellers, food for the pets and many other essential data needed to manage the store.

References:

- Profitableventure - https://www.profitableventure.com/pet-food-shop-business-plan/
- Relationaldbdesign - https://www.relationaldbdesign.com/programming-plsql/module1/database-pet-store-schema.php
- Clickedidia - https://www.clickedindia.net/Visakhapatnam/pet-store-C690.htm
- Sdrventures - https://sdrventures.com/newsletters/pet-reports