Game Shop Database

April 24, 2014

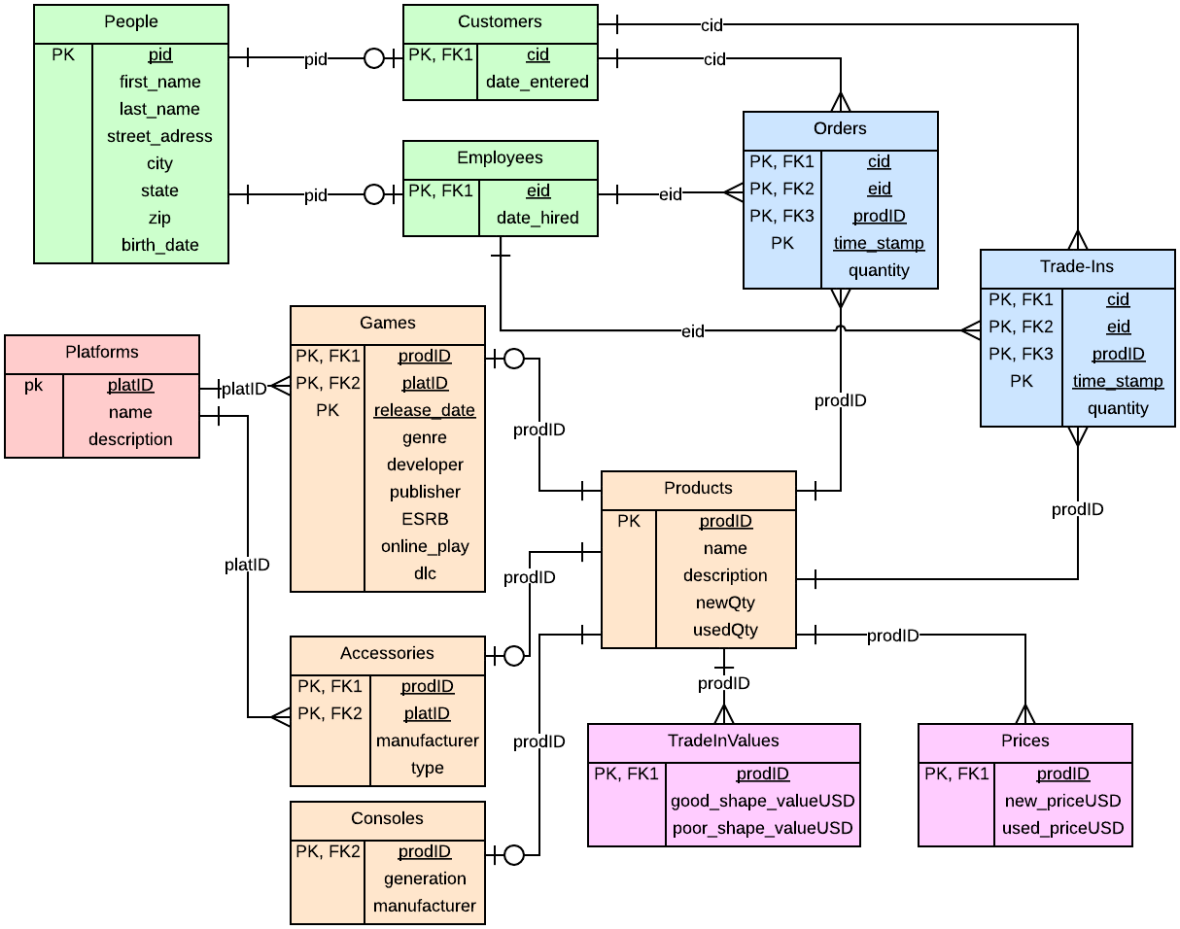
Christopher Barnett

Executive Summary

This is a database design for a store that sells video games and video game accessories. The store sells both new and used products, and will also buy used products from customers.

The overall goal of this database is to provide the store with an quick and easy way to access and update information about products, inventory, customers, employees, and transactions. In order to accomplish this, the data needs to be stored in a well structured, centralized database that can be easily accessed and will take measures to ensure that inconsistent data is kept to a minimum.

Entity Relationship Diagram



Tables

People

Purpose

This table will hold the basic information for all people entered in the system. The information presented here is fairly standard.

Create Statement

```
CREATE TABLE people (
  pid             char(4) not null,
  first_Name      text,
  last_Name       text,
  street_address  text,
  city            text,
  state           char(2),
  zip             char(5),
  birth_Date      date,
 primary key(pid)
);
```

Functional Dependencies

pid → first_Name, last_Name, street_address, city, state, zip, birth_date

Sample Data

	pid character(4)	first_name text	last_name text	street_address text	city text	state character(2)	zip character(5)	birth_date date
1	p001	Jason	Smith	123 Fake St.	Poughkeepsie	NY	12603	1953-05-16
2	p002	Pierce	Brosnan	40 Real St.	Wappinger Falls	NY	12590	1982-07-12
3	p003	Bob	Smith	300 Fake Rd.	Somewhere	CA	90000	1992-12-21
4	p004	Donald	Jump	80 Real Rd.	Wappinger Falls	NY	12590	2006-07-12
5	p005	Jet	Black	33 Bebop St.	Nowhere	RI	00000	1989-09-25
6	p006	Cave	Johnson	1 Aperture St	Poughkeepse	NY	12601	1942-01-01

Customers

Purpose

This table will hold specific information that only customers have. The date_entered field is the date that the customer's information was first registered in the database.

Create Statement

```
CREATE TABLE customers (
  cid           char(4) not null references people(pid),
  date_Entered date,
 primary key(cid)
);
```

Functional Dependencies

cid → date_entered

Sample Data

	cid character(4)	date_entered date
1	p001	2010-05-16
2	p002	2011-01-12
3	p003	2011-02-27
4	p004	2011-05-06
5	p005	2012-10-31

Employees

Purpose

This table will hold specific information that only employees have. The date_hired field is the date that the employee's information was first registered in the database.

Create Statement

```
CREATE TABLE employees (
  eid           char(4) not null references people(pid),
  date_hired    date,
 primary key(eid)
);
```

Functional Dependencies

eid → date_hired

Sample Data

	eid character(4)	date_hired date
1	p001	2010-06-1
2	p002	2011-02-1
3	p006	2012-08-0

Platforms

Purpose

This table will hold information regarding what gaming platform the product works with. This table will provide the full name of the platform as well as a description of the platform.

Create Statement

```
CREATE TABLE platforms (
  platID       char(4) not null,
  name         text,
  description text,
 primary key(platID)
);
```

Functional Dependencies

platid → name, description

Sample Data

	platid character(4)	name text	description text
1	NES	Nintendo Entertainment System	First home video game console released by Nintendo.
2	SNES	Super Nintendo Entertainment System	Second home video game console released by Nintendo.
3	N64	Nintendo 64	Third home video game console released by Nintendo.
4	X360	Xbox 360	Second home video game console released by Microsoft.
5	PS3	Playstation 3	Third home video game console released by Sony.
6	DC	Sega Dreamcast	The best thing ever.

Products

Purpose

This table will hold the basic information that all products share. The newQty field keeps track of the quantity of brand new items the store has. The usedQty field keeps track of the used items that have been purchased from customers.

Create Statement

```
CREATE TABLE products (
  prodID          char(4) not null,
  name            text,
  description     text,
  newQty          int,
  usedQty         int,
 primary key(prodID)
);
```

Functional Dependencies

prodID → name, description, newQty, usedQty

Sample Data

	prodid character(4)	name text	description text	newqty integer	usedqty integer
1	0001	Super Mario Bros.	A classic platforming game.	0	3
2	0002	Final Fantasy VI.	An iconic RPG	0	1
3	0003	Super Smash Bros.	The original Smash Bros. game.	0	2
4	0004	Portal 2	A first person puzzle game.	10	2
5	0005	Metal Gear Solid HD Collection	A combination of several classic Metal Gear Solid games in HD.	20	4
6	0006	Sonic Adventure	A 3D platformer.	0	2
7	0007	Xbox 360 Wireless Controller.	Wireless Game Controller.	10	3
8	0008	Playstation 4	The latest console from Sony	1	0
9	0009	Xbox One	The latest console from Microsoft.	20	10

Games

Purpose

This table will hold the specific information for video games.

Create Statement

```
 CREATE TABLE games (
  prodID         char(4) not null references products(prodID),
  platID         char(4) not null references platforms(platID),
  release_date   date not null,
  genre          text,
  developer      text,
  publisher      text,
  ESRB           char(3),
  online_play    boolean,
  dlc            boolean,
 primary key(prodID, platID, release_date)
);
```

Functional Dependencies

prodID, platID, release_date → genre, developer, publisher, ESRB, online_play, dlc

Sample Data

	prodid character(4)	platid character(4)	release_date date	genre text	developer text	publisher text	esrb character(3)	online_play boolean	dlc boolean
1	0001	NES	1985-09-13	Platforming	Nintendo	Nintendo	E	f	f
2	0002	SNES	1994-08-20	Role-playing	Square	Square	E10	f	f
3	0003	N64	1999-04-26	Fighting	HAL Laboratory	Nintendo	E	f	f
4	0004	X360	2011-04-19	Puzzle	Valve	Valve	E10	t	t
5	0005	PS3	2011-11-08	Action Adventure	Kojima Productions	Konami	M	f	f
6	0006	DC	1999-12-23	Platformer	Sonic Team	Sega	E	f	f

Accessories

Purpose

This table will hold the specific information for peripherals that work with various video game platforms. These could be items such as controllers, memory cards, cables, etc.

Create Statement

```
 CREATE TABLE accessories (
  prodID        char(4) not null references products(prodID),
  platID        char(4) not null references platforms(platID),
  manufacturer  text,
  type          text,
 primary key(prodID, platID)
);
```

Functional Dependencies

prodID, platID, → manufacturer, type

Sample Data

	prodid character(4)	platid character(4)	manufacturer text	type text
1	0007	X360	Microsoft	Controller

Consoles

Purpose

This table will hold the specific information for gaming consoles that are for sale.

Create Statement

```
CREATE TABLE consoles (
 prodID         char(4) not null references products(prodID),
 generation     int,
 manufacturer   text,
primary key(prodID)
);
```

Functional Dependencies

prodID → generation, manufacturer

Sample Data

	prodid character(4)	generation integer	manufacturer text
1	0008	8	Sony
2	0009	8	Microsoft

Prices

Purpose

This table will hold the pricing information for all products. This includes both the price for brand new products as well as the price for used products sold by the store.

Create Statement

```
 CREATE TABLE prices (
  prodID         char(4) not null references products(prodID),
  new_priceUSD   numeric(12,2),
  used_priceUSD numeric(12,2),
 primary key(prodID)
);
```

Functional Dependencies

prodID → new_priceUSD, used_priceUSD

Sample Data

	prodid character(4)	new_priceusd numeric(12,2)	used_priceusd numeric(12,2)
1	0001	100.00	20.00
2	0002	70.00	30.00
3	0003	60.00	30.00
4	0004	40.00	20.00
5	0005	40.00	25.00
6	0006	70.00	20.00
7	0007	50.00	25.00
8	0008	400.00	380.00
9	0009	500.00	480.00

TradeInValues

Purpose

This table will hold the values that the store will pay to customers who trade in their used games, accessories, and game consoles. The quality of the used item will determine how much money the customer will get for trading in that product

Create Statement

```
 CREATE TABLE tradeinvalues (
  prodID                char(4) not null references products(prodID),
  good_shape_valueUSD   numeric(12,2),
  poor_shape_valueUSD   numeric(12,2),
 primary key(prodID)
);
```

Functional Dependencies

prodID → good_shape_valueUSD, poor_shape_valueUSD

Sample Data

	prodid character(4)	good_shape_valueusd numeric(12,2)	poor_shape_valueusd numeric(12,2)
1	0001	20.00	10.00
2	0002	20.00	10.00
3	0003	15.00	9.00
4	0004	7.00	5.00
5	0005	6.00	4.00
6	0006	20.00	10.00
7	0007	11.00	7.00
8	0008	180.00	125.00
9	0009	180.00	125.00

Orders

Purpose

This table will keep track of purchases made by customers. The table will keep track of who made the purchase, which employee handled the purchase, what the customer purchased, when the order occurred, and how many items the customer purchased.

Create Statement

```
 CREATE TABLE orders (
  cid         char(4) not null references customers(cid),
  eid         char(4) not null references employees(eid),
  prodID      char(4) not null references products(prodID),
  time_stamp timestamp without time zone not null,
  quantity    int not null,
 primary key(cid, eid, prodID, time_stamp)
);
```

Functional Dependencies

cid, eid, prodID, time_stamp → quantity

Sample Data

	cid character(4)	eid character(4)	prodid character(4)	time_stamp timestamp without time zone	quantity integer
1	p001	p002	0009	2014-04-24 18:22:22.141	1
2	p002	p006	0003	2014-04-24 18:24:44.137	1
3	p005	p001	0006	2014-04-24 18:25:31.978	1

TradeIns

Purpose

This table will keep track of product trade-ins made by customers. The table will keep track of who made traded-in the product, which employee handled the trade-in, what the customer traded-in, when the trade-in occurred, and how many items the customer traded-in.

Create Statement

```
 CREATE TABLE tradeins (
  cid         char(4) not null references customers(cid),
  eid         char(4) not null references employees(eid),
  prodID      char(4) not null references products(prodID),
  time_stamp timestamp without time zone not null,
  quantity    int not null,
 primary key(cid, eid, prodID, time_stamp)
);
```

Functional Dependencies

cid, eid, prodID, time_stamp → quantity

Sample Data

	cid character(4)	eid character(4)	prodid character(4)	time_stamp timestamp without time zone	quantity integer
1	p003	p002	0005	2014-04-24 18:28:17.321	1
2	p004	p006	0007	2014-04-24 18:29:22.055	1

Views

oldGameInventory

Purpose

This view will display the stock levels of games released before the year 2000.

Create Statement

```
create view oldGameInventory as
select p.prodid as Product_ID, p.name as Game, p.newQty as
New_Copies, p.usedQty as Used_Copies,
       plat.name as Platform, g.publisher as Publisher, g.
       release_date
from products p, platforms plat, games g
where p.prodid = g.prodid
```

Reports

GamesSoldByEmployee

Purpose

This query will list the games sold by an individual employee

Query

```
select p.name, g.prodid, g.platid, g.genre
from games g, products p, orders o
where g.prodid = p.prodID
  and p.prodid = o.prodID
  and o.eid = 'p001'
order by p.name asc;
```

Platformers

Purpose

This query will list the names of customers who purchased platforming games

Query

```
select first_name, last_name
from people
where pid in(select cid
        from customers
        where cid in(select cid
            from orders
            where prodid in(select prodid
                    from products
                    where prodid in (select prodid
                                from games
                                where genre = 'Platforming')
                    )
            )
        )
order by first_name asc;
```

Stored Procedures

getAge()

Purpose

Calculates a user's age based on their date of birth compared to the current date. This is important for determining if a customer is eligible to buy certain games.

Create Statement

```
create function getAge(char(4)) returns integer as
$$
declare
   personID  char(4)   := $1;
   age integer;

begin
   select extract(year from age((select birth_date
                  from people
                  where pid = personID))) into age;
   return age;
end;
$$
language plpgsql;
```

Triggers

matureGame

Purpose

Detects if customer is old enough to buy a game that has a Mature ESRB rating. If a customer is under the age of 17, they are not allowed to purchase a Mature rated game. This trigger will stop the insert of a new order if the customer is too young.

Create Statement

```
create or replace function matureGame() returns trigger as $
mature_game$
declare
    age integer;
    rating char(3);
begin
    select getAge(new.cid) into age;

    select ESRB
    from games
    where games.prodid = new.prodid into rating;

    if  age < 17 and rating = 'M'
    then
       raise exception 'This customer is under 17 and cannot
       purchase Mature games.';
    end if;
    return new;
end;
$mature_game$ language plpgsql;

create trigger mature_game before insert or update on orders
    for each row execute procedure matureGame();
```

Security

In order to ensure security, two different types of users will be created.

admin

Purpose

The administrator will have complete access to the entire database in order to maintain and expand it.

Create Statement

```
create role admin
grant select, update, insert, alter
on all tables in schema public
to admin;
```

standard_user

Purpose

A standard user can only select information from the database and cannot edit it. This is to protect the data from unauthorized changes.

Create Statement

```
create role standard_user
grant select
on all tables in schema public
to standard user;
```

Known Problems

- It is currently not possible to have a game fall under more than one genre.
- It is not possible for a game to have more than one developer
- There is currently no way to stop a customer from ordering more products than the store has inventory for.

Future Enhancements

- Add a way to alter the inventory levels of new and used products when orders and trade-ins occur.
- Add methods to keep track of revenue from sales and business expenses.
- Add ability to track shipments of new games from suppliers.
- Add support for online shopping