QuickFill Database Design

Created on: April, 24, 2016

Author: Vallie M Joseph

Table of Contents

Executive Summary

Overview

QuikFill is a new, up and coming gas station that offers its patrons a fast and easy automated gas filling service. When a customer arrives, they pull into the drive-through style building while either an attendant or gas robot fill their tank, while also allowing for in-car shopping through the convenient store attached

Goals and Objectives

<Goals and objectives of the system – these were determined during requirements analysis. The goals and objectives should result from the system request and requirements determination, including interviews with company representatives. >

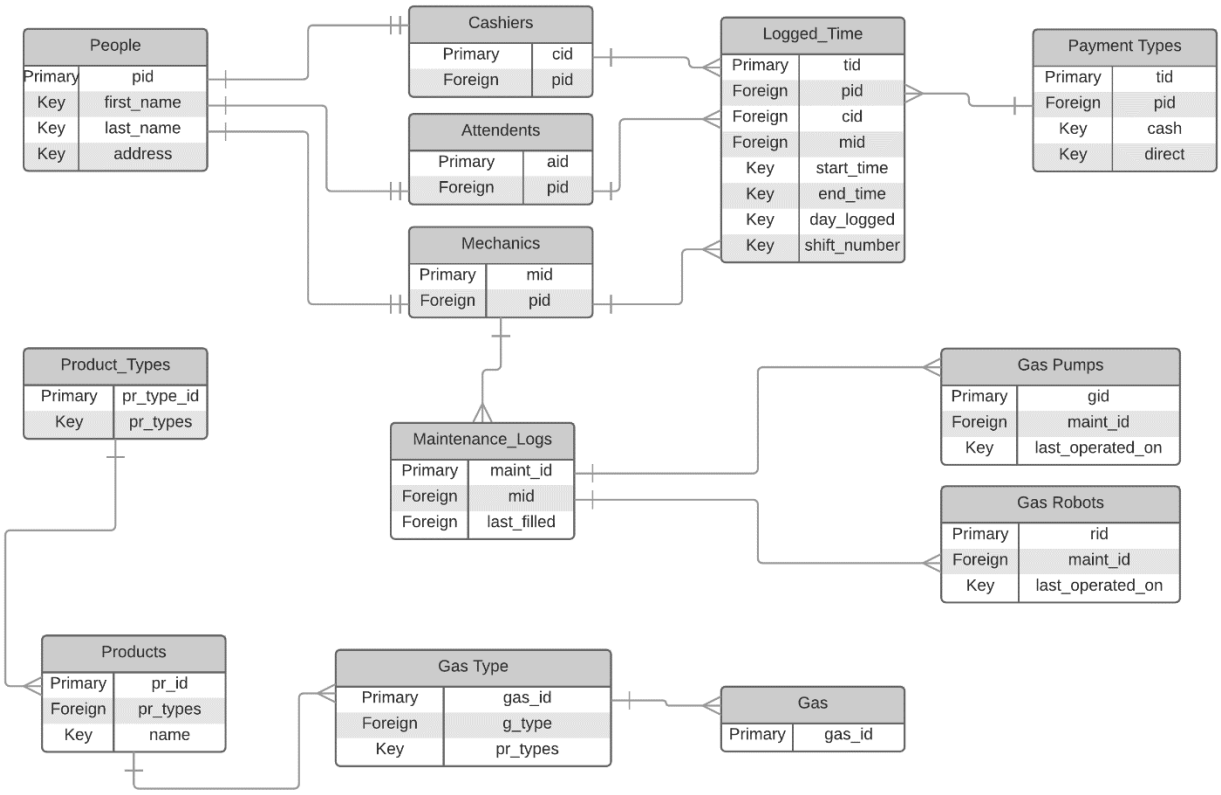
< Business problems and needs of your client which the recommended system will address and solve -You are going to provide a brief description of the problems you discovered with respect to their existing system.>

<What the new system will do – The functional benefits of the new system>

<How the new system will solve/address your client business problems/needs add value to the business – Explain how the new system addresses and solves the functional and nonfunctional problems your client is encountering. Important – your recommended system must solve all of the problems/needs you identify. >

ENTITY RELATIONSHIP

Entity Relationship Diagram



Tables

People

Description:

The purpose of the people table to store the first and last names of all employees as well as their addresses.

Create Statement

```
CREATE TABLE if not exists if not exists people (
  pid int NOT NULL,
  first_name VARCHAR(50) NOT NULL,
  last_name VARCHAR(50) NOT NULL,
  address  VARCHAR(100) NOT NULL,
  PRIMARY KEY( pid )
);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	PID
Functional Dependencies:	PID → First Name, Last Name, Address

Data Example:

	pid integer	first_name character varying(50)	last_name character varying(50)	address character varying(100)
1	1	Jane	Doe	123 Example Drive NYC, NY 12500
2	2	John	Doe	456 Sample Street Albany, NY 12700
3	3	John	Doe	789 Placeholder Ave Yonkers, NY 12810
4	4	Melanie	Sutherland	5877 8th Street South Stillwater, MN 55082
5	5	Jane	Bower	2787 William Street Olive Branch, MS 38654
6	6	Angela	Metcalfe	1254 Myrtle Avenue Mountain View, CA 94043
7	8	Madeleine	Peake	62 North Street Bradenton, FL 34203
8	9	Rachel	Robertson	435 Catherine Street East Hartford, CT 06118

Cashiers

Description:

The purpose of the cashier table is to hold all employees who are cashiers.

Create Statement:

```
CREATE TABLE if not exists cashiers (
  cid int NOT NULL,
  pid int NOT NULL references people(pid),
   PRIMARY KEY(cid)
);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	CID
Foreign Key:	**PID**
Functional Dependencies:	CID → PID

Data Example:

	pid integer	first_name character varying(50)	last_name character varying(50)	address character varying(100)	cid integer	pid integer
1	2	John	Doe	456 Sample Street Albany, NY 12700	2	2
2	4	Melanie	Sutherland	5877 8th Street South Stillwater, MN 55082	4	4

Attendants

Description:

The purpose of the cashier table is to hold all employees who are attendants.

Create Statement:

```
CREATE TABLE if not exists attendants (
  aid int NOT NULL,
  pid int NOT NULL references people(pid),
   PRIMARY KEY(aid)
);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	AID
Foreign Key:	**PID**
Functional Dependencies:	AID → PID

Data Example

	pid integer	first_name character varying(50)	last_name character varying(50)	address character varying(100)	aid integer	pid integer
1	1	Jane	Doe	123 Example Drive NYC, NY 12500	1	1
2	3	John	Doe	789 Placeholder Ave Yonkers, NY 12810	3	3

Mechanics

Description:

The purpose of the cashier table is to hold all employees who are mechanics.

Create Statement:

```
CREATE TABLE if not exists mechanics (
  mid int NOT NULL,
  pid int NOT NULL references people(pid),
   PRIMARY KEY(mid)
);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	MID
Foreign Key:	**PID**
Functional Dependencies:	MID → PID

Data Example

	pid integer	first_name character varying(50)	last_name character varying(50)	address character varying(100)	mid integer	pid integer
1	5	Jane	Bower	2787 William Street Olive Branch, MS 38654	5	5
2	8	Madeleine	Peake	62 North Street Bradenton, FL 34203	8	8

Logged Time

Description:

The purpose of the logged time table is to store all shift and shift hours completed by the Gas&Go employees. The table will be able to hold when the employee clocked in and out, as well as how many shifts they have completed. The clock in and out times will be stored as a timestamp so that both the day and hour of the day will be recorded. This allows the data to be stored in the most basic of forms data so that it can be manipulated to calculate total time that will later be sent to the financial software Gas&Go will utilize. Each time entry will have a different id to help in differentiating shifts.

Create Statement:

```
CREATE TABLE if not exists logged_time (
 tid int NOT NULL,
 pid int NOT NULL references people(pid),
 cid int NOT NULL references cashiers(cid),
 mid int NOT NULL references mechanics(mid),
 start_time TIMESTAMP NOT NULL,
 end_time TIMESTAMP NOT NULL,
 day_logged DATE NOT NULL,
 shift_number int NOT NULL,
  PRIMARY KEY(tid)
);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	TID
Foreign Key:	**PID, CID, MID**
Functional Dependencies:	TID → start_time, end_time, day_logged, shift_number

Data Example

	tid integer	pid integer	cid integer	mid integer	start_time timestamp without time zone	end_time timestamp without time zone	day_logged date	shift_number integer	aid integer
1	1	5	<NULL>	5	2016-04-08 08:23:54	2016-04-08 16:05:00	2016-04-08	1	<NULL>
2	2	2	2	<NULL>	2016-04-08 08:23:54	2016-04-08 16:05:00	2016-04-08	1	<NULL>
3	3	3	<NULL>	<NULL>	2016-04-08 08:23:54	2016-04-08 16:05:00	2016-04-08	3	3
4	4	4	4	<NULL>	2016-04-08 08:23:54	2016-04-08 16:05:00	2016-04-08	1	<NULL>
5	5	3	<NULL>	<NULL>	2016-04-18 09:23:54	2016-04-18 17:05:00	2016-04-08	1	3
6	6	8	<NULL>	8	2016-04-07 07:30:14	2016-04-07 14:20:10	2016-04-07	1	<NULL>

Product Types

Description:

The purpose of the product types table is to give a description or category to items that will be sold within the convenience store as well as the gas itself.

Create Statement:

```
CREATE TABLE if not exists product_types (
  pr_type_id int NOT NULL,
  pr_types varchar(50) NOT NULL,
  PRIMARY KEY(pr_type_id)
);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	TID
Foreign Key:	**PID, CID, MID**
Functional Dependencies:	TID → start_time, end_time, day_logged, shift_number

Data Example

	pr_type_id integer	pr_types character varying(50)
1	1	snack
2	2	drink
3	3	gas
4	4	food
5	5	meat
6	6	parishables
7	7	pastries

Products

Description:

The purpose of the products table will be to hold the actual individual products that the convenience side part of the store will sell .

Create Statement:

```
create table products (

pr_id int not null,

pr_type_id int not null references product_types(pr_type_id),

PRIMARY KEY (pr_id)

);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	Pr_id
Foreign Key:	
Functional Dependencies:	TID → start_time, end_time, day_logged, shift_number

Data Example

	pr_id integer	pr_type_id integer	title character varying(50)	pr_type_id integer	pr_types character varying(50)
1	1	1	chips	1	snack
2	2	2	mountain dew	2	drink
3	3	3	gas	3	gas
4	4	4	jerkey	4	food
5	5	5	hotdogs	5	meat
6	6	6	milk	6	parishables
7	7	7	donuts	7	pastries

Gas Type

Description:

The purpose of the gas type table is to classify all incoming gas into categories. This allows for easy access for gas information of varying types.

Create Statement:

```
CREATE TABLE if not exists gas_type (
 g_type varchar(50) NOT NULL,
 pr_types varchar(50) NOT NULL,
 PRIMARY KEY(g_type)
);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	G_tye
Foreign Key:	**Pr_types**
Functional Dependencies:	G_type→pr_types

Data Example

	gas_id integer	g_type character varying(50)	pr_type_id integer
1	1	premium	3
2	2	regular	3
3	3	diesel	3
4	4	ethanol	3
5	5	Octane 78	3

Gas

Description:

The purpose of the product types table is to give a description or category to items that will be sold within the convenience store as well as the gas itself.

Create Statement:

```
CREATE TABLE if not exists gas (
  gid int NOT NULL,
  g_type  varchar(50) NOT NULL references gas_type(g_type),
  PRIMARY KEY(gid)
);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	gid
Foreign Key:	**gtype**
Functional Dependencies:	Gid→g_type

Data Example

	gid integer	g_type character varying(50)	gas_id integer	g_type character varying(50)	pr_type_id integer
1	1	2	1	premium	3
2	2	1	2	regular	3
3	3	2	3	diesel	3
4	4	5	4	ethanol	3
5	5	3	5	Octane 78	3

Maintenance Logs

Description:

The purpose of the product types table is to give a description or category to items that will be sold within the convenience store as well as the gas itself.

Create Statement:

```
CREATE TABLE maintenance_logs(
maint_id int NOT NULL,
mid int not null,
last_filled TIMESTAMP,
PRIMARY KEY (maint_id)
)
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	Maint_id
Foreign Key:	**mid**
Functional Dependencies:	Maint_id → last_filled

Data Example

	maint_id integer	mid integer	last_filled timestamp without time zone
1	1	5	2016-07-26 09:15:00
2	2	8	2016-10-16 15:50:29

Gas Pumps

Description:

The purpose of the product types table is to give a description or category to items that will be sold within the convenience store as well as the gas itself.

Create Statement:

```
CREATE TABLE if not exists gas_pumps (
  gid int NOT NULL references gas(gid),
  maint_id int NOT NULL references maintenance_logs(maint_id),
  last_operated_on Timestamp NOT NULL,
  PRIMARY KEY(gid,maint_id)
);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	gid
Foreign Key:	**Maint_id, last_operated_on**
Functional Dependencies:	gid → maint_id, last_operated_on

Data Example

	gid integer	maint_id integer	last_operated_on timestamp without time zone	maint_id integer	mid integer	last_filled timestamp without time zone
1	1	1	2016-07-26 13:30:12	1	5	2016-07-26 09:15:00
2	2	2	2016-07-26 13:30:12	2	8	2016-10-16 15:50:29

Gas Robots

Description:

The purpose of the product types table is to give a description or category to items that will be sold within the convenience store as well as the gas itself.

Create Statement:

```
CREATE TABLE if not exists gas_robots (
  rid int NOT NULL,
  maint_id  int NOT NULL references maintenance_logs(maint_id),
  last_operated_on Timestamp NOT NULL,
  PRIMARY KEY(rid)
);
```

Primary Keys, Foreign Keys and Functional Dependencies:

Primary Key:	rid
Foreign Key:	**Maint_id, last_operated_on**
Functional Dependencies:	rid → maint_id, last_operated_on

Data Example

	rid integer	maint_id integer	last_operated_on timestamp without time zone
1	1	2	2016-01-25 17:12:29
2	2	1	2016-01-25 17:12:29

Staff who Hold Positions

Create Statement

```
CREATE VIEW StaffwPositions AS

SELECT pid, first_name, last_name, address FROM people WHERE pid IN (select pid from attendants)

OR pid IN (select pid from mechanics)

OR pid in (SELECT pid FROM cashiers);
```

Data Example:

	pid integer	first_name character varying(50)	last_name character varying(50)	address character varying(100)
1	1	Jane	Doe	123 Example Drive NYC, NY 12500
2	2	John	Doe	456 Sample Street Albany, NY 12700
3	3	John	Doe	789 Placeholder Ave Yonkers, NY 12810
4	4	Melanie	Sutherland	5877 8th Street South Stillwater, MN 55082
5	5	Jane	Bower	2787 William Street Olive Branch, MS 38654
6	8	Madeleine	Peake	62 North Street Bradenton, FL 34203

Views

Staff Filled Positions

Create Statement

```
CREATE VIEW StaffFilledPositions AS

SELECT p.pid, p.first_name, last_name, m.jobtitle from people p

INNER JOIN mechanics m on p.pid=m.pid

Union

SELECT p.pid, p.first_name, last_name, a.jobtitle from people p

INNER JOIN attendants a on p.pid=a.pid

Union

SELECT p.pid, p.first_name, last_name, c.jobtitle from people p

INNER JOIN cashiers c on p.pid=c.pid

order by pid asc
```

Data Sample:

	pid integer	first_name character varying(50)	last_name character varying(50)	jobtitle character(50)
1	1	Jane	Doe	attendants
2	2	John	Doe	cashiers
3	3	John	Doe	attendants
4	4	Melanie	Sutherland	cashiers
5	5	Jane	Bower	mechanic
6	8	Madeleine	Peake	mechanic

Product Inventory

Create Statement

```
CREATE VIEW product_info AS

select t.pr_type_id, p.title, t.pr_types, p.qty

FROM product_types t

LEFT JOIN products p on p.pr_type_id=t.pr_type_id
```

Data Sample:

	pr_type_id integer	title character varying(50)	pr_types character varying(50)	qty integer
1	1	chips	snack	50
2	2	mountain dew	drink	999
3	3	gas	gas	33
4	4	jerkey	food	27
5	5	hotdogs	meat	33
6	6	milk	parishables	16
7	7	donuts	pastries	5

Stored Procedure

Getting Employee Total Hours

Description:

T

Create Statement:

```
create or replace function getEmployeeTotalHours(int, REFCURSOR) returns refcursor as
$$
declare

  user_pid int      := $1;
  resultset   REFCURSOR := $2;
begin
  open resultset for

          select p.pid, p.first_name, p.last_name,start_time, end_time, day_logged,
shift_number,(end_time - start_time )as total_hours_worked
          from logged_time lt
          LEFT JOIN people p on p.pid=lt.pid
     where  user_pid >=p.pid;
  return resultset;
end;
$$
language plpgsql;
```

Security

Store Manager Admin

CREATE ROLE manager_admin;
GRANT ALL ON ALL TABLES in schema public
TO manager_admin

Mechanic Time Log Users

CREATE ROLE mechanic_admin;
GRANT SELECT, UPDATE, INSERT on maintenance_logs, gas_pumps, gas_robots TO mechanic_admin;

Cashiers

CREATE ROLE cashier_employee;
GRANT SELECT, INSERT, UPDATE on logged_hours TO cashier_employee;

Attendants

CREATE ROLE attendant_employee;
GRANT SELECT, INSERT, UPDATE on logged_time TO attendant_employee;

Triggers

Description:

The purpose of this trigger is to update the total hours of each employee when a user gives the id from the people table

Create Statement

```
CREATE TRIGGER viewEmployeeHours

AFTER UPDATE ON logged_time

FOR EACH ROW EXECUTE PROCEDURE getEmployeeTotalHours ();
```

Reports

View Mechanics and Pumps Operated on

Description:

This report will show who and when a mechanic operated on a specific gas pump in the case of the manager needing to contact said mechanic.

Create Statement

```
select ml.maint_id, p.first_name, p.last_name, ml.mid as mechanic_id,
gp.last_operated_on as gas_pump_last_maintained_date, gr.last_operated_on as
gas_pump_last_maintained_date FROM maintenance_logs ml

INNER JOIN gas_pumps gp on gp.maint_id=ml.maint_id

INNER JOIN gas_robots gr on gr.maint_id=ml.maint_id

INNER JOIN mechanics m on m.mid=ml.mid

inner join people p on p.pid=m.pid
```

	maint_id integer	first_name character varying(50)	last_name character varying(50)	mechanic_id integer	gas_pump_last_maintained_date timestamp without time zone	gas_pump_last_maintained_date timestamp without time zone
1	1	Jane	Bower	5	2016-07-26 13:30:12	2016-01-25 17:12:29
2	2	Madeleine	Peake	8	2016-07-26 13:30:12	2016-01-25 17:12:29

Implementation Notes

In order for this database design to be implemented with as little error as possible, the following notes should be taken into consideration:

- Gas ID should not need to change unless there is a new type of gas added, in which case both the gas_type and product table would need to be updated.
- Employees who are assume more than one role should be added twice, with a different role for each entry.
-

Known Problems

Although this database design was created, reviewed and carefully inspected for errors, here is a list of the current known issues:

- The updating of one product/gas type will require the updating of their connecting tables
- When updating maintenance logs, readability can be difficult and may require another view creation.

Future Enhancements

If given more time, the following compile a list of possible/ suggested improvements to this database design

- Adding an employee hire and fire column
- Adding an employee payment preference
- Adding more detailed maintenance notes
- Active inventory quantity counter for items