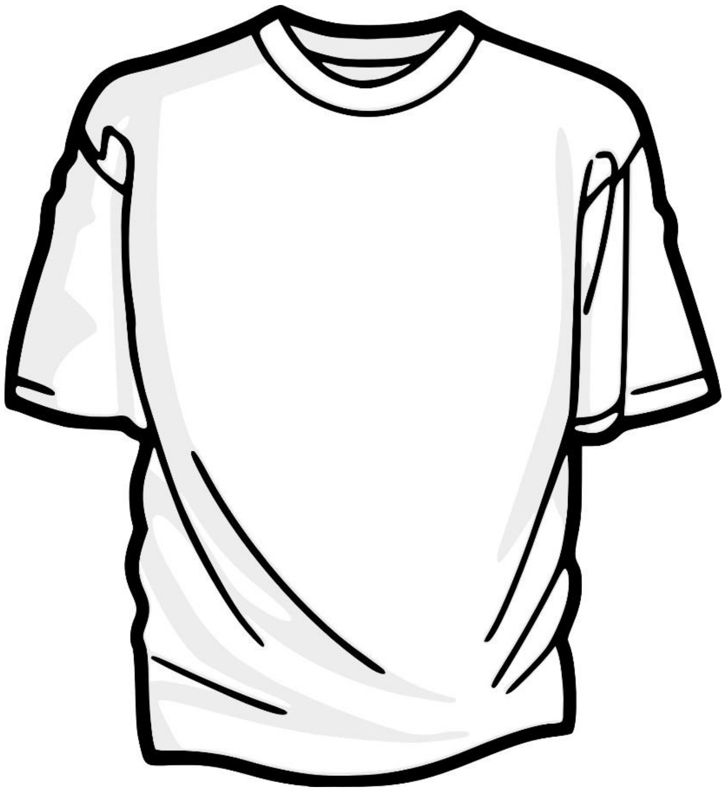
Big Clothing Store



Database Design Specs

April 20, 2016

Jacob Levinson

Table of Contents

Executive Summary

Overview

People do not realize how much work goes into running something as simple-sounding as a clothing store. Over the summer, I worked in a four-story clothing store in Times Square, New York and during this time I learned a lot of what goes into making a store like that run. For example, each floor has a General Manager that oversees everyone on that floor. Under them are ordinary managers that deal with managing the people working on the floor at the time they are on duty. Finally there are sales associates that do all the grunt work. There is also a stock team that works in the stock room to help bring clothes that are needed onto each floor.

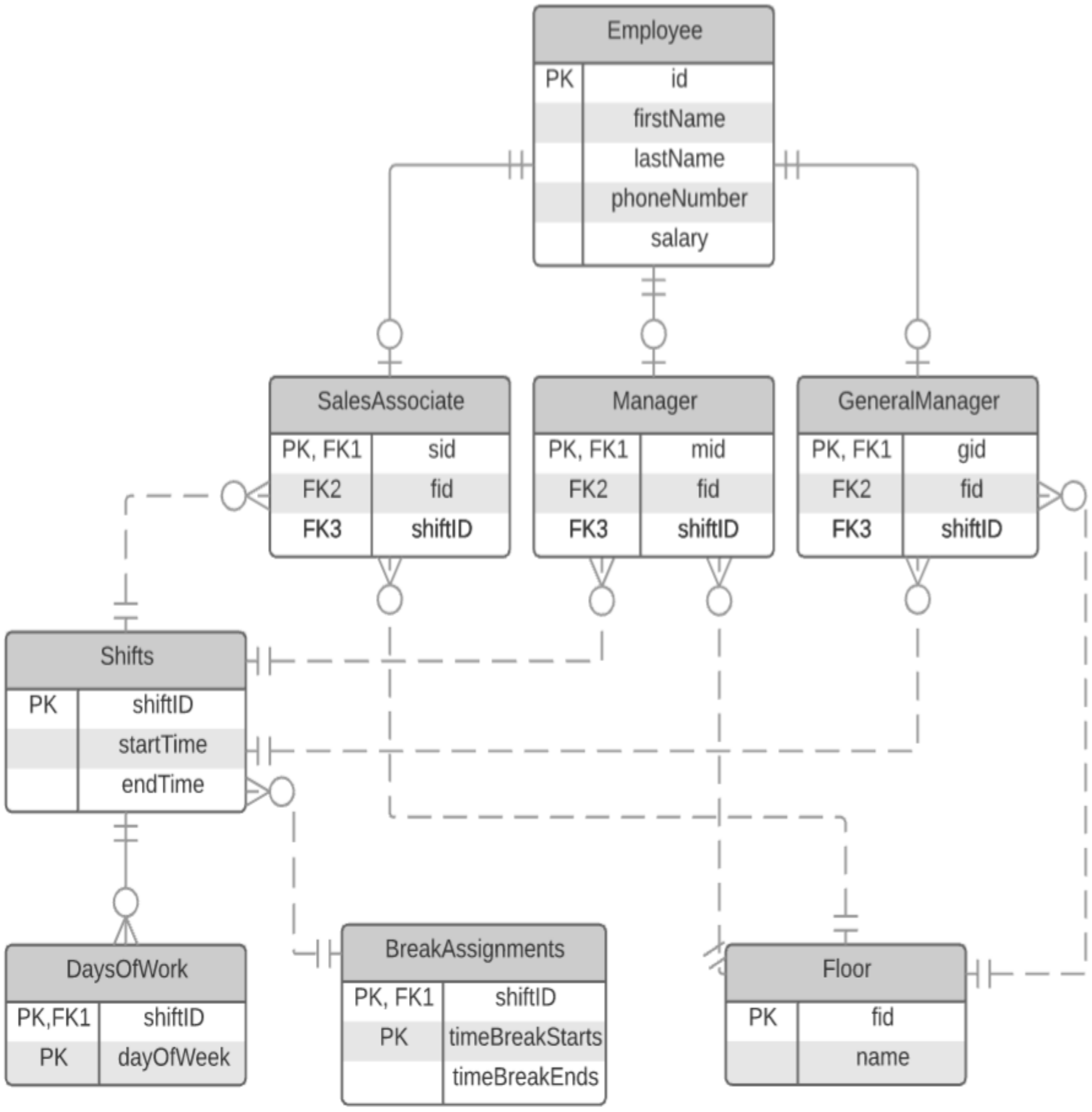
A store like this can be more efficiently ran in a number of different ways, however many of these are on a level that cannot be handled by programs. Some of these include better relationships between managers and associates as well as better ways to monitor how well the employees are working. For example, under some managers, I would get more work done because we had a mutual respect and friendship, while other managers were too focused on work which made me not want to be there. On the other hand, as an associate, I had better insight on the work my fellow associates were doing, therefore, I could see who was slacking and who was not. A change in training for managers could probably solve both of those problems. Another problem is wasted time for managers which is what this documents purpose is to solve.

Objectives

I do not know all of the details of what a manager does, however I do know that time is wasted when an employee asks a manager when they should go on break and the manager has to take time to figure out when people should go on break. The purpose of this database system outline is to find a way to save managers a decent chunk of time each shift. When you are managing a store as large as the one I worked at, you have to deal with a lot of employees. When you are dealing with dozens of employees a day, the small things begin to add up.

This document will provide an overview of the database which will serve as a guideline to General Managers on how to plan breaks for their employees before their shift rather than during. This will save the time of the managers and allow them to get other work done such as crunching numbers and helping customers and employees on the floor. This document will include a comprehensive E-R diagram, the tables and their dependencies, views, reports, stored procedures, triggers and more.

Entity Relationship Diagram



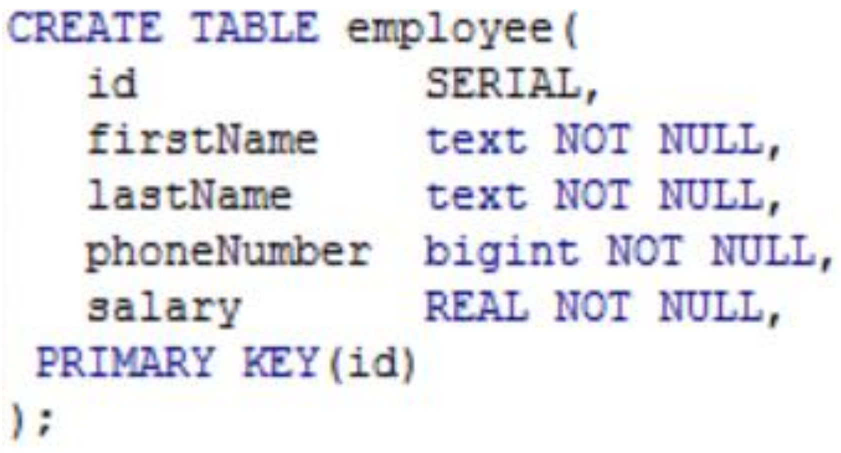
Tables

Employee

Purpose

The employee table stores first names, last names, phone numbers, and the salaries of each employee in the store. The table stores each employee with a unique id number that auto-increments as each new employee is made. The salaries are in terms of dollars per hour.

Create Statement

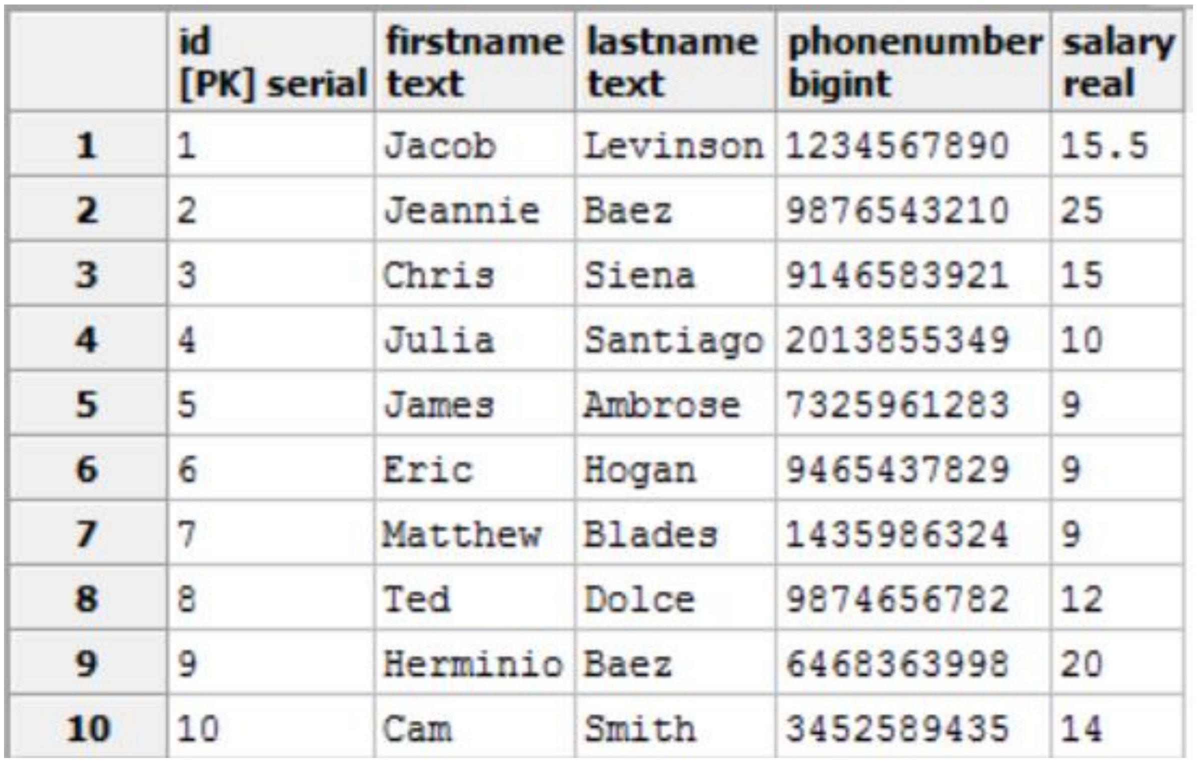


```
CREATE TABLE employee(
   id             SERIAL,
   firstName      text NOT NULL,
   lastName       text NOT NULL,
   phoneNumber    bigint NOT NULL,
   salary         REAL NOT NULL,
 PRIMARY KEY(id)
);
```

Functional Dependencies

id → firstName, lastName, phoneNumber, salary

Sample Data



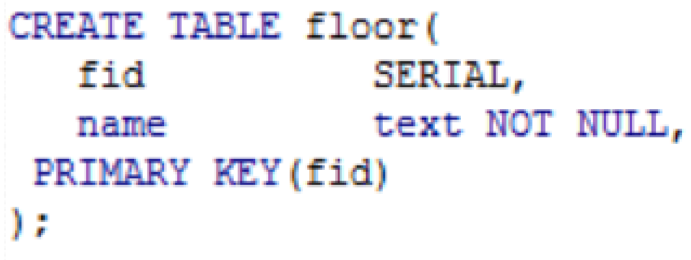
	id [PK] serial	firstname text	lastname text	phonenumber bigint	salary real
1	1	Jacob	Levinson	1234567890	15.5
2	2	Jeannie	Baez	9876543210	25
3	3	Chris	Siena	9146583921	15
4	4	Julia	Santiago	2013855349	10
5	5	James	Ambrose	7325961283	9
6	6	Eric	Hogan	9465437829	9
7	7	Matthew	Blades	1435986324	9
8	8	Ted	Dolce	9874656782	12
9	9	Herminio	Baez	6468363998	20
10	10	Cam	Smith	3452589435	14

Floor

Purpose

The floor table lists the floors in the store. Each floor has a serial ID that auto-increments as well as a text name.

Create Statement

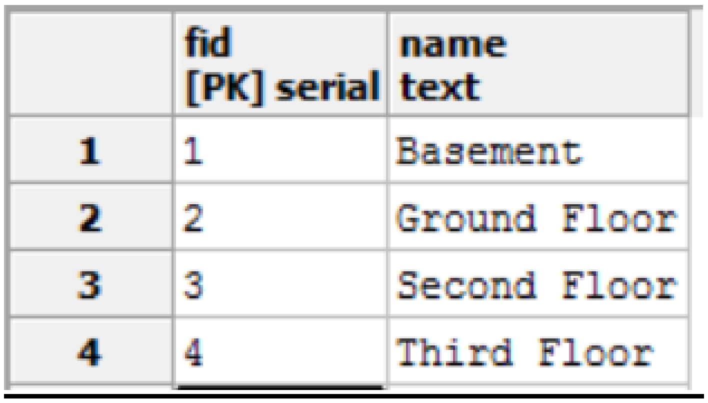


```
CREATE TABLE floor(
   fid            SERIAL,
   name           text NOT NULL,
 PRIMARY KEY(fid)
);
```

Functional Dependencies

fid → name

Sample Data



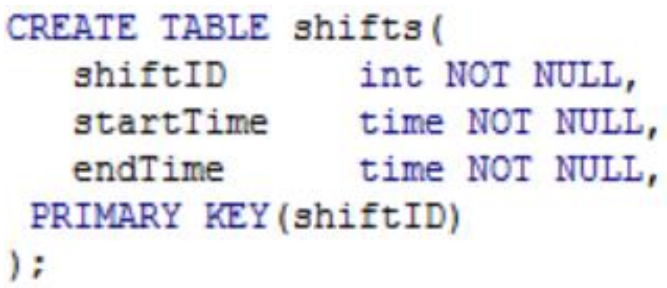
	fid [PK] serial	name text
1	1	Basement
2	2	Ground Floor
3	3	Second Floor
4	4	Third Floor

Shifts

Purpose

The shifts table lists the different shifts of work available for each day. Each shift has an ID number, as well as a starting and ending time.

Create Statement

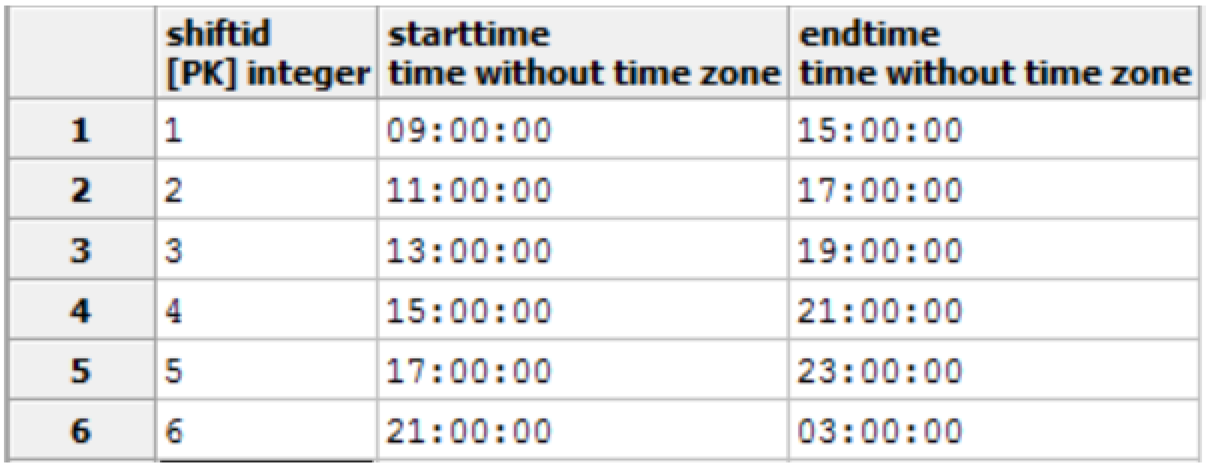


```
CREATE TABLE shifts(
   shiftID      int NOT NULL,
   startTime    time NOT NULL,
   endTime      time NOT NULL,
 PRIMARY KEY(shiftID)
);
```

Functional Dependencies

shiftID → startTime, endTime

Sample Data



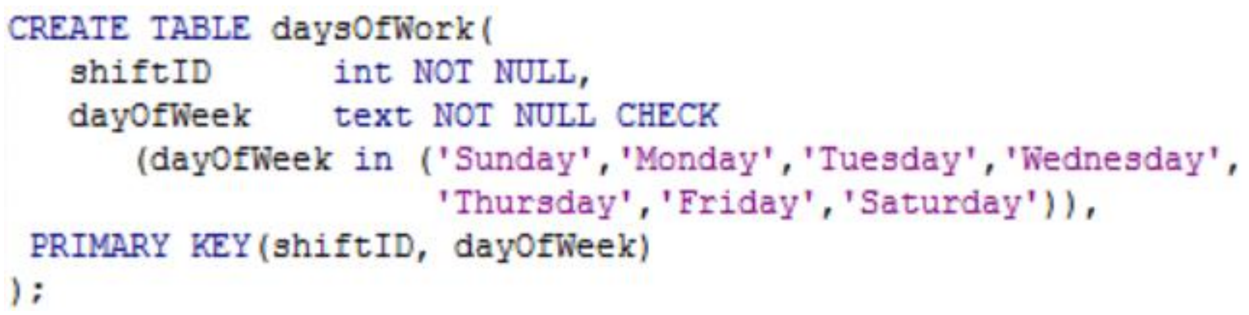
	shiftid [PK] integer	starttime time without time zone	endtime time without time zone
1	1	09:00:00	15:00:00
2	2	11:00:00	17:00:00
3	3	13:00:00	19:00:00
4	4	15:00:00	21:00:00
5	5	17:00:00	23:00:00
6	6	21:00:00	03:00:00

daysOfWork

Purpose

The daysOfWork table shows what shifts are available during what days. For example, weekends might have more shifts due to heavier shopping traffic on the weekends. In this case the shifts were kept the same for each day, however the managers and general managers would be able to add day-specific shifts whenever they like. The table also checks to make sure that the user is typing in the day of the week correctly.

Create Statement

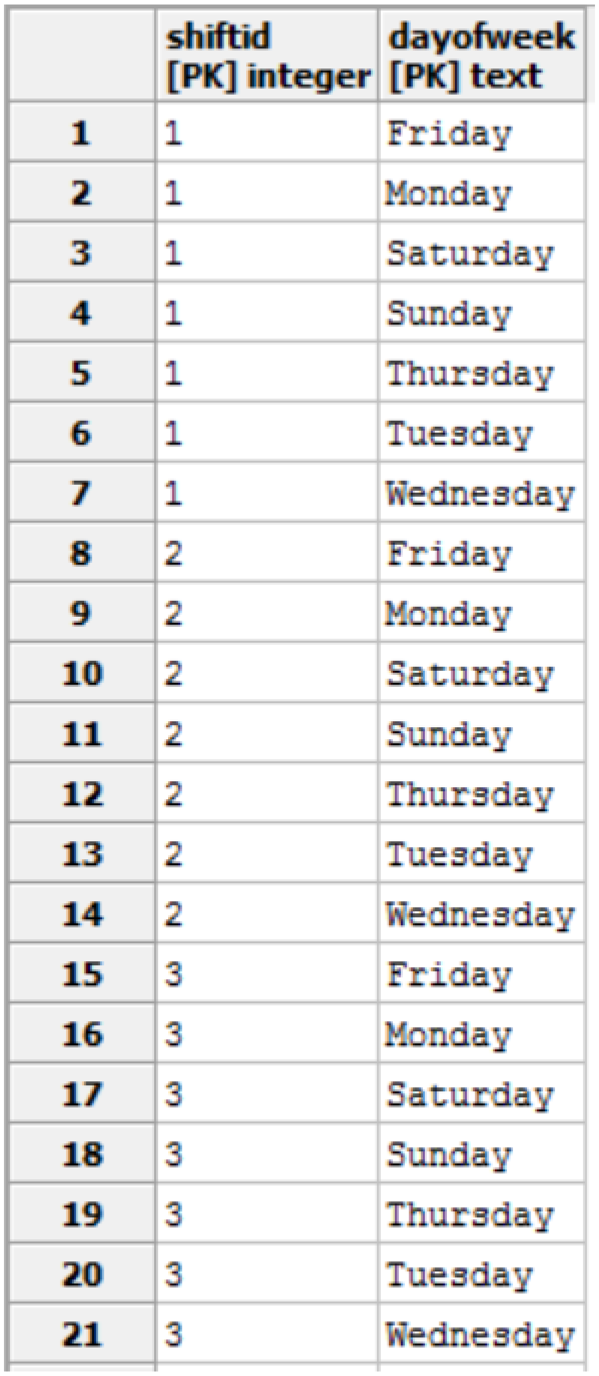
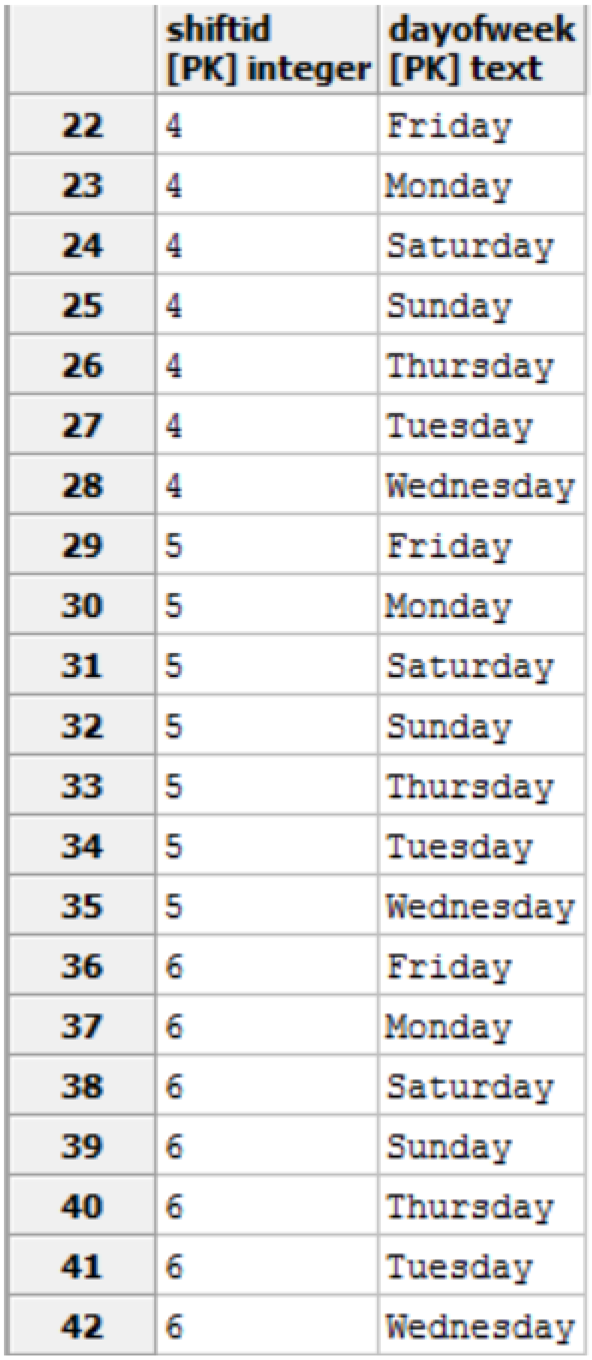


```
CREATE TABLE daysOfWork(
   shiftID      int NOT NULL,
   dayOfWeek    text NOT NULL CHECK
      (dayOfWeek in ('Sunday','Monday','Tuesday','Wednesday',
                     'Thursday','Friday','Saturday')),
 PRIMARY KEY(shiftID, dayOfWeek)
);
```

Functional Dependencies

shiftID, daysOfWeek →

Sample Data

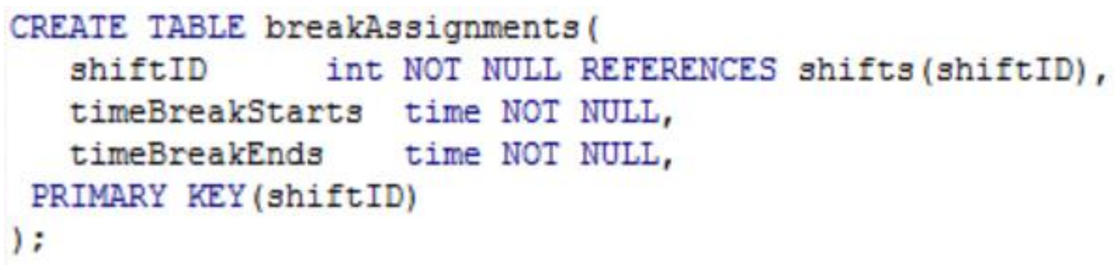
	shiftid [PK] integer	dayofweek [PK] text
1	1	Friday
2	1	Monday
3	1	Saturday
4	1	Sunday
5	1	Thursday
6	1	Tuesday
7	1	Wednesday
8	2	Friday
9	2	Monday
10	2	Saturday
11	2	Sunday
12	2	Thursday
13	2	Tuesday
14	2	Wednesday
15	3	Friday
16	3	Monday
17	3	Saturday
18	3	Sunday
19	3	Thursday
20	3	Tuesday
21	3	Wednesday
22	4	Friday
23	4	Monday
24	4	Saturday
25	4	Sunday
26	4	Thursday
27	4	Tuesday
28	4	Wednesday
29	5	Friday
30	5	Monday
31	5	Saturday
32	5	Sunday
33	5	Thursday
34	5	Tuesday
35	5	Wednesday
36	6	Friday
37	6	Monday
38	6	Saturday
39	6	Sunday
40	6	Thursday
41	6	Tuesday
42	6	Wednesday

BreakAssignments

Purpose

The breakAssignments table shows the employees when their breaks start and end. The table references the shiftID of the shift that the employee is working and then shows the starting and end times of breaks.

Create Statement

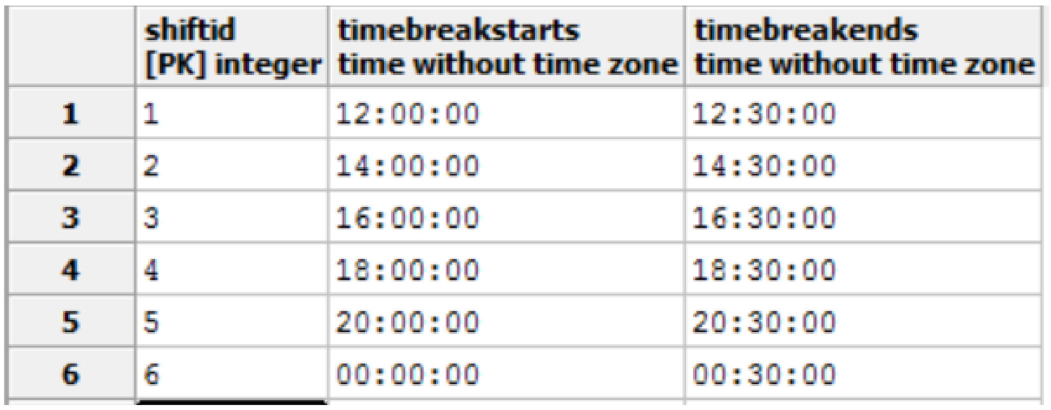


```
CREATE TABLE breakAssignments(
   shiftID      int NOT NULL REFERENCES shifts(shiftID),
   timeBreakStarts  time NOT NULL,
   timeBreakEnds    time NOT NULL,
 PRIMARY KEY(shiftID)
);
```

Functional Dependencies

shiftID → timeBreakStarts, timeBreakEnds

Sample Data



	shiftid [PK] integer	timebreakstarts time without time zone	timebreakends time without time zone
1	1	12:00:00	12:30:00
2	2	14:00:00	14:30:00
3	3	16:00:00	16:30:00
4	4	18:00:00	18:30:00
5	5	20:00:00	20:30:00
6	6	00:00:00	00:30:00

SalesAssociate

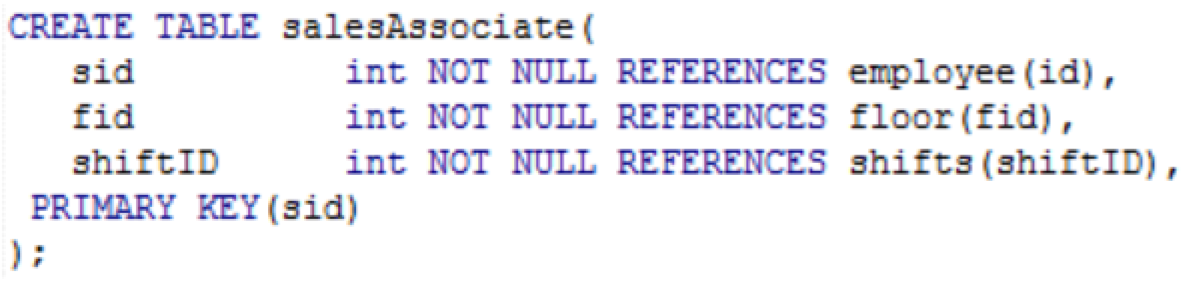
Purpose

This table shows which employees are sales associates and which are not.

Each sales associate has an ID, sid, which references the employee’s id.

They also are assigned a floor and a shift.

Create Statement

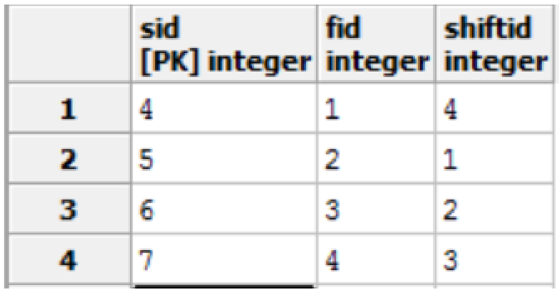


```
CREATE TABLE salesAssociate(
   sid          int NOT NULL REFERENCES employee(id),
   fid          int NOT NULL REFERENCES floor(fid),
   shiftID      int NOT NULL REFERENCES shifts(shiftID),
 PRIMARY KEY(sid)
);
```

Functional Dependencies

sid → fid, shiftID

Sample Data



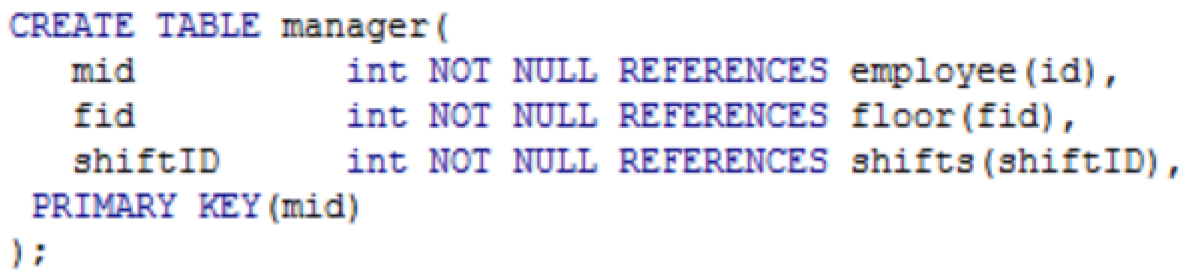
	sid [PK] integer	fid integer	shiftid integer
1	4	1	4
2	5	2	1
3	6	3	2
4	7	4	3

Manager

Purpose

This table shows which employees are managers and which are not. Each manager has an ID, mid, which references the employee’s id. They also are assigned a floor and a shift.

Create Statement

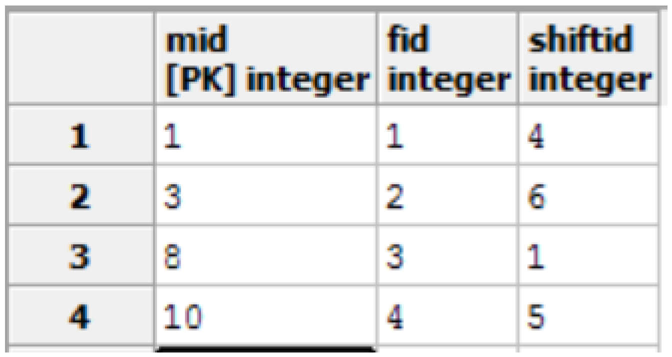


```
CREATE TABLE manager(
   mid          int NOT NULL REFERENCES employee(id),
   fid          int NOT NULL REFERENCES floor(fid),
   shiftID      int NOT NULL REFERENCES shifts(shiftID),
 PRIMARY KEY(mid)
);
```

Functional Dependencies

mid → fid, shiftID

Sample Data



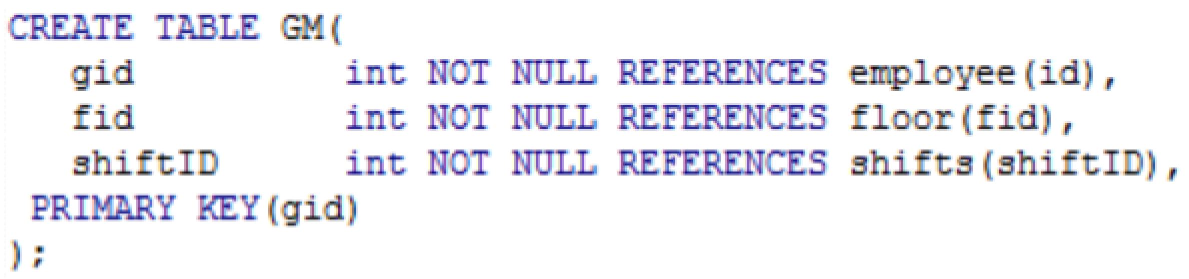
	mid [PK] integer	fid integer	shiftid integer
1	1	1	4
2	3	2	6
3	8	3	1
4	10	4	5

GM

Purpose

This table shows which employees are managers and which are not. Each manager has an ID, mid, which references the employee's id. They also are assigned a floor and a shift.

Create Statement

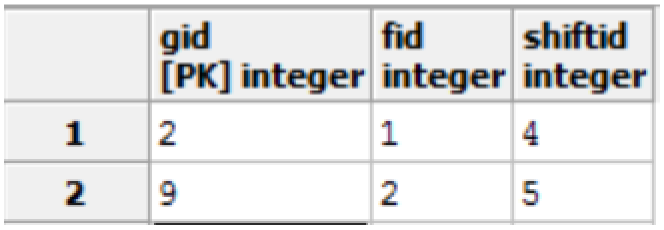


```
CREATE TABLE GM(
   gid          int NOT NULL REFERENCES employee(id),
   fid          int NOT NULL REFERENCES floor(fid),
   shiftID      int NOT NULL REFERENCES shifts(shiftID),
 PRIMARY KEY(gid)
);
```

Functional Dependencies

gid → fid, shiftID

Sample Data



	gid [PK] integer	fid integer	shiftid integer
1	2	1	4
2	9	2	5

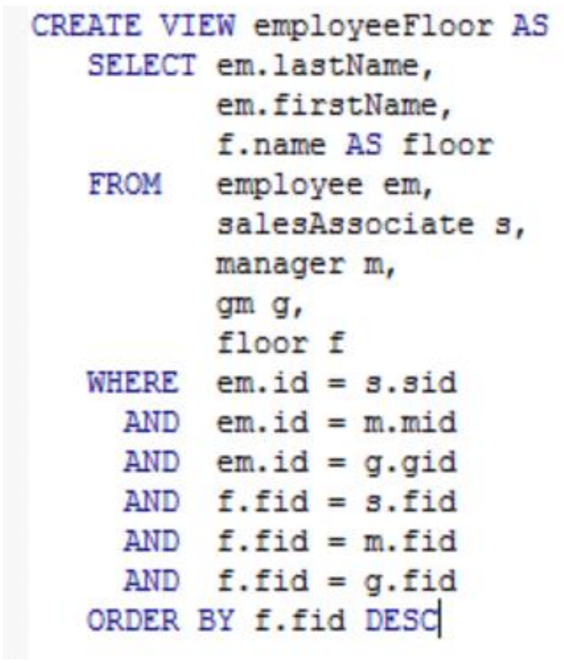
Views

EmployeeFloors

Purpose

It is important to for managers to know who is working on their floor. This allows managers to figure out who is working with them and at what times. Using this information, managers can know where to put people to work at certain times, as well as whether or not they need to call other employees to come in and work if they are low on staff.

Create Statement



```
CREATE VIEW employeeFloor AS
   SELECT em.lastName,
          em.firstName,
          f.name AS floor
   FROM   employee em,
          salesAssociate s,
          manager m,
          gm g,
          floor f
   WHERE  em.id = s.sid
     AND  em.id = m.mid
     AND  em.id = g.gid
     AND  f.fid = s.fid
     AND  f.fid = m.fid
     AND  f.fid = g.fid
   ORDER BY f.fid DESC
```

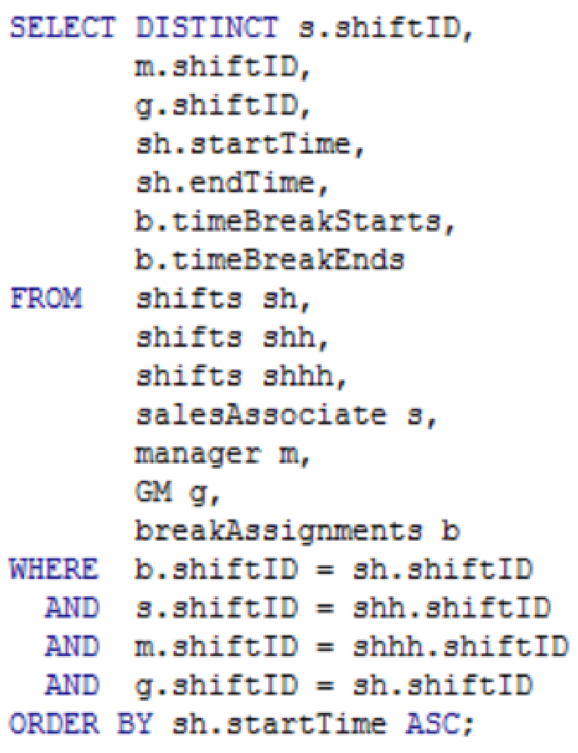
Reports

Checking when break times are for different shifts

Purpose

During work, a manager takes time out to determine when each employee, including themselves, should go on a break. Rather than wasting a few minutes every time an employee needs to go on break, there are now set break times. A manager can check the times and know exactly when to send each employee on break, saving a lot of time in the long run.

Create Statement

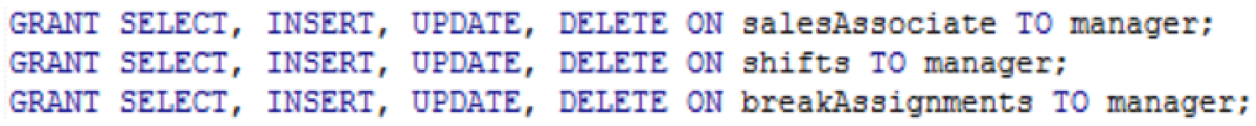


```
SELECT DISTINCT s.shiftID,
       m.shiftID,
       g.shiftID,
       sh.startTime,
       sh.endTime,
       b.timeBreakStarts,
       b.timeBreakEnds
FROM   shifts sh,
       shifts shh,
       shifts shhh,
       salesAssociate s,
       manager m,
       GM g,
       breakAssignments b
WHERE  b.shiftID = sh.shiftID
  AND  s.shiftID = shh.shiftID
  AND  m.shiftID = shhh.shiftID
  AND  g.shiftID = sh.shiftID
ORDER BY sh.startTime ASC;
```

Security

Manager

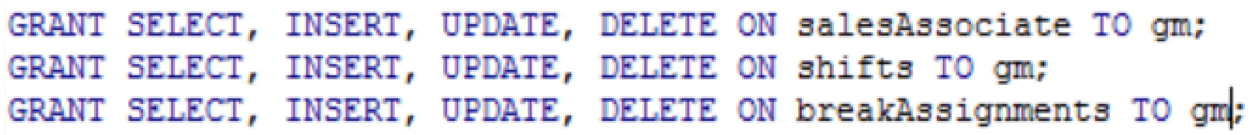
Managers are in charge of the sales associates on their floors. They need to be able to see who is working that day, as well as be able to add in employees that are picking up a shift. They need to see when people are working and when their breaks are.



```
GRANT SELECT, INSERT, UPDATE, DELETE ON salesAssociate TO manager;
GRANT SELECT, INSERT, UPDATE, DELETE ON shifts TO manager;
GRANT SELECT, INSERT, UPDATE, DELETE ON breakAssignments TO manager;
```

General Manager

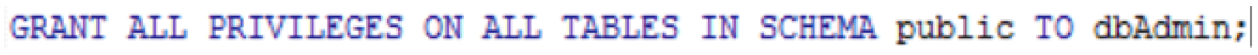
GMs are the managers of everyone on their floors. They need to be able to see who is working that day, as well as be able to add in employees that are picking up a shift. They need to see when people are working and when their breaks are.



```
GRANT SELECT, INSERT, UPDATE, DELETE ON salesAssociate TO gm;
GRANT SELECT, INSERT, UPDATE, DELETE ON shifts TO gm;
GRANT SELECT, INSERT, UPDATE, DELETE ON breakAssignments TO gm;
```

Database Admin

The database administrator is in charge of it all. This is either the store own or someone the store owner hired to oversee the DB.



```
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO dbAdmin;
```

Implementation Notes

Managers and GMs should both be taught how to add in employees, shifts, and how to assign shifts to employees. GMs should be taught how to add in break assignments depending on what the shifts are. Also GMs should add every possible shift into the shifts table.

Known Problems

Some of the known problems include:

- There are only 6 hour shift periods as well as only breaks for those set shifts. Sometimes people work more or less than 6 hours however this can be fixed by adding in all of the different combinations for shifts and their corresponding break times.
- There is no way to choose different shifts for different days. This can become tedious as you would have to change the shift that each employee has every day.

Future Enhancements

- Allowing for employees to work different shifts for different days. This would also allow for an easy way to determine how much each employee would get paid in a week.