The Gray Inn

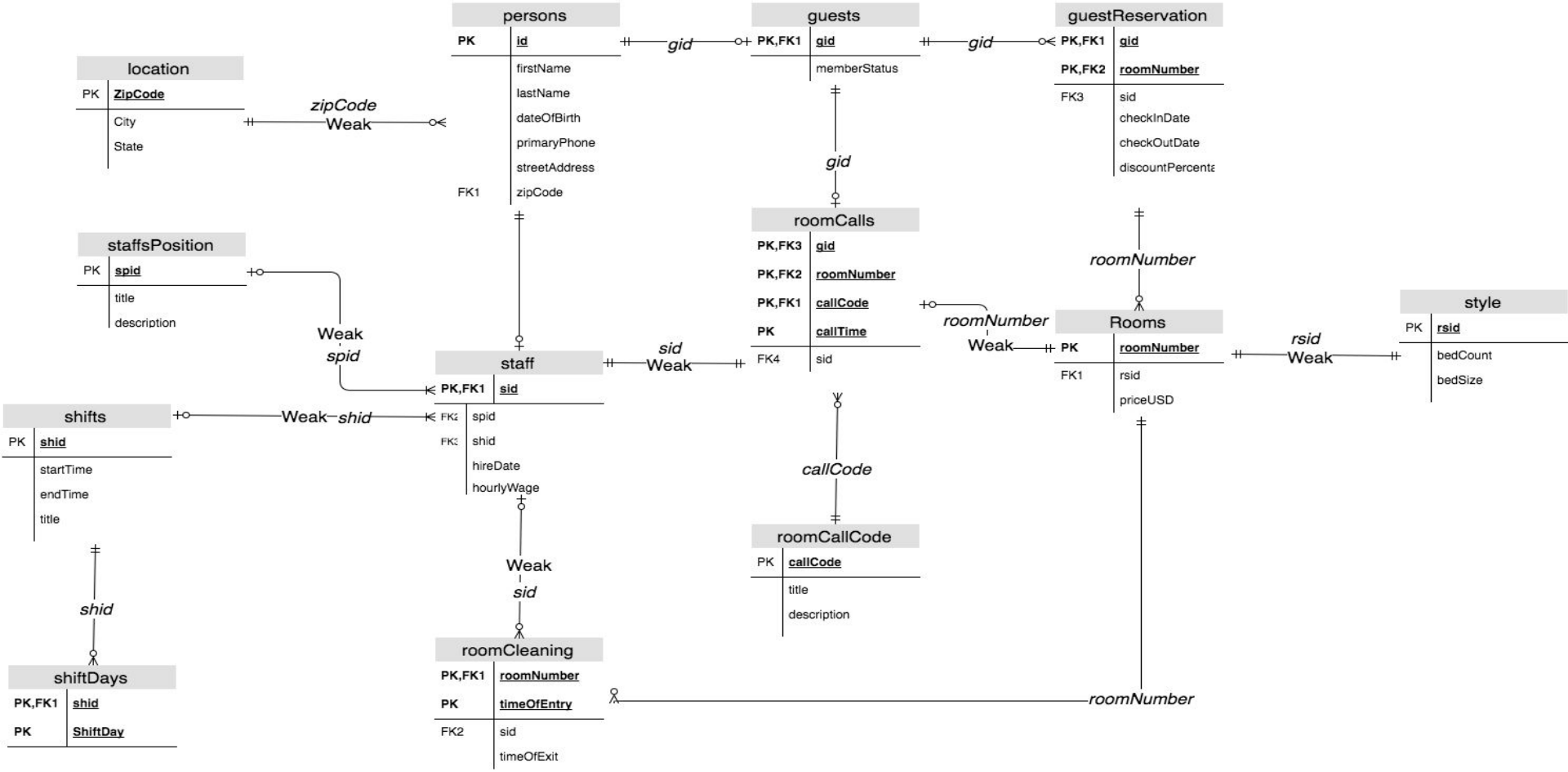
By: Dan Njoku

Table of Content

Executive Summary & Overview

The Gray Inn is a chain of hotels located all throughout the continental United States. This database in particular concentrates solely of just one hotel, however, an expanded database may be soon to come. The hospitality industry is a multi-billion dollar business that grossed an average of $175 billion in the United States in the year 2015. With the average hotel room costing around $120 dollars per night, it is safe to say that the owning a hotel can prove to be a is a fairly lucrative business venture. In the year 2015 the average occupancy rate of hotels across the United States reached a staggering 65.6 percent, an increase of 11 percent dated 6 years prior, during recession of 2009. The purpose of this database is to track the daily operations within the hotel. The data can be used by administration, through the use of queries in order to track the day to day operations that are taking place within the hotel. Through managing The Gray Inn, this database implementation will be able to track the hotels staff and guests in a more accurate and concise manner.

The Gray Inn - ER Diagram



Persons Table

Due to the fact that staff members may also be guests in the hotel, there is a separate table titled persons, used to store information.

```
DROP TABLE IF EXISTS persons
CREATE TABLE IF NOT EXISTS persons (
    pid             INTEGER     NOT NULL,
    firstName       VARCHAR(50) NOT NULL,
    lastName        VARCHAR(50) NOT NULL,
    dateOfBirth     DATE        NOT NULL,
    phonePrimary    CHAR(15)    NOT NULL,
    streetAddress   VARCHAR(50) NOT NULL,
    zipCode         INTEGER     NOT NULL,
PRIMARY KEY(pid)
FOREIGN KEY(zipCode) REFERENCES location(zipCode)
);
```

pid	firstName	lastName	dateOfBirth	primaryPhone	streetAddress	zipCode
p001	Dan	Rogers	11/4/1949	(690)983-6071	741 James Lane	34281
p002	Tory	Kelly	8/30/1955	(405)819-6376	211 Elm Street	34472
p003	Jim	Beam	3/19/1958	(111)495-8787	434 Jackson Road	55578
p004	Jack	Daniels	12/7/1959	(219)239-7966	324 Janos Lane	88020
p005	Mike	Spencer	11/19/1963	(689)369-1282	43 Taylor Ave	90101
p006	Ruben	Guiterez	7/30/1964	(689)369.1282	99 Washington Street	98210
p007	Gabe	Avalos	7/19/1981	(516)990-0999	93 Oak Street	88525
p008	Carter	Francis	12/2/1987	(322)234-2343	82 Woodfield Road	28602
p009	Jennifer	Wright	6/30/1988	(939)212-4322	34 Jason Street	12601
p010	James	Jones	1/15/1993	(323)231-2131	383 Dan Rode	12586
p011	Alfred	Weiss	3/22/1994	(423)321-3491	33 Wadley Way	12590
p012	Joe	Delot	11/11/1998	(734)212-2342	342 Mark Drive	11552
p013	Albert	Einstein	11/22/1977	(212)213-8585	92 Oak Road	62241

Functional Dependencies

Pid → firstName, lastName, dateOfBirth, primaryPhone, streetAddress, zipCode

Location The locations table stores the city and state for its given zip code.

```
DROP TABLE IF EXISTS location
CREATE TABLE IF NOT EXISTS location (
    zipCode INTEGER     NOT NULL UNIQUE,
    city    VARCHAR(50) NOT NULL,
    state   TEXT(2)     NOT NULL,
PRIMARY KEY(zipCode)
);
```

Function Dependencies

zipCode → city, state

zipCode	city	state
34281	Bradenton	FL
34472	Ocala	FL
35201	Birmingam	AL
55578	Maple Palm	MN
88020	Animas	NM
98101	Seatle	WA
90101	Los Angeles	CA
90210	Bell	CA
88525	El Pasp	TX
28602	Sparta	NY
12601	Poughkeepsie	NY
12586	Walden	NY
12590	Wappengers	NY
11552	West Hemp.	NY
62241	Ellis Grove	NY

Guests

The purpose of the guest table is to identify each guest based on their gid/pid, and also to specify their member status

```
DROP TABLE IF EXISTS guests
CREATE TABLE IF NOT EXISTS guests (
    gid             INTEGER    NOT NULL,
    memberStatus    TEXT       NOT NULL,
PRIMARY KEY(gid) REFERENCES persons(pid)
);
```

gid	memberStatus
p001	Bronze
p002	Gold
p003	Diamond
p004	Bronze
p005	Silver
p006	Platinum
p007	Bronze
p008	Silver

Function Dependencies

gid → memberStatus

Room Calls

The objective of this table is to track each guest's room calls

```
DROP TABLE IF EXISTS roomCall
CREATE TABLE IF NOT EXISTS roomCall (
    gid         INTEGER     NOT NULL,
    roomNumber  VARCHAR(5)  NOT NULL,
    callCode    INTEGER     NOT NULL,
    callTime    TIMESTAMP   NOT NULL,
    sid     INTEGER     NOT NULL,
PRIMARY KEY(callTime),
PRIMARY KEY(gid)        REFERENCES guests(gid),
FOREIGN KEY(gid)        REFERENCES guests(gid),
PRIMARY KEY(roomNumber) REFERENCES rooms(roomNumber),
FOREIGN KEY(roomNumber) REFERENCES rooms(roomNumber),
PRIMARY KEY(callCode)   REFERENCES roomCallCodes(callCode),
FOREIGN KEY(callCode)   REFERENCES roomCallCodes(callCode),
PRIMARY KEY(callTime)   REFERENCES guests(gid),
FOREIGN KEY(callTime)   REFERENCES guests(gid),
FOREIGN KEY(sid)        REFERENCES staff(sid)
```

gid	roomNumber	callCode	callTime	sid
p001	101	2	5/31/2015 5:36;32 AM	p009
p002	102	2	6/1/2015 4:34;33PM	p009
p003	103	2	5/31/2015 9:25;23 PM	p013
p004	104	1	6/1/2015 3:25;40 PM	p013
p005	105	2	6/2/2015 8:25;34 PM	p013
p006	106	4	6/3/2015 8;25;23 PM	p013
p007	201	4	6/4/2015 4;25;12 PM	p013
p008	202	3	6/5/2015 1:45;35 PM	p013
p001	101	3	6/6/2015 3:56;25 PM	p013
p002	102	3	6/7/2015 3:30;25 PM	p013
p003	103	2	6/8/2015 8:25;30 AM	p009

Function Dependencies

gid, roomNumber, callCode, callTime → sid

Room Call Codes

This table gives some rudimentary examples to what some call codes can consist of.

```
DROP TABLE IF EXISTS roomCallCode
CREATE TABLE IF NOT EXISTS roomCallCode (
    callCode        integer     NOT NULL,
    title           VARCHAR(50) NOT NULL,
    description     TEXT        NOT NULL,
PRIMARY KEY(callCode)
);
```

callCode	title	description
1	Maintenance	Faulty Appliance
2	Kitchen	Order Food
3	Room Service	Clean Room
4	Help Desk	Needs Info

Function Dependencies

callCode → title, description

Staff

This table provides us with the basic information needed in regard to the hotels employees and their job entailment.

```
DROP TABLE IF EXISTS staff
CREATE TABLE IF NOT EXISTS staff (
    sid             INTEGER     NOT NULL,
    spid            INTEGER     NOT NULL,
    shid            INTEGER     NOT NULL,
    hireDate        DATE        NOT NULL,
    hourlyWageUSD   DECIMAL       NOT NULL,
PRIMARY KEY(sid) REFERENCES persons(pid),
FOREIGN KEY(sid) REFERENCES persons(pid),
FOREIGN KEY(spid) REFERENCES staffsPosition(pid),
FOREIGN KEY(shid) REFERENCES shifts(shid)
);
```

sid	spid	shid	hireDate	hourlyWageUSDS
p009	1	0	5/10/2015	15.1
p010	2	1	5/15/2015	10.75
p011	3	0	5/14/2015	11.25
p012	3	4	5/15/2015	9.75
p013	1	4	5/12/2015	16.25

Function Dependencies

sid → spid, shid, hireDate, hourlyWage

Shifts This table gives some rudimentary shift hours that the staff could possible work.

```
DROP TABLE IF EXISTS shifts
CREATE TABLE IF NOT EXISTS shifts (
    shid        INTEGER     NOT NULL,
    startTime   TIME        NOT NULL,
    endTime     TIME        NOT NULL,
    title       TEXT,
PRIMARY KEY(shid)
);
```

shid	startTime	endTime	title
0	12:00:00 AM	12:00:00 PM	Late Shift
1	12:00:00 PM	6:00:00 PM	Afternoon Shift
3	6:00:00 PM	12:00:00 AM	Nighe Shift
4	12:00:00 PM	12:00:00 AM	Early Shift

Function Dependencies

shid → startTime, endTime, title

Shift Days

```
DROP TABLE IF EXISTS shiftDays
CREATE TABLE IF NOT EXISTS shiftDays (
    shid        INTEGER     NOT NULL,
    shiftDay    TEXT        NOT NULL    CHECK,
                (shiftDay IN ('Sunday', 'Monday','Tuesday', 'Wednesday', 'Thursday','Friday','Saturday')),
PRIMARY KEY(shid, shiftDay),
FOREIGN KEY(shid) REFERENCES shifts(shid)
);
```

Function Dependencies

shid, shiftDay →

shid	shiftDay
0	Monday
0	Tuesday
0	Wednesday
0	Thursday
0	Friday
1	Saturday
1	Sunday
2	Monday
2	Tuesday
2	Thursday
2	Friday

Staff's Position Breaks down the staffs' positions into three main categories

```
DROP TABLE IF EXISTS staffsPosition
CREATE TABLE IF NOT EXISTS staffsPosition (
    spid         INTEGER     NOT NULL,
    title        INTEGER     NOT NULL,
    decription   TEXT,
PRIMARY KEY(spid)
);
```

spid	title	Decription
1	Administrator	
2	Kitchen	
3	HouseKeeping	

Function Dependencies

spid → title, description

Rooms

```
DROP TABLE IF EXISTS rooms
CREATE TABLE IF NOT EXISTS rooms (
    roomNumber  VARCHAR(5)  NOT NULL,
    rsid        INTEGER     NOT NULL,
    priceUSD    DECIMAL     NOT NULL,
PRIMARY KEY(roomNumber),
FOREIGN KEY(rsid) REFERENCES style(rsid)
);
```

roomNumber	rsid	priceUSD
101	8	250
102	7	350.59
103	6	199.89
104	5	210.01
105	4	300.99
106	3	150.45
201	3	150.45
202	3	150.45
203	1	210.69
204	1	210.69
205	2	120.49
206	8	250

Function Dependencies

roomNumber→ rsid, priceUSD

Style

This table represents the style of each individual room, in regard to bed count and the sizes of the bed(s).

```
DROP TABLE IF EXISTS style
CREATE TABLE IF NOT EXISTS style (
    rsid        INTEGER     NOT NULL,
    bedCount    CHAR(1)     NOT NULL,
    bedSize     TEXT        NOT NULL,
PRIMARY KEY(roomNumber)
);
```

rsid	bedCount	bedSize
1	1	King
2	1	Queen
3	1	Full
4	2	Queen
5	2	Full
6	2	Twin
7	2	King
8	3	Twin

Function Dependencies

rsid → bedCount, bedSize

Guest Reservation

This tables tracks the basic information of each guest through the duration of their stay.

```
DROP TABLE IF EXISTS guestReservation
CREATE TABLE IF NOT EXISTS guestReservation (
    gid                 INTEGER     NOT NULL,
    roomNumber          VARCHAR(5)  NOT NULL,
    sid                 INTEGER     NOT NULL,
    checkInDate         TIMESTAMP   NOT NULL,
    checkOutDate        TIMESTAMP   NOT NULL,
    discountPercentage  DECIMAL     NOT NULL,
PRIMARY KEY(gid) REFERENCES guest(gid),
FOREIGN KEY(gid) REFERENCES persons(pid),
PRIMARY KEY(roomNumber) REFERENCES rooms(roomNumber),
FOREIGN KEY(roomNumber) REFERENCES rooms(roomNumber),
PRIMARY KEY(sid) REFERENCES staff(sid),
);
```

gid	roomNumber	sid	checkInDate	CheckOutDate	DiscountPercentage
p001	101	p009	5/31/2015	6/4/2015	0
p002	102	p009	6/1/2015	6/3/2015	10
p003	103	p009	5/31/2015	6/2/2015	0
p004	104	p009	6/1/2015	6/10/2015	15
p005	105	p013	6/1/2015	6/3/2015	0
p006	106	p013	6/3/2015	6/4/2015	0
p007	201	p013	6/4/2015	6/7/2015	0
p008	202	p013	6/1/2015	6/3/2015	20

Function Dependencies

gid, roomNumber → sid, checkInDate, checkOutDate, discountPercentage

Room Cleaning

Specifies when each hotel room was last cleaned and which staff member cleaned it

```
DROP TABLE IF EXISTS roomCleaning
CREATE TABLE IF NOT EXISTS roomCleaning (
    roomNumber  INTEGER     NOT NULL,
    timeOfEntry TIMESTAMP   NOT NULL,
    sid         INTEGER     NOT NULL,
    timeOfExit  TIMESTAMP   NOT NULL,
PRIMARY KEY(roomNumber) REFERENCES rooms(roomNumber),
FOREIGN KEY(roomNumber) REFERENCES rooms(roomNumber),
PRIMARY KEY(roomNumber) REFERENCES rooms(roomNumber),
FOREIGN KEY(sid) REFERENCES staff(sid)
);
```

roomNumber	timeOfEntry	sid	timeOfExit
101	5/31/2015 5:39;32 AM	p011	5/31/2015 6:36;32 PM
102	6/1/2015 4:39;33PM	p011	6/1/2015 4:54;33 PM
103	5/31/2015 8:45;23 PM	p011	5/31/2015 9:00;23 PM
104	6/1/2015 3:00;40 PM	p011	6/1/2015 3:20;40 PM
105	6/2/2015 8:30;34 PM	p012	6/2/2015 8455;34 PM
106	6/3/2015 8;40;23 PM	p011	6/3/2015 8;55;23 PM
201	6/4/2015 4;50;12 PM	p012	6/4/2015 5:30;12 PM
202	6/5/2015 2:45;35 PM	p012	6/5/2015 3:45;35 PM

1 of 10 items

Function Dependencies

roomNumber, timeOfEntry → sid, timeOfExit

Views

GuestRooms

- The objective of this view is to track what guests have checked out of the hotel.

Create Statement

```
CREATE VIEW guestRooms AS
SELECT per.firstName,
       per.lastName,
FROM   persons per,
       guests  g,
       guestReservation gr,
       rooms r
WHERE    per.pid = g.gid
AND      gr.gid = g.gid
AND      gr.checkOutDate IS NOT NULL
ORDER BY per.lastname ASC;
```

RoomAvailability

- This view provides the hotel with information related to what rooms are currently available

Create statement

```
CREATE VIEW roomAvailability AS
SELECT rooms.roomNumber AS Room,
FROM rooms,
WHERE rooms.roomNumber NOT IN (
                              SELECT roomNumber
                              FROM   guestReservation
                              WHERE  checkOutDate IS NULL)
ORDER BY rooms.roomNumber ASC;
```

Reports

The Average Amount of time it takes to clean a room

- The objective of this query is to track how long it takes for each employee to clean each individual room

Query

```
SELECT staff.sid As Cleaner
    avg(rc.timeOfEntry - rc.timeOfExit) AS AVG Cleaning Time
FROM    staff s
        roomCleaning rc
WHERE rc.timeOfExit IS NOT NULL
AND     s.sid = rc.sid;
GROUP BY s.sid;
```

Average amount of time a guest stays in the hotel

- The objective of the following query is to track the amount of time the average quest stays at the Gray INN

Query

```
SELECT g.gid as GuestStay
    avg(gr.checkInDate - gr.checkOutDate)  AS Average Stay
FROM    guestReservation gr,
        guest g,
        rooms r
WHERE   gr.checkOutDate IS NOT NULL
AND     r.roomNumber = gr.roomNumber
GROUP BY g.gid;
```

Stored Procedures

addNewReservation

```
CREATE OR REPLACE FUNCTION insertGuestReservation ()
RETURNS tigger AS
$$
    BEGIN
        IF NEW.roomNumber IS NULL THEN
            RAISE EXPECTATION 'INVALID ROOM NUMBER!';
        END IF;
        IF NEW.checkInTime IS NULL THEN
            RAISE EXCPECTION 'This room is currently available';
        END IF;
        IF old.checkInTime IS NOT NULL THEN
            RAISE EXCPECTION 'This room is already occupied';
        END IF;
        INSERT INTO guestReservation (roomNumber, checkInTime)
                            VALUES    (NEW.roomNumber, 'now');
        Return NEW
    END;
$$ LANGUAGE pipgsql;
```

guestCheckOut

```
CREATE OR REPLACE FUNCTION guestCheckOut
RETURNS trigger AS
$$
    DECLARE
    ID INTEGER
    checkInTime TIMESTAMP
BEGIN
    IF NEW.checkOutTime IS NOT NULL THEN
    RAISE EXPCEPTION ('BY PROCEEDING YOU WILL VOID YOUR RESERVATION!')
END IF;

pid = NEW.gid
checkInTime = NEW.checkInTime

    Update guestReservation
    SET checkOutTime = 'now'
    WHERE pid = gid
        AND checkInTime = checkInTime
        AND checkOutTime IS NULL
    RETURN NEW
END;
$$ LANGUAGE pipgsql;
```

Triggers

AddRoomtoCleaningJob

- The purpose of this trigger in this scenario is to add a room to be cleaned once the guest has checked out of their room. Rooms may still be clean when occupied by the guest

Query

```
CREATE TRIGGER addRoomToRoomCleaning
AFTER UPDATE ON guestReservation
FOR EACH ROW EXECUTE PROCEDURE  insertRoomCleaning();
```

Security

There are three primary users within the hotel. For each role the specified user is allowed certain permissions are only given to certain staff members as shown.

Guests

```
GRANT INSERT ON roomCalls TO guests;
```

Housekeeping

```
GRANT SELECT, UPDATE ON roomCleaning TO houseKeeping;
GRANT SELECT ON rooms TO houseKeeping;
```

Administration

```
GRANT SELECT ON location TO hospitalAdministrator;
GRANT SELECT, INSERT, UPDATE ON persons TO Administration;
GRANT SELECT, INSERT, UPDATE ON guests TO Administration;
GRANT SELECT, INSERT, UPDATE ON staff TO Administration;
GRANT SELECT, INSERT, UPDATE, DELETE ON staffPosition TO Administration;
GRANT SELECT, INSERT, UPDATE, DELETE ON shifts TO Administration;
GRANT SELECT, INSERT, UPDATE, DELETE ON shiftDays TO Administration;
GRANT SELECT, INSERT, UPDATE ON patientVisits TO Administration;
GRANT SELECT, INSERT, UPDATE ON bedAssignments TO Administration;
GRANT SELECT, UPDATE ON roomCalls TO Administration;
GRANT SELECT, INSERT, UPDATE, DELETE ON callCodes TO Administration;
GRANT SELECT, UPDATE ON roomCleaning TO Administration;
GRANT SELECT, UPDATE ON rooms TO Administration;
GRANT SELECT, UPDATE ON style TO Administration;
```

Notes - Issues - Future Enhancements

Some future enhancements that will prove to be beneficial to the day to day functionality of the hospital would be to allowing a more diverse staff selection, giving staff members more roles besides housekeeping, administration, and working in the kitchen. Also, there should be a lot more flexibility with the amount of shifts, in regard to the shift hours and shift days. Through the expansion of the hotel database we can potentially add more space for private events in order to generate more revenue, and as well as increase the total number of rooms available.

Perhaps if I implemented more test data, similar to that of a real hotel, I would have been able to create more advanced queries. For future implementation I would like to specify different administrative roles, as well as move into the daily operations of the kitchen and housekeeping staff.