The Gray Inn

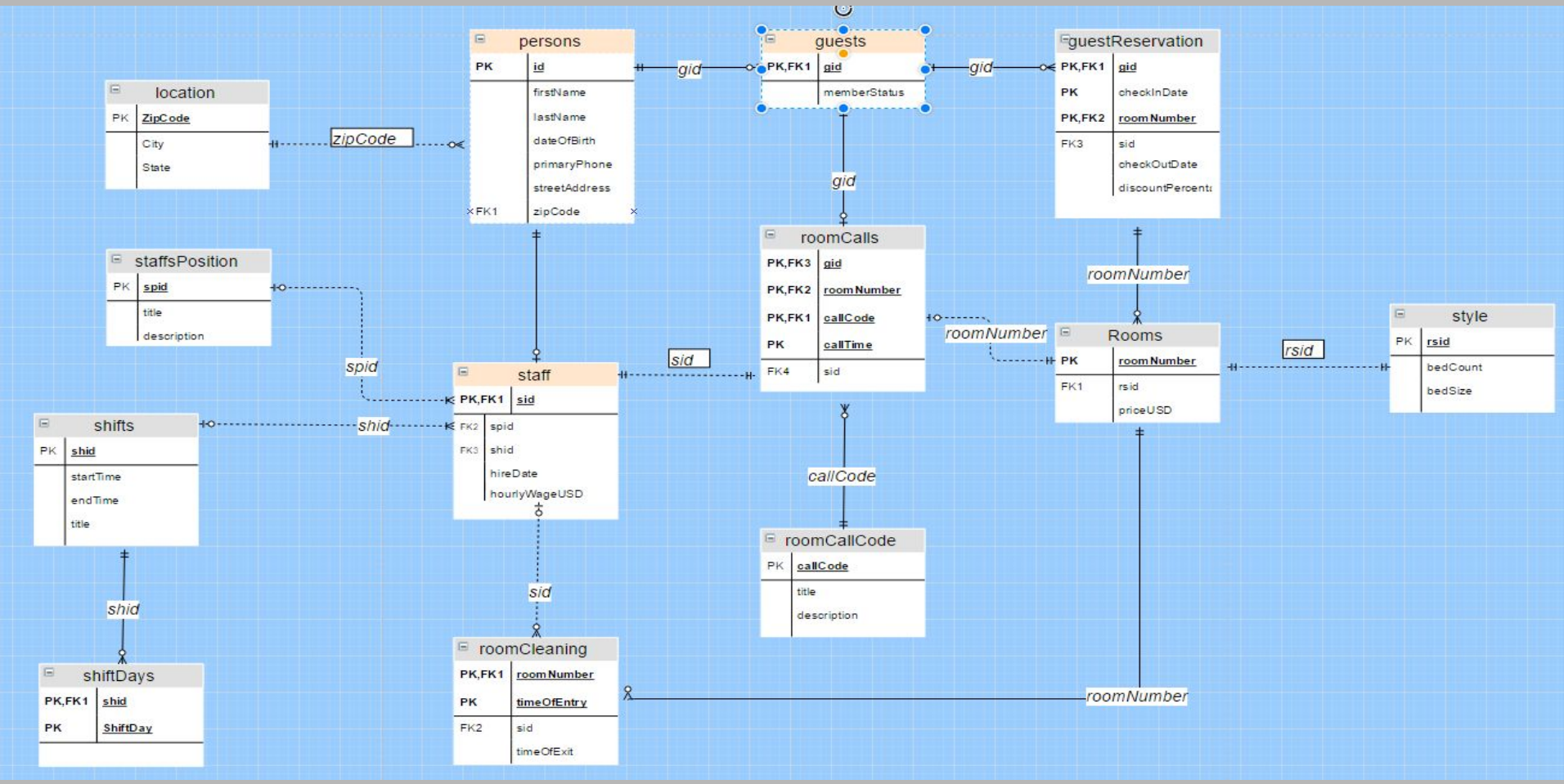
By: Dan Njoku

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



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

```
CREATE TABLE location (
    zipCode          INTEGER          NOT NULL,
    city             VARCHAR(50)      NOT NULL,
    state            VARCHAR(2)       NOT NULL,
PRIMARY KEY(zipCode)
);
```

Function Dependencies

zipCode → city, state

	zipcode [PK] integer	city character varying(50)	state character varying(2)
1	9093	Eugene	OR
2	11533	Soddy Daisy	TX
3	12366	Smyra	CA
4	12433	Campbell	NC
5	12601	Poughkeepsie	NY
6	31021	Dublin	GA
7	34234	Jacksonville	FL
8	34343	Buffalo	NY
9	37379	Soddy Daisy	TN
10	45431	Uniondale	NY
11	99959	Garden City	NY

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 basic perosnal information regarding the individual.

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

Functional Dependencies

Pid → firstName, lastName, dateOfBirth,

primaryPhone, streetAddress, zipCode

Table is on the following page

	pid [PK] integer	firstname character varying(50)	lastname character varying(50)	dateofbirth timestamp without time zone	phoneprimary character(15)	streetaddress character varying(50)	zipcode integer
1	1	Dan	Smith	1968-12-08 00:00:00	231323276	22 Oak Street	37379
2	2	Will	Atkins	1963-08-30 00:00:00	8996439355	654 Glenwood Drive	31021
3	3	Bill	Cosby	1952-02-11 00:00:00	8442851051	252 13th Street	9093
4	4	Amy	Shummer	1958-09-19 00:00:00	8110049929	396 Grove Street	12366
5	5	Josh	Merrick	1983-09-02 00:00:00	8554809490	227 Devonshire Drive	45431
6	6	Randy	Queen	1963-07-02 00:00:00	8993496951	138 Devon Court	99959
7	7	Dante	Matthew	1960-05-29 00:00:00	8553744253	19 Water Street	34234
8	8	Harry	Johnson	1958-01-09 00:00:00	8555815117	125 Spring Street	11533
9	9	Marco	Birkshire	1993-07-02 00:00:00	8339754754	440 Canal Street	11533
10	10	Ruben	Hathaway	1971-09-19 00:00:00	8990479981	736 3rd Street North	11533
11	11	Mike	Forte	1981-12-08 00:00:00	8446578110	529 Laurel Street	12601
12	12	Cameron	Nero	1955-12-27 00:00:00	8220221397	595 Parker Street	12601
13	13	Katie	Carty	1992-05-22 00:00:00	8557264814	602 6th Street North	12601
14	14	Siobain	Rogers	1978-06-08 00:00:00	8116286109	315 Redwood Drive	12601
15	15	Scott	Botts	1977-04-20 00:00:00	8226757964	973 Lakeview Drive	12601
16	16	Sarah	Rogers	1969-01-08 00:00:00	8334068368	92 Beechwood Drive	12601
17	17	Annie	Dumpty	1988-07-05 00:00:00	8442569423	618 Madison Avenue	12601
18	18	Lee	Wee	1998-12-08 00:00:00	8444814554	618 Madison Avenue	12601
19	19	Will	Williams	1977-02-08 00:00:00	8994765763	944 George Street	12601
20	20	Amanda	Scott	1994-11-08 00:00:00	8441270429	644 Marshall Street	12601

Guests

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

```
CREATE TABLE guests (
    gid             INTEGER    NOT NULL  REFERENCES persons(pid),
    memberStatus    VARCHAR    NOT NULL  CHECK
    (memberStatus IN ('Bronze', 'Silver','Gold', 'Platinum', 'Diamond')),
PRIMARY KEY(gid)
);
```

Function Dependencies

gid → memberStatus

	gid [PK] integer	memberstatus character varying
1	1	Bronze
2	2	Bronze
3	3	Bronze
4	4	Bronze
5	5	Silver
6	6	Bronze
7	7	Bronze
8	8	Gold
9	9	Bronze
10	10	Platinum

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

Function Dependencies

shid → startTime, endTime, title

```
CREATE TABLE shifts (
    shid            INTEGER             NOT NULL,
    startTime       TIME                NOT NULL,
    endTime         TIME                NOT NULL,
    title           VARCHAR,
PRIMARY KEY(shid)
);
```

	shid [PK] integer	starttime time without time zone	endtime time without time zone	title character varying
1	0	00:00:00	06:00:00	Morning
2	1	06:00:00	12:00:00	Day
3	2	12:00:00	18:00:00	Evening
4	3	18:00:00	00:00:00	Night

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

```
CREATE TABLE staffsPosition (
    spid            INTEGER             NOT NULL,
    title           VARCHAR             NOT NULL,
    description     VARCHAR,
PRIMARY KEY(spid)
);
```

Function Dependencies

spid → title, description

	spid [PK] integer	title character varying	description character varying
1	1	Administration	Managerial roll
2	2	Cleaner	Cleans the hotel room

Style

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

```
CREATE TABLE style (
    rsid            INTEGER             NOT NULL,
    bedCount        CHAR(1)             NOT NULL,
    bedSize         VARCHAR             NOT NULL,
PRIMARY KEY(rsid)
);
```

	rsid [PK] integer	bedcount character(1)	bedsize character varying
1	1	1	King
2	2	1	Queen
3	3	1	Full
4	4	2	Queen
5	5	2	Full
6	6	2	Twin
7	7	2	King

Function Dependencies

rsid → bedCount, bedSize

Rooms Gives the details of each room located in the hotel

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

	roomnumber [PK] character varying(5)	rsid integer	priceusd numeric
1	101	1	200
2	102	2	150
3	103	3	120
4	104	4	250
5	105	5	200
6	201	6	200
7	202	2	150
8	203	3	120
9	204	4	250
10	205	7	350

Function Dependencies

roomNumber→ rsid, priceUSD

Shift Days Specifies what shifts are available on each specific day of the week

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

Function Dependencies

shid, shiftDay →

Table is located on the following page

Shift Days Table

	shid [PK] integer	shiftday [PK] character varying
1	0	Friday
2	0	Monday
3	0	Saturday
4	0	Sunday
5	0	Thursday
6	0	Tuesday
7	0	Wednesday
8	1	Friday
9	1	Monday
10	1	Saturday
11	1	Sunday
12	1	Thursday
13	1	Tuesday
14	1	Wednesday
15	2	Friday
16	2	Monday
17	2	Saturday
18	2	Sunday
19	2	Thursday
20	2	Tuesday
21	2	Wednesday
22	3	Friday
23	3	Monday
24	3	Saturday
25	3	Sunday
26	3	Thursday
27	3	Tuesday
28	3	Wednesday

Room Call Codes

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

```
CREATE TABLE roomCallCode (
    callCode        INTEGER             NOT NULL,
    title           VARCHAR(50)         NOT NULL,
    description     VARCHAR             NOT NULL,
PRIMARY KEY(callCode)
);
```

Function Dependencies

callCode → title, description

	callcode [PK] integer	title character varying(50)	description character varying
1	1	Maintainance	If its broke we fix it
2	2	Cleaning	New sheets? No problem
3	3	Room Service	You Hungry?
4	4	Help Desk	You have a question? Well we have answers

Staff

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

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

	sid [PK] integer	spid integer	shid integer	hiredate timestamp without time zone	hourlywageusd numeric
1	11	1	3	2015-02-11 00:00:00	15.25
2	12	1	3	2015-09-15 00:00:00	10.25
3	13	1	2	2015-04-13 00:00:00	14.25
4	14	1	2	2015-08-12 00:00:00	10.25
5	15	1	0	2015-11-12 00:00:00	10.25
6	16	1	2	2015-02-13 00:00:00	13.25
7	17	2	0	2015-02-11 00:00:00	10.25
8	18	2	1	2015-12-17 00:00:00	11.25
9	19	2	2	2015-12-11 00:00:00	9.25
10	20	2	3	2015-02-11 00:00:00	9.25

Function Dependencies

sid → spid, shid, hireDate, hourlyWage

Room Calls

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

```
CREATE TABLE roomCall (
    gid             INTEGER             NOT NULL REFERENCES guests(gid),
    roomNumber      VARCHAR(5)          NOT NULL REFERENCES rooms(roomNumber),
    callCode        INTEGER             NOT NULL REFERENCES roomCallCode(callCode),
    callTime        TIME                NOT NULL,
    sid             INTEGER             NOT NULL REFERENCES staff(sid),
PRIMARY KEY(callTime, roomNumber, callCode, gid)
);
```

	gid integer	roomnumber character varying(5)	callcode integer	calltime [PK] time without time zone	sid integer
1	1	101	4	18:19:19	11
2	3	103	2	19:20:19	12
3	1	101	4	20:19:19	11

Function Dependencies

gid, roomNumber, callCode, callTime → sid

Room Cleaning

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

```
CREATE TABLE roomCleaning (
    roomNumber    VARCHAR(5)            NOT NULL  REFERENCES rooms(roomNumber),
    timeOfEntry   TIME                  NOT NULL,
    sid           INTEGER               NOT NULL  REFERENCES staff(sid),
    timeOfExit    TIME                  NOT NULL,
PRIMARY KEY(roomNumber, timeOfEntry)
);
```

	roomnumber [PK] character varying(5)	timeofentry [PK] time without time zone	sid integer	timeofexit time without time zone
1	101	12:00:00	17	12:30:00
2	103	15:00:00	17	15:30:00
3	105	14:00:00	17	14:30:00
4	201	16:00:00	17	16:30:00

Function Dependencies

roomNumber, timeOfEntry → sid, timeOfExit

Guest Reservation

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

```
CREATE TABLE guestReservation (
    gid                 INTEGER     NOT NULL REFERENCES guests(gid),
    roomNumber          VARCHAR(5)  NOT NULL REFERENCES rooms(roomNumber),
    sid                 INTEGER     NOT NULL REFERENCES staff(sid),
    checkInDate         TIMESTAMP   NOT NULL,
    checkOutDate        TIMESTAMP   NOT NULL,
    discountPercentage  DECIMAL     NOT NULL,
PRIMARY KEY(gid,roomNumber,checkInDate)
);
```

Function Dependencies

gid, roomNumber → sid,checkInDate, checkOutDate, discountPercentage

	gid [PK] integer	roomnumber [PK] character varying(5)	sid integer	checkindate [PK] timestamp without time zone	checkoutdate timestamp without time zone	discountpercentage numeric
1	1	101	13	2016-04-21 00:00:00	2016-04-23 00:00:00	0
2	2	102	13	2016-04-22 00:00:00	2016-04-23 00:00:00	0
3	3	103	13	2016-04-23 00:00:00	2016-04-24 00:00:00	0
4	4	104	13	2016-04-23 00:00:00	2016-04-25 00:00:00	0
5	5	105	13	2016-04-20 00:00:00	2016-04-23 00:00:00	5
6	6	201	13	2016-04-20 00:00:00	2016-04-23 00:00:00	0
7	7	202	13	2016-04-21 00:00:00	2016-04-23 00:00:00	0
8	8	203	13	2016-04-22 00:00:00	2016-04-26 00:00:00	10
9	9	204	13	2016-04-21 00:00:00	2016-04-23 00:00:00	0
10	10	205	13	2016-04-24 00:00:00	2016-04-27 00:00:00	12.5

Views

GuestRooms

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

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 = '2016-04-23'
ORDER BY per.lastname ASC;
```

staffInformation

- This view keeps a record of each employee throughout the hotel. It tracks their first name, last name,date of hire, and their hourly wage in USD.

Create statement

```
CREATE OR REPLACE VIEW staffInformation AS
       SELECT per.firstName,
              per.lastName,
              s.hireDate,
              s.hourlyWageUSD
       FROM   persons per,
              staff s
       WHERE  s.sid = per.pid
       ORDER BY per.pid DESC
```

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 cleaner;
GRANT SELECT ON rooms TO cleaner;
```

Administration

```
GRANT SELECT ON location TO Administration;
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.