Project Title: JAT Hotel Reservation
Team Name: The Data Miners

Team Members: Jay Patel, Antonio Ontiveros, Tran Nguyen

Database Schema

```
CREATE DATABASE  IF NOT EXISTS `jat_reservation` /*!40100 DEFAULT CHARACTER SET
latin1 */;
USE `jat_reservation`;
-- MySQL dump 10.13  Distrib 5.6.17, for Win32 (x86)
--
-- Host: 127.0.0.1    Database: jat_reservation
-- ------------------------------------------------------
-- Server version  5.6.20

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0
*/;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `customer`
--

DROP TABLE IF EXISTS `customer`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `customer` (
  `cID` int(11) NOT NULL,
  `hID` int(11) NOT NULL,
  `rID` int(11) NOT NULL,
  `name` varchar(20) NOT NULL,
  `address` varchar(250) DEFAULT NULL,
  `ccNo` bigint(20) NOT NULL,
  `smoker` tinyint(1) NOT NULL DEFAULT '0',
  `rStartDate` date NOT NULL,
  `rEndDate` date NOT NULL,
  `discount` int(11) DEFAULT '0',
  `updatedAt` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
  PRIMARY KEY (`cID`,`hID`),
  KEY `fk_Customer_Rooms1_idx` (`rID`,`hID`),
  CONSTRAINT `fk_Customer_Rooms1` FOREIGN KEY (`rID`, `hID`) REFERENCES `rooms`
(`rID`, `hID`) ON DELETE CASCADE ON UPDATE NO ACTION
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `customerarchiving`
--

DROP TABLE IF EXISTS `customerarchiving`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `customerarchiving` (
  `cID` int(11) NOT NULL,
  `hID` int(11) NOT NULL,
  `rID` int(11) DEFAULT NULL,
  `name` varchar(20) DEFAULT NULL,
  `address` varchar(250) DEFAULT NULL,
  `ccNo` bigint(20) DEFAULT NULL,
  `smoker` tinyint(1) DEFAULT NULL,
  `rStartDate` date DEFAULT NULL,
  `rEndDate` date DEFAULT NULL,
  `discount` int(11) DEFAULT NULL,
  `updatedAt` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`cID`,`hID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

/*!50003 SET @saved_cs_client      = @@character_set_client */ ;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;
/*!50003 SET @saved_col_connection = @@collation_connection */ ;
/*!50003 SET character_set_client  = utf8 */ ;
/*!50003 SET character_set_results = utf8 */ ;
/*!50003 SET collation_connection  = utf8_general_ci */ ;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;
/*!50003 SET sql_mode              = 'NO_ENGINE_SUBSTITUTION' */ ;
DELIMITER ;;
/*!50003 CREATE*/ /*!50017 DEFINER=`root`@`localhost`*/ /*!50003 TRIGGER
`courtesyValet`
AFTER INSERT ON `customer`
FOR EACH ROW
BEGIN
IF(DATEDIFF(NEW.rEndDate, NEW.rStartDate) > 14 AND (New.cID, New.hID) NOT IN
(SELECT cID, hID FROM parking where hID = NEW.hID))
THEN
INSERT INTO parking (hID, valet, cID)
VALUES (NEW.hID, 1, NEW.cID);
END IF;
END */;;
DELIMITER ;
```

```
/*!50003 SET sql_mode              = @saved_sql_mode */ ;
/*!50003 SET character_set_client  = @saved_cs_client */ ;
/*!50003 SET character_set_results = @saved_cs_results */ ;
/*!50003 SET collation_connection  = @saved_col_connection */ ;

--
-- Table structure for table `employee`
--

DROP TABLE IF EXISTS `employee`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `employee` (
  `eID` int(11) NOT NULL,
  `hID` int(11) NOT NULL,
  `name` varchar(20) NOT NULL,
  `position` varchar(30) NOT NULL,
  `salary` double NOT NULL DEFAULT '40000',
  PRIMARY KEY (`eID`,`hID`),
  KEY `fk_Employee_Hotels1_idx` (`hID`),
  CONSTRAINT `fk_Employee_Hotels1` FOREIGN KEY (`hID`) REFERENCES `hotels`
(`hID`) ON DELETE CASCADE ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `hotels`
--

DROP TABLE IF EXISTS `hotels`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `hotels` (
  `hID` int(11) NOT NULL AUTO_INCREMENT,
  `companyName` varchar(50) NOT NULL,
  `location` varchar(250) NOT NULL,
  `totalrooms` int(11) NOT NULL DEFAULT '10',
  PRIMARY KEY (`hID`)
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;



--
-- Table structure for table `managerlogin`
--

DROP TABLE IF EXISTS `managerlogin`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
```

```
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `managerlogin` (
  `username` varchar(25) NOT NULL,
  `password` varchar(25) NOT NULL,
  `employee_eID` int(11) NOT NULL,
  `employee_hID` int(11) NOT NULL,
  PRIMARY KEY (`username`,`employee_eID`,`employee_hID`),
  KEY `fk_managerlogin_employee1_idx` (`employee_eID`,`employee_hID`),
  CONSTRAINT `fk_managerlogin_employee1` FOREIGN KEY (`employee_eID`,
`employee_hID`) REFERENCES `employee` (`eID`, `hID`) ON DELETE CASCADE ON
UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `parking`
--

DROP TABLE IF EXISTS `parking`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `parking` (
  `pID` int(11) NOT NULL AUTO_INCREMENT,
  `hID` int(11) NOT NULL,
  `valet` tinyint(1) NOT NULL DEFAULT '1',
  `cID` int(11) NOT NULL,
  `updatedAt` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
  PRIMARY KEY (`pID`,`hID`),
  KEY `fk_Parking_Customer1_idx` (`cID`,`hID`),
  CONSTRAINT `fk_Parking_Customer1` FOREIGN KEY (`cID`, `hID`) REFERENCES
`customer` (`cID`, `hID`) ON DELETE CASCADE ON UPDATE NO ACTION
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `parkingarchiving`
--

DROP TABLE IF EXISTS `parkingarchiving`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `parkingarchiving` (
  `pID` int(11) NOT NULL,
  `hID` int(11) NOT NULL,
  `valet` tinyint(1) DEFAULT NULL,
  `cID` int(11) DEFAULT NULL,
  `updatedAt` timestamp NULL DEFAULT NULL,
```

```
  PRIMARY KEY (`pID`,`hID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `rating`
--

DROP TABLE IF EXISTS `rating`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `rating` (
  `ratingID` int(11) NOT NULL AUTO_INCREMENT,
  `hID` int(11) NOT NULL,
  `rating` int(11) NOT NULL DEFAULT '0',
  `review` varchar(500) DEFAULT NULL,
  PRIMARY KEY (`ratingID`),
  KEY `fk_rating_hotels1_idx` (`hID`),
  CONSTRAINT `fk_rating_hotels1` FOREIGN KEY (`hID`) REFERENCES `hotels`
(`hID`) ON DELETE CASCADE ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `rating`
--

LOCK TABLES `rating` WRITE;
/*!40000 ALTER TABLE `rating` DISABLE KEYS */;
/*!40000 ALTER TABLE `rating` ENABLE KEYS */;
UNLOCK TABLES;
/*!50003 SET @saved_cs_client      = @@character_set_client */ ;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;
/*!50003 SET @saved_col_connection = @@collation_connection */ ;
/*!50003 SET character_set_client  = utf8 */ ;
/*!50003 SET character_set_results = utf8 */ ;
/*!50003 SET collation_connection  = utf8_general_ci */ ;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;
/*!50003 SET sql_mode              = 'NO_ENGINE_SUBSTITUTION' */ ;
DELIMITER ;;
/*!50003 CREATE*/ /*!50017 DEFINER=`root`@`localhost`*/ /*!50003 TRIGGER bonus
AFTER INSERT ON rating
FOR EACH ROW
BEGIN
      IF (new.rating >= 4) THEN
            UPDATE employee
            SET salary = salary + 10
            WHERE employee.hID = new.hID;
```

```
      END IF ;
END */;;
DELIMITER ;
/*!50003 SET sql_mode              = @saved_sql_mode */ ;
/*!50003 SET character_set_client  = @saved_cs_client */ ;
/*!50003 SET character_set_results = @saved_cs_results */ ;
/*!50003 SET collation_connection  = @saved_col_connection */ ;

--
-- Table structure for table `rooms`
--

DROP TABLE IF EXISTS `rooms`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `rooms` (
  `rID` int(11) NOT NULL,
  `hID` int(11) NOT NULL,
  `smoking` tinyint(1) NOT NULL DEFAULT '0',
  `price` double NOT NULL DEFAULT '100',
  PRIMARY KEY (`rID`,`hID`),
  KEY `fk_Rooms_Hotels_idx` (`hID`),
  CONSTRAINT `fk_Rooms_Hotels` FOREIGN KEY (`hID`) REFERENCES `hotels` (`hID`)
ON DELETE CASCADE ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Temporary table structure for view `viewratings`
--

DROP TABLE IF EXISTS `viewratings`;
/*!50001 DROP VIEW IF EXISTS `viewratings`*/;
SET @saved_cs_client     = @@character_set_client;
SET character_set_client = utf8;
/*!50001 CREATE TABLE `viewratings` (
  `companyName` tinyint NOT NULL,
  `rating` tinyint NOT NULL,
  `review` tinyint NOT NULL
) ENGINE=MyISAM */;
SET character_set_client = @saved_cs_client;

--
-- Dumping routines for database 'jat_reservation'
--
/*!50003 DROP PROCEDURE IF EXISTS `archive` */;
/*!50003 SET @saved_cs_client      = @@character_set_client */ ;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;
```

```
/*!50003 SET @saved_col_connection = @@collation_connection */ ;
/*!50003 SET character_set_client  = utf8 */ ;
/*!50003 SET character_set_results = utf8 */ ;
/*!50003 SET collation_connection  = utf8_general_ci */ ;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;
/*!50003 SET sql_mode              = 'NO_ENGINE_SUBSTITUTION' */ ;
DELIMITER ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `archive`(IN d DATE)
BEGIN
      INSERT INTO customerArchiving
      SELECT *
      FROM customer
      WHERE updatedAt < d AND (cID, hID) NOT IN (select cID, hID FROM
customerArchiving);

      INSERT INTO parkingArchiving
      SELECT *
      FROM parking
      WHERE updatedAt < d AND (pID, hID) NOT IN (select pID, hID FROM
parkingArchiving);
END ;;
DELIMITER ;
/*!50003 SET sql_mode              = @saved_sql_mode */ ;
/*!50003 SET character_set_client  = @saved_cs_client */ ;
/*!50003 SET character_set_results = @saved_cs_results */ ;
/*!50003 SET collation_connection  = @saved_col_connection */ ;
/*!50003 DROP PROCEDURE IF EXISTS `cancelReservation` */;
/*!50003 SET @saved_cs_client      = @@character_set_client */ ;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;
/*!50003 SET @saved_col_connection = @@collation_connection */ ;
/*!50003 SET character_set_client  = utf8 */ ;
/*!50003 SET character_set_results = utf8 */ ;
/*!50003 SET collation_connection  = utf8_general_ci */ ;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;
/*!50003 SET sql_mode              = 'NO_ENGINE_SUBSTITUTION' */ ;
DELIMITER ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `cancelReservation`(IN sDate DATE,
IN eDate DATE, IN hotel VARCHAR(50) CHARSET utf8, IN roomid INT, IN loc
VARCHAR(50) CHARSET utf8)
BEGIN
      DELETE FROM customer
      WHERE rID=roomid AND rStartDate=sDate AND rEndDate=eDate
      AND hID IN (SELECT hID FROM hotels WHERE companyName=hotel AND
location=loc);
END ;;
DELIMITER ;
/*!50003 SET sql_mode              = @saved_sql_mode */ ;
/*!50003 SET character_set_client  = @saved_cs_client */ ;
```

```
/*!50003 SET character_set_results = @saved_cs_results */ ;
/*!50003 SET collation_connection  = @saved_col_connection */ ;
/*!50003 DROP PROCEDURE IF EXISTS `ComputeTotalPrice` */;
/*!50003 SET @saved_cs_client      = @@character_set_client */ ;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;
/*!50003 SET @saved_col_connection = @@collation_connection */ ;
/*!50003 SET character_set_client  = utf8 */ ;
/*!50003 SET character_set_results = utf8 */ ;
/*!50003 SET collation_connection  = utf8_general_ci */ ;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;
/*!50003 SET sql_mode              = 'NO_ENGINE_SUBSTITUTION' */ ;
DELIMITER ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `ComputeTotalPrice`(IN cID INT, IN
hID INT, IN name VARCHAR(20), OUT price DOUBLE)
BEGIN
      Select ((r.price * (rEndDate - rStartDate)) - (r.price * (rEndDate -
rStartDate) * (c.discount / 100))) INTO price
      From customer c natural join rooms r
      where c.cID = cID and c.hID = hID and c.name = name;
END ;;
DELIMITER ;
/*!50003 SET sql_mode              = @saved_sql_mode */ ;
/*!50003 SET character_set_client  = @saved_cs_client */ ;
/*!50003 SET character_set_results = @saved_cs_results */ ;
/*!50003 SET collation_connection  = @saved_col_connection */ ;
/*!50003 DROP PROCEDURE IF EXISTS `rateHotel` */;
/*!50003 SET @saved_cs_client      = @@character_set_client */ ;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;
/*!50003 SET @saved_col_connection = @@collation_connection */ ;
/*!50003 SET character_set_client  = utf8 */ ;
/*!50003 SET character_set_results = utf8 */ ;
/*!50003 SET collation_connection  = utf8_general_ci */ ;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;
/*!50003 SET sql_mode              = 'NO_ENGINE_SUBSTITUTION' */ ;
DELIMITER ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `rateHotel`(IN hid INT, IN numstars
INT, IN review VARCHAR(500) CHARSET utf8)
BEGIN
      INSERT INTO rating (hID, rating, review)
      VALUES(hid, numstars, review);
END ;;
DELIMITER ;
/*!50003 SET sql_mode              = @saved_sql_mode */ ;
/*!50003 SET character_set_client  = @saved_cs_client */ ;
/*!50003 SET character_set_results = @saved_cs_results */ ;
/*!50003 SET collation_connection  = @saved_col_connection */ ;

--
```

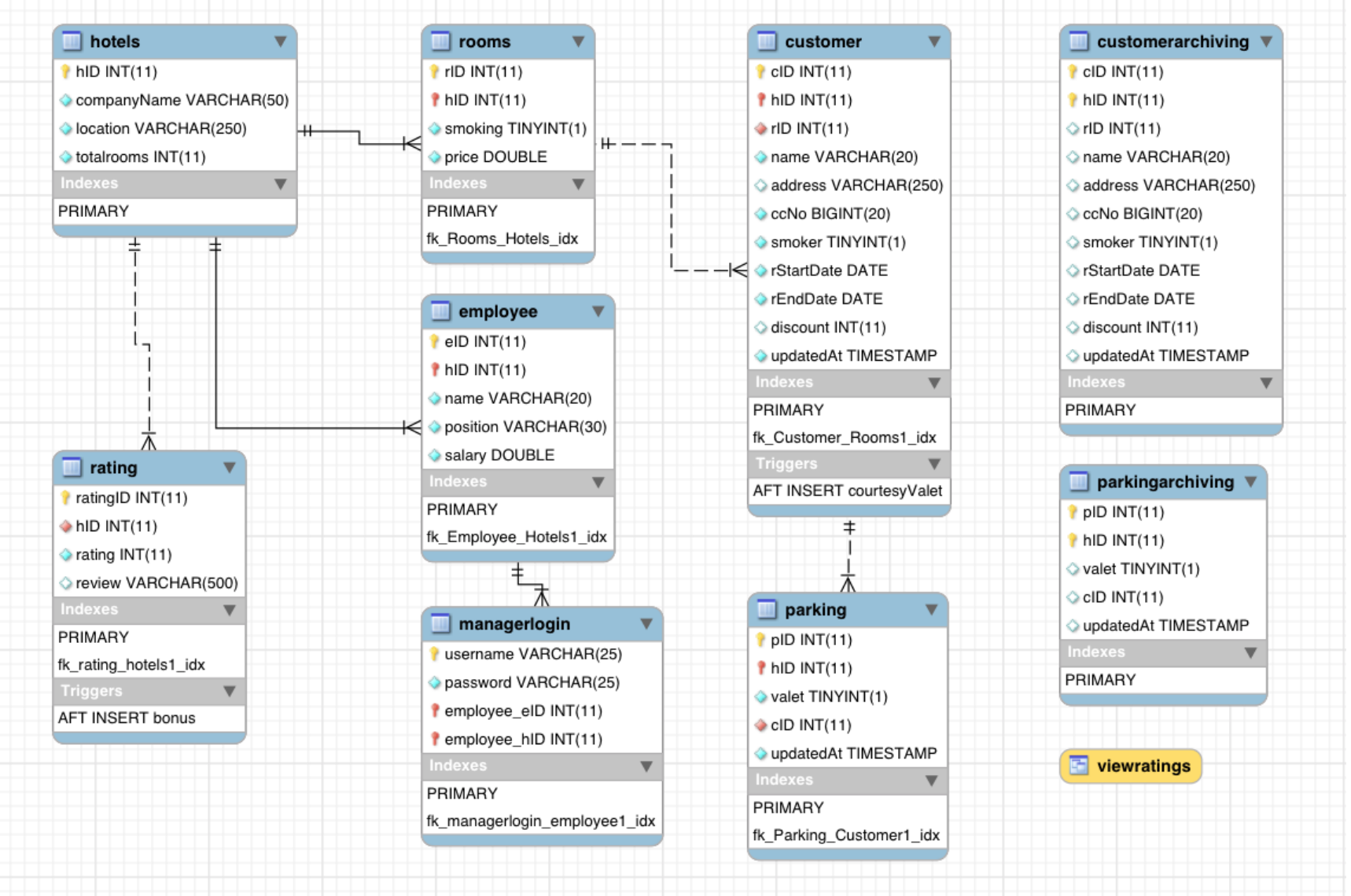
```
-- Final view structure for view `viewratings`
--

/*!50001 DROP TABLE IF EXISTS `viewratings`*/;
/*!50001 DROP VIEW IF EXISTS `viewratings`*/;
/*!50001 SET @saved_cs_client          = @@character_set_client */;
/*!50001 SET @saved_cs_results         = @@character_set_results */;
/*!50001 SET @saved_col_connection     = @@collation_connection */;
/*!50001 SET character_set_client      = utf8 */;
/*!50001 SET character_set_results     = utf8 */;
/*!50001 SET collation_connection      = utf8_general_ci */;
/*!50001 CREATE ALGORITHM=UNDEFINED */
/*!50013 DEFINER=`root`@`localhost` SQL SECURITY DEFINER */
/*!50001 VIEW `viewratings` AS select `hotels`.`companyName` AS
`companyName`,`rating`.`rating` AS `rating`,`rating`.`review` AS `review` from
(`rating` left join `hotels` on((`rating`.`hID` = `hotels`.`hID`))) order by
`hotels`.`companyName` */;
/*!50001 SET character_set_client      = @saved_cs_client */;
/*!50001 SET character_set_results     = @saved_cs_results */;
/*!50001 SET collation_connection      = @saved_col_connection */;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;
-- Dump completed on 2014-12-04 10:54:10


-- Load Functionality

LOAD DATA LOCAL INFILE 'c:\\mysql\\hotels.txt' INTO TABLE HOTELS;
LOAD DATA LOCAL INFILE 'c:\\mysql\\rooms.txt' INTO TABLE ROOMS;
LOAD DATA LOCAL INFILE 'c:\\mysql\\customers.txt' INTO TABLE CUSTOMER;
LOAD DATA LOCAL INFILE 'c:\\mysql\\employee.txt' INTO TABLE EMPLOYEE;
LOAD DATA LOCAL INFILE 'c:\\mysql\\managerlogin.txt' INTO TABLE MANAGERLOGIN;
```


hotels
hID INT(11)
companyName VARCHAR(50)
location VARCHAR(250)
totalrooms INT(11)
Indexes
PRIMARY
rooms
rID INT(11)
hID INT(11)
smoking TINYINT(1)
price DOUBLE
Indexes
PRIMARY
fk_Rooms_Hotels_idx
customer
cID INT(11)
hID INT(11)
rID INT(11)
name VARCHAR(20)
address VARCHAR(250)
ccNo BIGINT(20)
smoker TINYINT(1)
rStartDate DATE
rEndDate DATE
discount INT(11)
updatedAt TIMESTAMP
Indexes
PRIMARY
fk_Customer_Rooms1_idx
Triggers
AFT INSERT courtesyValet
customerarchiving
cID INT(11)
hID INT(11)
rID INT(11)
name VARCHAR(20)
address VARCHAR(250)
ccNo BIGINT(20)
smoker TINYINT(1)
rStartDate DATE
rEndDate DATE
discount INT(11)
updatedAt TIMESTAMP
Indexes
PRIMARY
employee
eID INT(11)
hID INT(11)
name VARCHAR(20)
position VARCHAR(30)
salary DOUBLE
Indexes
PRIMARY
fk_Employee_Hotels1_idx
rating
ratingID INT(11)
hID INT(11)
rating INT(11)
review VARCHAR(500)
Indexes
PRIMARY
fk_rating_hotels1_idx
Triggers
AFT INSERT bonus
managerlogin
username VARCHAR(25)
password VARCHAR(25)
employee_eID INT(11)
employee_hID INT(11)
Indexes
PRIMARY
fk_managerlogin_employee1_idx
parking
pID INT(11)
hID INT(11)
valet TINYINT(1)
cID INT(11)
updatedAt TIMESTAMP
Indexes
PRIMARY
fk_Parking_Customer1_idx
parkingarchiving
pID INT(11)
hID INT(11)
valet TINYINT(1)
cID INT(11)
updatedAt TIMESTAMP
Indexes
PRIMARY
viewratings

Snapshot of Relations After Populating Initial Data

customer

cID	hID	rID	name	address	ccNo	smoker	rStartDate	rEndDate	discount	updatedAt
1	1	1	Bruce Wayne	9999 Main St. San Jose, CA 95131	111122223334	1	2014-12-22	2014-12-30	0	2014-11-16 17:10:53
1	2	1	Loki	9999 Some Universer Asgard, OS 96131	111122223324	1	2015-01-22	2015-01-30	5	2014-11-16 17:10:53
1	3	1	Anna	8888 Some Universer Asgard, OS 97131	111322223324	1	2015-10-22	2015-10-30	15	2014-11-16 17:10:53
1	4	1	Sofia	8888 New York St. San Mateo, CA 95131	110022223324	1	2015-01-20	2015-01-30	10	2014-11-16 17:10:53
1	5	1	Colton	777 Main St. San Jose, CA 95131	111120023334	1	2014-12-20	2014-12-30	0	2014-11-16 17:10:53
2	1	2	Clark Kent	777 First St. Washington, MA 95131	111122223335	1	2015-04-14	2015-04-30	10	2014-11-16 17:10:53
2	2	2	Iron Man	666 Universal Studios Anahiem, CA 978651	111122223315	1	2015-06-14	2015-06-30	0	2014-11-16 17:10:53
2	3	2	Aaliyah	909 Universal Studios Anahiem, CA 978651	111132223315	1	2015-11-14	2015-11-30	5	2014-11-16 17:10:53
2	4	2	Grace	707 Professor Oak Pokemon, LN 12345	111022223315	1	2015-06-14	2015-06-20	10	2014-11-16 17:10:53
2	5	2	Jace	807 No St. Washington, MA 95131	111122223300	1	2015-04-01	2015-04-30	0	2014-11-16 17:10:53
3	1	6	Diana	907 Second St. New York, NY 95131	111122223336	0	2014-12-22	2014-12-30	0	2014-11-16 17:10:53
3	2	6	Super Man	888 California St. San Francisco, CA 95131	111122223339	0	2015-12-22	2015-12-30	5	2014-11-16 17:10:53
3	3	6	Alexis	8888 California St. San Francisco, CA 95131	111123223339	0	2015-09-22	2015-09-30	5	2014-11-16 17:10:53
3	4	6	Peyton	8888 Washington St. Gotham, NH 95131	110122223339	0	2015-12-22	2015-12-25	10	2014-11-16 17:10:53
3	5	6	Angel	8888 Some St. New York, NY 95131	111002223336	0	2015-11-22	2015-12-05	0	2014-11-16 17:10:53

customerarchiving

cID	hID	rID	name	address	ccNo	smoker	rStartDate	rEndDate	discount	updatedAt
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

employee

eID	hID	name	position	salary
1	1	James	Owner	100000
1	2	George	Owner	120000
1	3	Jeff	Owner	105000
1	4	Joseph	Owner	60000
1	5	Kevin	Owner	200000
2	1	John	Manager	60000
2	2	Steven	Manager	80000
2	3	Lee	Manager	65000
2	4	Thomas	Manager	50000
2	5	Jason	Manager	105000
3	1	Robert	Regular	40000
3	2	Edward	Regular	60000
3	3	Ray	Regular	45000
3	4	Donald	Regular	40000
3	5	Gary	Regular	80000
NULL	NULL	NULL	NULL	NULL

hotels

hID	companyName	location	totalrooms
1	Hilton	San Francisco	30
2	Marriott	New York	50
3	Embassy Suites	Boston	35
4	Hyatt	Chicago	20
5	Caesars Palace	Las Vegas	60
NULL	NULL	NULL	NULL

managerlogin

username	password	employee_eID	employee_hID
john	secret	2	1
NULL	NULL	NULL	NULL

parking

pID	hID	valet	cID	updatedAt
1	1	1	2	2014-12-06 14:47:06
2	2	1	2	2014-12-06 14:47:06
3	3	1	2	2014-12-06 14:47:06
4	5	1	2	2014-12-06 14:47:06
NULL	NULL	NULL	NULL	NULL

parkingarchiving

pID	hID	valet	cID	updatedAt
NULL	NULL	NULL	NULL	NULL

rating

ratingID	hID	rating	review
NULL	NULL	NULL	NULL

rooms

rID	hID	smoking	price
1	1	1	150
2	1	1	150
3	1	1	150
4	1	1	150
5	1	1	150
6	1	0	125
7	1	0	125
8	1	0	125
9	1	0	125
10	1	0	125
1	2	1	175
2	2	1	175
3	2	1	175
4	2	1	175
5	2	1	175
6	2	0	150
7	2	0	150
8	2	0	150
9	2	0	150
10	2	0	150

rID	hID	smoking	price
1	3	1	150
2	3	1	150
3	3	1	150
4	3	1	150
5	3	1	150
6	3	0	125
7	3	0	125
8	3	0	125
9	3	0	125
10	3	0	125
1	4	1	125
2	4	1	125
3	4	1	125
4	4	1	125
5	4	1	125
6	4	0	100
7	4	0	100
8	4	0	100
9	4	0	100
10	4	0	100

rID	hID	smoking	price
1	5	1	250
2	5	1	250
3	5	1	250
4	5	1	250
5	5	1	250
6	5	0	225
7	5	0	225
8	5	0	225
9	5	0	225
10	5	0	225
NULL	NULL	NULL	NULL

15+ Distinct Functions

1. Reserve a Room
2. Book a Valet parking or Non-Valet Parking
3. Cancel Reservation
4. Leave Hotel Ratings
5. View Hotel Ratings
6. Compare/Contrast Hotel Prices
7. Manager--Registration
8. Manager--Login
9. Manager--Logout
10. Manager--View Revenue
11. Manager--Charge Customer
12. Manager--Reassign Customer's Room
13. Manager--Cancel Customer's Reservation
14. Manager--Check Available Rooms
15. Manager--Hire an Employee
16. Manager--Fire an Employee
17. Manager--Transfer an Employee

All SQL Select Statements

```
SELECT MIN(rooms.rID) rID
FROM rooms where hID = $hID
and rID NOT IN
      (SELECT rooms.rID
      FROM customer JOIN rooms
      WHERE customer.rID=rooms.rID AND customer.hID=$hID
      AND rStartDate<='$startDate' AND rEndDate>='$endDate')
and smoking = $smoke
```

---

[1]
```
SELECT distinct hotels.hID AS HID, hotels.companyName AS CNAME, rooms.rID AS
RID, rooms.smoking AS SMOKE, rooms.price AS PRICE
FROM hotels
      NATURAL JOIN rooms
      JOIN customer
WHERE location = '$location'
AND rooms.rID NOT IN
      (SELECT rooms.rID
      FROM customer
      WHERE customer.rID = rooms.rID
            AND rStartDate <= '$sDate'
            AND rEndDate >=  '$eDate')
```

[1] This is one of our complex query. This query satisfies the correlated query requirement.

[2]
```
SELECT sum(total) as revenue
FROM
      (SELECT a.hID, a.rID, price, name, smoker, ((rEndDate - rStartDate) *
price) - ((rEndDate - rStartDate) * price * (discount / 100)) as total
      from
            (select h.hID, r.rID, price FROM hotels h natural join rooms r) a
            left outer join
            customer c
            on a.rID=c.rID and a.hID=c.hID
      where cID is not null AND SMOKER=$option AND a.hID=$hid) q
```

[3]
```
SELECT sum(total) as revenue
From
      (SELECT a.hID, a.rID, price, name, smoker, ((rEndDate - rStartDate) *
price) - ((rEndDate - rStartDate) * price * (discount / 100)) as total
      from
            (select h.hID, r.rID, price FROM hotels h natural join rooms r) a
            left outer join customer c
            on a.rID=c.rID and a.hID=c.hID
      where cID is not null AND a.hID=$hid) q
```

[4]
```
SELECT companyName, avg(price) as price
FROM ROOMS NATURAL JOIN HOTELS
GROUP BY hID, SMOKING
HAVING smoking=$option
ORDER BY avg(price)
```

[5]
```
SELECT companyName, min(price) as min, max(price) as max
FROM ROOMS NATURAL JOIN HOTELS
GROUP BY hID
ORDER BY min(price), max(price)
```

```
Select ((r.price * (rEndDate - rStartDate)) - (r.price * (rEndDate -
rStartDate) * (c.discount / 100))) INTO price
From customer c natural join rooms r
where c.cID = cID and c.hID = hID and c.name = name;
```

[2] This is one of our complex query. This query satisfies the Outer Join query requirement.
[3] This is one of our complex query. This query satisfies the Outer Join query requirement.
[4] This is one of our complex query. This query satisfies the Group By and Having query requirement.
[5] This is one of our complex query. This query satisfies the aggregation query requirement.

---

[6]
```
SELECT rID
FROM
      (SELECT rID, hID
      FROM rooms
      WHERE hID = '$hID') AS a
LEFT OUTER JOIN
      (SELECT rID, hID
      FROM customer
      WHERE (rStartDate>='$startDate' AND  rEndDate<='$endDate')
      AND hID = '$hID') AS b
USING (rID, hID) WHERE b.rID IS NULL
```

```
SELECT rID
FROM
      (SELECT rID, hID FROM rooms WHERE hID=$hID) AS a
LEFT OUTER JOIN
      (SELECT rID, hID
      FROM customer
      WHERE
      DATE_FORMAT(rStartDate, \"%Y-%m\")>=DATE_FORMAT('$startDate', \"%Y-%m\")
      AND DATE_FORMAT(rStartDate, \"%Y-%m\")<=DATE_FORMAT('$endDate', \"%Y-%m\"))
      AND hID = $hID) AS b
USING (rID, hID) WHERE b.rID IS NULL
```

```
select location
from hotels
where hID=$hID
```

```
SELECT *
from hotels
WHERE companyName = '$hotel' AND location = '$city'
```

```
SELECT *
from customer
WHERE rStartDate='$sDate'
AND rEndDate='$eDate' AND rID='$roomid'
```

```
select companyName, location
from hotels
group by companyName
```

[6] This is one of our complex query. This query satisfies the Set operation query requirement. Since mysql does not support differance, we implemented the difference logic using Left outer join. The idea here is to figure out available rooms within certain time range. The way how we achieve this requirement is by ALL ROOMS - (CUSTOMERS living in a rooms during the entered time range).

```
SELECT *
FROM customer
where hID=$hID
```

```
SELECT @price AS price
```

```
SELECT *
FROM managerlogin
WHERE username='$username' and password='$password'
```

```
SELECT name
FROM employee
WHERE eID=$eid and hID=$hid
```

```
SELECT companyName
FROM hotels
WHERE hID=$hid
```

```
SELECT max(pID) + 1 AS number
FROM parking
```

```
SELECT max(eID) + 1
FROM employee
WHERE hID='$newhID';
```

```
SELECT companyName
FROM hotels
ORDER BY companyName
```

```
SELECT avg(rating) as avg, companyName, review
FROM viewratings
WHERE companyName=’companyName’
```

```
SELECT name, eID
FROM employee
WHERE hID='$hID' AND name<>'$managername'
```

```
select hID, eID
from employee natural join hotels
where position = \"Manager\" and name = \"$name\"
```

```
SELECT name, eID
FROM employee
WHERE hID = '$hID'
AND name <> '$managername'
AND position <> 'Owner'
```

```
SELECT companyName, hID
FROM hotels
WHERE hID <> '$hID'
```

```
select companyName
from hotels
```

```
SELECT hID
FROM hotels
WHERE companyName='$hotel'
```

```
SELECT max(cID) + 1 AS number
FROM customer
```

```
SELECT *
FROM hotels
ORDER BY companyName
```

```
SELECT *
FROM viewratings
WHERE companyName=’$h_name’
```

```
SELECT *
FROM viewratings
```

```
SELECT avg(rating) as avg, companyName
FROM viewratings
WHERE companyName='companyName’
```

All SQL Update Statements

```
UPDATE customer
SET rID=$rID
WHERE cID=$customerId AND hID=$hID
```

```
UPDATE employee
SET eID='$value', hID='$newhID'
WHERE eID='$eID' AND hID=$hID;
```

All SQL Delete Statements

```
DELETE FROM employee
WHERE eID='$eID' AND hID=$hID
```

```
DELETE FROM customer
      WHERE rID=roomid AND rStartDate=sDate AND rEndDate=eDate
      AND hID IN
            (SELECT hID FROM hotels
            WHERE companyName=hotel AND location=loc);
```

All SQL Insert Statements

```
INSERT INTO `parking` (`pID`, `hID`, `valet`, `cID`, `updatedAt`)
VALUES( '$pcount', '$hotel', '$parking', '$customer', CURRENT_TIMESTAMP);
```

```
INSERT INTO managerlogin(employee_hID, employee_eID, username, password)
values ($hID, $eID, \"$username\", \"$password\")
```

```
INSERT INTO employee
SELECT (max(eID) + 1), '$hID', '$employeeName', '$employeePosition',
'$employeeSalary'
FROM employee
WHERE hID = '$hID';
```

```
INSERT INTO `customer` (`cID`, `hID`, `rID`, `name`, `address`, `ccNo`,
`smoker`, `rStartDate`, `rEndDate`, `discount`, `updatedAt`)
VALUES( '$counter', '$HID', '$room', '$name', '$address', '$credit', '$smoke',
'$sDate', '$eDate', '$discount', CURRENT_TIMESTAMP);
```

```
INSERT INTO customerArchiving
      SELECT *
      FROM customer
      WHERE updatedAt < d AND (cID, hID) NOT IN (select cID, hID FROM
customerArchiving);
```

```
INSERT INTO parkingArchiving
      SELECT *
      FROM parking
      WHERE updatedAt < d AND (pID, hID) NOT IN (select pID, hID FROM
parkingArchiving);
```

```
INSERT INTO rating (hID, rating, review)
      VALUES(hid, numstars, review);
```

All SQL Triggers Statements

```
CREATE DEFINER=`root`@`localhost` TRIGGER `courtesyValet`
AFTER INSERT ON `customer`
FOR EACH ROW
BEGIN
      IF(DATEDIFF(NEW.rEndDate, NEW.rStartDate) > 14
      AND (New.cID, New.hID) NOT IN
            (SELECT cID, hID FROM parking where hID = NEW.hID))
      THEN
            INSERT INTO parking (hID, valet, cID)
            VALUES (NEW.hID, 1, NEW.cID);
END IF;
END
```

```
CREATE DEFINER=`root`@`localhost` TRIGGER bonus
AFTER INSERT ON rating
FOR EACH ROW
BEGIN
      IF (new.rating >= 4) THEN
            UPDATE employee
            SET salary = salary + 10
            WHERE employee.hID = new.hID;
      END IF ;
END
```

All Stored Procedures

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `archive`(IN d DATE)
BEGIN
      INSERT INTO customerArchiving
      SELECT *
      FROM customer
      WHERE updatedAt < d AND (cID, hID) NOT IN (select cID, hID FROM
customerArchiving);

      INSERT INTO parkingArchiving
      SELECT *
      FROM parking
      WHERE updatedAt < d AND (pID, hID) NOT IN (select pID, hID FROM
parkingArchiving);
END
CALL archive('2014-12-12');
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `cancelReservation`(IN sDate DATE,
IN eDate DATE, IN hotel VARCHAR(50) CHARSET utf8, IN roomid INT, IN loc
VARCHAR(50) CHARSET utf8)
BEGIN
      DELETE FROM customer
      WHERE rID=roomid AND rStartDate=sDate AND rEndDate=eDate
      AND hID IN (SELECT hID FROM hotels WHERE companyName=hotel AND
location=loc);
END
CALL cancelReservation('$sDate', '$eDate', '$hotel', '$roomid', '$city')
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `ComputeTotalPrice`(IN cID INT, IN
hID INT, IN name VARCHAR(20), OUT price DOUBLE)
BEGIN
      Select ((r.price * (rEndDate - rStartDate)) - (r.price * (rEndDate -
rStartDate) * (c.discount / 100))) INTO price
      From customer c natural join rooms r
      where c.cID = cID and c.hID = hID and c.name = name;
END
CALL ComputeTotalPrice(:customerId, :hotelId, :customerName, @price)
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `rateHotel`(IN hid INT, IN numstars
INT, IN review VARCHAR(500) CHARSET utf8)
BEGIN
      INSERT INTO rating (hID, rating, review)
      VALUES(hid, numstars, review);
END
CALL rateHotel('$hotelID', '$stars', '$areview')
```

Testing Manual

15 Functional Requirements

1) Reserve a Room

1. Navigate to your localhost
2. Click on “Hotel Reservation” or “Rent A Room” from the “Hotel Reservation” dropdown menu
3. Enter in the start date (format: YYYY-MM-DD, e.g. 2014-12-11)
4. Enter in the end date (format: YYYY-MM-DD, e.g. 2014-12-18)
5. Select a location from the dropdown menu
6. Click “Check for available rooms!” button
7. Make your selection choice of hotel and room number from the dropdown
8. Enter in your name, address, and credit card number
9. If applicable, select discount
10. If you will be smoking in your room, select “smoking”. Otherwise, select “non-smoking”

2) Book a Valet or Non-Valet Parking

11. Select your preferred choice of parking--"valet" or "non-valet".

Results: Confirmation page should show. If reserving for more than 2 weeks (14 days) then complimentary parking is included and option 11 is not applicable.

3) Cancel Reservation

Using your entry values from "Reserve a Room," follow these instructions:

1. Navigate to your localhost
2. Mouse over "Hotel Reservation" and select "Cancel Reservation" from the dropdown menu
3. From the three dropdown menus, select your hotel's location, the hotel, and room number
4. Enter in your reservation start and end date (format: YYYY-MM-DD, e.g. 2014-12-11)
5. Click "Submit Request"

Result: "Hip hip! Hooray!" confirmation page should appear.

4) Leave Hotel Ratings

1. Navigate to your localhost
2. Mouse over "Hotel Reservation" and select "Leave Feedback" from the dropdown menu
3. Select your hotel and your choice of a 5-star rating
4. If you would like, type in your comments/feedbacks
5. Click "Submit Feedback"

Result: Thank you message displays.

5) View Hotel Ratings

1. Navigate to your localhost
2. Mouse over "Hotel Reservation" and select "Hotel Ratings" from the dropdown menu

Result: All hotel ratings should be displayed.

Filtering Hotel Ratings

1. Use the "Friendly Filter" to choose which hotel ratings you want to view
2. Click "Submit" to filter

Result: Results should be filtered accordingly.

6) Compare/Contrast Hotel Prices

1. Navigate to your localhost
2. Mouse over "Hotel Reservation" and select "Compare Prices" from the dropdown menu

Result: Prices to hotels are listed.

Filtering Prices by Rooms

1. Use the “Friendly Filter” to choose the types of rooms you want to view
2. Click “Submit” to filter

Result: Prices for choice are listed.

7) Manager--Registration

eID	hID	name	position	salary
2	1	John	Manager	60000
2	2	Steven	Manager	80000
2	3	Lee	Manager	65000
2	4	Thomas	Manager	50010
2	5	Jason	Manager	105000
2	6	Stevie	Manager	105000
2	7	Frankie	Manager	105000

1. Navigate to your localhost
2. Mouse over “Manager Services” and select “Register” from the dropdown menu
3. Using the employee relation, choose a manager and use their credentials for registration

Result: Successful registration will redirect to Manager’s Homepage. Unsuccessful registration will prompt a retry.

8) Manager--Login

1. Navigate to your localhost
2. Mouse over “Manager Services” and select “Login” from the dropdown menu
3. Using the credentials from “Manager--Registration”, enter in the username and password

Result: Successful login will redirect to the Manager’s Homepage. Unsuccessful login will prompt a retry.

9) Manager--Logout

If not logged in already, start with step 1. If logged in, being at step 4.

1. Do “Manager--Login”
2. Click “Logout”

Result: Successful logout will redirect to JAT Hotel Reservation Homepage.

10) Manager--View Revenue

1. Do “Manager--Login”
2. Click “View Revenue”

Result: Total revenue will display.

Filter by Rooms

1. Select filter of your choice
2. Click “Submit”

Result: Revenue should update accordingly.

11) Manager--Charge Customer

1. Do “Manager--Login”
2. Click “Charge Customer”
3. Select Customer to charge from the dropdown
4. Click “Submit”

Result: Successful statement displays.

12) Manager--Reassign Customer’s Room

1. Do “Manager--Login”
2. Click “Assign New room to Customer”
3. Select “Smoking” or “Non-Smoking”
4. Click “Submit”

Result: Customer’s room is updated.

13) Manager--Cancel Customer’s Reservation

1. Do “Manager--Login”
2. Click “Cancel Customer’s Reservation”
3. Select customer to cancel
4. Click “Submit”

Result: Customer’s reservation is canceled.

14) Manager--Check Available Rooms

1. Do “Manager--Login”
2. Click “Check Available Rooms”
3. Enter in the start and end date of your choice (format: YYYY-MM-DD, e.g. 2014-12-11)
4. Select view choice by Date or Month

Result: Available rooms will be listed by their room number.

15) Manager--Hire an Employee

1. Do “Manager--Login”
2. Click “Hire an employee”
3. Enter in their name, position, and salary
4. Click “Add this employee”

Result: Success page loads.

16) Manager--Fire an Employee

1. Do “Manager--Login”
2. Click “Fire an employee”
3. Select Employee from the dropdown menu
4. Click “Fire this employee”

Result: Success page loads.

17) Manager--Transfer an Employee

1. Do “Manager--Login”
2. Click “Transfer an employee”
3. Select Employee from dropdown menu
4. Select destination Hotel to transfer the chosen Employee to

Result: Success page loads.

Archiving

Using MySQLWorkbench, run the following queries:

```
-- Replace “YYYY-MM-DD” with the year, month and day
call jat_reservation.archive('YYYY-MM-DD');
SELECT * FROM jat_reservation.parkingarchiving;
SELECT * FROM jat_reservation.customerarchiving;
```

Result: parkingarchiving and customerarchiving should now contain data from parking and customer relations where updateAt < input date.

Key Constraint and Foreign Key Constraint Violation

In our schema, we made sure that we do not run into any of these violations by imposing primary key constraints and foreign key constraints.

Please use MySQLWorkbench to execute the queries.

customer

Primary Key: customer(cID, hID)

Key constraint will be violated when we attempt to add new (cID, hID) which matches an existing (cID, hID) in our customer table.

```
insert into customer(cID, hID, rID, name, address, ccNo, smoker,
rStartDate, rEndDate, discount)
values(1, 1, 10, "test", "1234 address here, San Jose, CA 95116",
123412341234, 1, 2014-12-11, 2014-12-12, 0);
```

Foreign Key: customer(rID, hID) references rooms(rID, hID)

Foreign Key constraint will be violated when we try to add (rID, hID) into customer table, but if the corresponding (rID, hID) is not available in rooms table, then foreign key constraints will be violated.

```
insert into customer(cID, hID, rID, name, address, ccNo, smoker,
rStartDate, rEndDate, discount)
values(6, 11, 11, "test", "1234 address here, San Jose, CA 95116",
123412341234, 1, "2014-12-11", "2014-12-12", 0);
```

---

employee

Primary Key: employee(eID, hID)

Key constraint will be violated when we attempt to add new (eID, hID) which matches an existing (eID, hID) in our employee table.

```
insert into employee(eID, hID, name, position, salary)
values(1, 1, "David", "Manager", 100000);
```

Foreign Key: employee(eID, hID) references hotels(hID)

Foreign Key constraint will be violated when we try to add (hID) into employee table, but if the corresponding (hID) is not available in the hotels table.

```
insert into employee(eID, hID, name, position, salary)
values(1, 15, "David", "Manager", 100000);
```

---

hotels

Primary Key: hotels(hID)

```
insert into hotels values(1, "Ceasars Palace", "San Francisco", 30);
```

---

managerlogin

Primary Key: managerlogin(username, employee_eID, employee_hID)

```
insert into managerlogin(username, password, employee_eID, employee_hID)
values ("john", "somethingelse", 2, 1);
```

Foreign Key: managerlogin(employee_eID, employee_hID) references employee(eID, hID)

```
insert into managerlogin(username, password, employee_eID, employee_hID)
values ("john", "somethingelse", 10, 1);
```

---

parking

Primary Key: parking(pID, hID, cID)

```
insert into parking values(1, 1, 1, 1, CURDATE());
```

Foreign Key: parking(hID, cID) references customer(cID, hID)

```
insert into parking values(1, 1000, 1, 1000, CURDATE());
```

---

parkingarchiving

In this case, we are assuming that we are trying to insert the values after the tables have been populated.

Primary Key: parkingarchiving(pID, hID)

```
insert into parkingarchiving(pID, hID, valet, cID, updatedAt)
values(1, 1, 1, 2, CURDATE());
```

Rating

Primary Key: rating(ratingID)

```
insert into rating(ratingID, hID, rating, review)
values(2, 1, 1, "This place is a dump!");

insert into rating(ratingID, hID, rating, review)
values(2, 1, 4, "This hotel is Great!!!!");
```

Foreign Key: rating(hID) references hotels(hID)

```
insert into rating(ratingID, hID, rating, review)
values(1, 20, 4, "Nice place!");
```

rooms

Primary Key: rooms(rID, hID)

```
insert into rooms(rID, hID, smoking, price)
values(1, 1, 0, 200);
```

Foreign Key: rooms(hID) references hotels(hID)

```
insert into rooms(rID, hID, smoking, price)
values(1, 15, 0, 200);
```