***Hotel Management Database System***

*For WolfVillas*

*CSC 440 Database Systems*

*Project Report #1*

*Joanna Pollard, Mohamad Saleh, Anusha Balaji, Alex Harrell*

*Please include one or more separate pages with a complete list of your assumptions about the project domain (as described in the* [*narrative*](http://courses.ncsu.edu/csc440/lec/001/wrap/narrative.pdf)*). Your entire report can be graded only with respect to your assumptions, and is not going to be graded in their absence.*

Assumptions on attributes:

* Staff attribute “**Hotel currently serving**” will have the value of “hotel ID”
* **Customer ID** assigned to customers is unique across the hotel chain and hence, customers can be identified with their ID across the whole hotel chain.
* **Hotel Room Number**: Every room in a hotel has a unique room number. So, we cannot have an economy room with number 1 and a deluxe room with number 1.
  + Also, we are going to assume **Economy rooms start with 1**
  + Deluxe rooms start with 2
  + Executive Suite start with 3 and
  + Presidential Suite start with 4
* Possible **services availed** by customers during stay **and their respective codes** include:
  + phone services (**1**), restaurant services (**2**), laundry services (**3**), and in-store/in-hotel purchases (**4**).
* We are going to use integers to represent **room types**. Possible room types in the hotel chain and their corresponding codes:
  + Economy (represented by 1)
  + Deluxe (represented by 2)
  + Executive Suite (represented by 3)
  + Presidential Suite (represented by 4)
* Customers with multiple reservations with the same check-in date must have them all in the same hotel. In other words, they can’t have concurrent reservations at different hotels.
* Possible categories for staff/possible values for job titles:
  + Manager
  + Front Desk Representative
  + Room Service Staff
  + Catering Staff
  + Laundry Staff
* Nightly rate is static and dependant only on the room number
* Presidential suite is assigned one room service staff and one catering staff
* The front desk representative handles all “billing staff” duties
* The customer associated with a reservation pays the bill.

Assumptions on design:

* Check in/Check out details fall under “**Reservation**” entity
* Front desk representatives handle **billing**; other staff can only “charge”/ add to the bill - they can’t “bill” customers.
* Customers **cannot** book a hotel/reserve a room. They have to call or meet the Front Desk Representative for a reservation as customers don’t have access to staff and room availability information.
* **Hotel managers** are able to access any data that front desk representatives and other staff can access. They are also able to complete tasks related to that data, such as bill a customer or make a reservation. This is in addition to what they can already access.

1. (10 points) Describe the problem concisely (fewer than about 250 words). Why is a database (as opposed to, say, a simple set of files) a good idea for this task? Submit your description and explanation.

WolfVillas is a hotel chain with various hotels, customers, and users. We will need to design and build a reliable database management system for WolfVillas to maintain, continually update, and track hotel availabilities and customer reservations. There are a few reasons why a database is a good idea for managing this task:

1. Information about hotel reservations and room availabilities is constantly changing. With users all over the country, a file system is insufficient for this task because users need all the latest information in one place at once. A database will allow WolfVillas management to monitor hotels easily, with updated and consistent information across the whole system.
2. They could also use database system to quickly and easily aggregate data for reporting, allowing the management to see each hotel’s occupancy, hire new, and monitor existing staff.
3. With several staff members working in the hotel and charging customers for different services, a file system would be inefficient as it would be difficult to know whether a customer's total bill accurately reflects all the costs. A database management system performs basic input validation like preventing duplicate entries and maintaining consistency.
4. A file system does not have security abilities that a database would have. If a customer is represented in a file, anyone who has access to it can see/modify the customer's payment information and SSN. A database allows us to restrict access to data depending on the user. So, users like catering staff won’t have access to a customer’s SSN.
5. (10 points) Describe the intended classes of users of your database system. (One possible class of users might be "staff who do billing".) Submit your descriptions.

**WolfVillas Company Management** are the users that oversee all hotel chains. They have access to each chain’s database and can add or modify data for staff and customers. They will also be able to view stats on specific hotels like occupancy information, specific staff and assigned customers, etc.

**Hotel Managers** are in charge of all other staff at the hotel. They will have access and can modify all information other staff have access to. They also can add new staff and modify the existing staff in the database.

**Front desk representatives** are users of the system that enter, lookup and modify customer information and reservations in the database. At the end of a stay, the front desk representative bills the customer.

**Other Staff (laundry, catering, room service)** will carry out services as requested by customers and other hotel staff. They will only have access to bill charges they can control for a customer’s reservation. In other words, room service staff can’t see nor bill for catering services. Catering and Room Service staff can also see list of customers they are responsible for (task 4, last operation)

**Customers** have reservations for a specific room. They can view and edit their own information, as well as see the information about their reservations, including any additional service charges. They can also view their past reservations in any WolfVillas hotel.

1. (10 points) Identify 5 main "things" about which you will need to keep information, and the information you will need to keep. Submit the names of the 5 "things" and the information you need to keep about them.

* Customer - Customer ID, name, gender, phone number, address, email address
* Staff - Staff ID, SSN, name, age, gender, job title, department, phone number, address, hotel ID
* Reservation - start date, end date, check in time, check out time, total bill, customer ID, room number, hotel ID, services availed (phone, laundry, restaurant, etc), SSN of payer, billing address, payment method, card number
* Room - Room number, room category, max occupancy, nightly rate, availability
* Hotel - Hotel ID, name, address, phone number, ID of hotel manager

1. (10 points) Describe realistic situations where using your database system will require handling any two of the operations (see Tasks and Operations in the [**narrative**](http://courses.ncsu.edu/csc440/lec/001/wrap/narrative.pdf)); in each situation, you may consider one or more operations. (The assignment is to describe a *total* of two operations, using a *total* of either one or two situations.) Describe each situation in about 50 words. Submit your descriptions.

* The manager has hired a new front desk representative and needs to enter the new hire’s information into the database, specifying his type of employment. On the new hire’s first day, he must be able to register customers and process reservations.
* A customer orders room service. The staff member that receives this request must be able to update the customer's bill and record what the customer purchased and what he was charged for. At the time that the customer must pay his bill, the front desk representative should be able to show the customer the charges that make the total bill.

1. (80 points) Sketch the APIs (application program interfaces; for this project, an API of an operation is just inputs and outputs for the operation) required for each of the operations listed in Tasks and Operations in the [**narrative**](http://courses.ncsu.edu/csc440/lec/001/wrap/narrative.pdf). (In some cases the output will be just a confirmation.) Submit the APIs. Note: A lot of points will be taken off if you miss some of the operations in the [narrative](http://courses.ncsu.edu/csc440/lec/001/wrap/narrative.pdf).

**Information Processing**

* **Relevant to Customer**

*addNewCustomer*(name, gender, phoneNumber, address, cardNumber)

* Return Customer ID upon success, null otherwise

*updateCustomer*(name, gender, phoneNumber, address, cardNumber)

* Return confirmation
* Will only be updated if value is not NULL

*getCustomer*(customerID)

* Return Customer: name, gender, phoneNumber, address, cardNumber

deleteCustomer(customerID)

* Return confirmation: true for success / false for failure

*sendBill(*customerID, totalBillAmount, servicesAvailed[]*)*

* return confirmation: true for success / false otherwise

* **Staff**

*addStaff(*SSN, firstName, lastName, age, gender, title, department, phoneNumber, address*)*

* return confirmation - true if success and false otherwise

*updateStaff*(SSN, firstName, lastName, age, gender, title, department, phoneNumber, address)

* return confirmation
* Will only be updated if no value is NULL

*deleteStaff*(staffID)

* return confirmation: true - success / false - failure

*viewStaff*(staffID)

* return name, age, gender, title, department, phone number, phone address, hotelID of specific staff member
* **Room**

*addRoom(*hotelID,roomNumber, roomCategory, maxOccupancy, nightlyRate, availability*)*

* Adds the given room to hotel
* Returns true upon success and false otherwise (if no space)

*updateRoom(*hotelID, roomNumber, newRoomCategory, newMaxOccupancy, newNightlyRate, availability*)*

* Updates the given room in hotel
* Returns true upon success and false otherwise (room doesn’t exist yet in database)

*deleteRoom(hotelID, roomNumber)*

* Deletes the given room in hotel
* Current reservations are billed as is
* Returns true upon success and false otherwise (room doesn’t exist in database)

*isRoomAvailable(*checkInDate, checkOutDate, roomNumber, hotel ID*)*

* return boolean about whether room in a given hotel is occupied for the given set of dates

*assignStaffToPresidentalSuite*(roomNumber, staffID, staffDepartment)

* return confirmation: true - success / false - if no dedicated room service and catering staff are available

*getRoomInfo(*hotelID, roomNumber)

* Returns a Room with the given roomNumber in the hotel with the given hotelID
* Return null upon failure

*getRoomsGivenType*(hotelID, roomType)

* Returns a set/list of (***available*** + **unavailable**) rooms of the given type in the hotel
* For presidential suites, caller should make sure dedicated staff is available

*getRoomsGivenOccupancy*(hotelID, occupancy)

* Returns a set/list of rooms in the hotel with occupancy greater than or equal to given occupancy

*getRoomsAvailable*(hotelID)

* Returns a set/list of ***available*** rooms in the given hotel

*getRoomsGivenBudget*(hotelID, maxBudgetPerNight)

* Returns a set/list of rooms (available and unavailable) with nightly rate <= to the given rate

*assignRoomToReservation(customerID, hotelID, roomNumber)*

* Identifies and retrieves reservation with the given parameters
* Adds the given room to the retrieved reservation
* The existing one is modified to reflect the room assignment
* Room is marked as unavailable
* Returns true upon success and false upon failure

*releaseRoom(*hotelID, roomNumber*)*

* Room is marked available
* Returns true upon success and false upon failure
* **Reservation**

*createReservation*(roomNumber, customerID, startDate, endDate, checkInTime, checkOutTime)

* Generates a reservation as well as a billing account
* Return confirmation: true if success and false otherwise

*modifyReservation*(roomNumber, customerID, startDate, endDate, checkInTime, checkOutTime)

* return confirmation

*deleteReservation*(customerID, roomNumber, hotelID)

* return confirmation

*getReservation(*customerID, roomNumber, hotelID*)*

* return roomNumber, customerID, startDate, endDate, checkInTime, checkOutTime of specific reservation or a new one if none exist with the given parameters
* A new billing account is generated if a new one is generated

**Maintain Services Availed**

*chargeForService*(serviceType, amountCharged, date, time, staffID, customerID, roomNumber, checkInDate)

* Creates an **Invoice** (entity) with the given service details
* Description for the invoice would contain date and time service was availed and staffID
* Would also update total bill in the reservation
* Returns confirmation: true if success / false otherwise

*delete*Service(serviceType, chargeAmt, date, time, staffID, customerID, roomNumber, checkInDate)

* Used in cases where a customer was wrongly charged for a service they didn’t avail
* Returns confirmation: true - success; false otherwise (service doesn’t exist with the specified amount)

*modify*Service(serviceType, chargeAmt, date, time, staffID, customerID, roomNumber, checkInDate)

* Setter for service, but the service must have already been added to the reservation
* Return true if success modifying in database and false otherwise

*getServicesAvailed*(customerID, roomNumber, checkInDate)

* return array of services availed in the specified reservation

**Maintain Billing Accounts**

*isRoomAvailable(*checkInDate, checkOutDate, roomNumber, hotel ID*)*

* Return boolean about whether room in a given hotel is occupied for the given set of dates

*generateBillingAccount(*customerID, roomNumber, visitLength)

* Creates a bill for the customer based on roomNumber and visit length (check-out time - check-in time)

modifyBillingAccount(customerID, roomNumber, visitLength)

* Used to modify a reservation when either the roomNumber or visit length changes. Bill is recalculated and updated

*getTotalBill*(customerID, roomNumber)

* return total bill = bill + charges for services availed

**Reports**

*getOccupancyByHotel(*hotelID)

* Returns an array of size 5, with percentages (0-100%) rounded to the hundredth for each type of room in the hotel *chain* where index 0 has occupancy for Economy rooms, index 1 is for Deluxe rooms, 2 is for Executive Suite, 3 is for Presidential Suite, and **4 is for total occupancy**
* Example: Hotel 1 with ID = 1 has only 3 types of rooms - economy, deluxe, and executive suite and data is as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Room type | Occupied Rooms | Vacant Rooms | Total Rooms | % Occupied |
| Economy | 5 | 2 | 7 | 5/7 = 71.43% |
| Deluxe | 3 | 2 | 5 | 3/5 = 60% |
| Executive Suite | 2 | 1 | 3 | 2/3 = 66.67% |
| Total | 10 | 5 | 15 | 10/15 = 66.67% |

* The output from this method call would be: [71.43, 60.00, 66.67, 0, 66.67 ]

*getOccupancyByRoomType(*roomType)

* Returns a percentage (0-100) rounded to the nearest hundredth for the occupancy of a specific room type in **all** the hotels in WolfVillas
* **roomType is an integer 1 - 4** where 1 is for Economy rooms, 2 is for Deluxe rooms, 3 is for Executive Suite, 4 is for Presidential Suites

*getOccupancyByDateRange*(startDate, endDate, checkInTime, checkOutTime)

* Returns a percentage (0-100) rounded to the nearest hundredth for the occupancy in **all** the hotels of WolVillas during the given period.
* startDate and endDate are Strings in the following format (MMDDYYYY)
* CheckInTime and CheckOutTime are integers 0-23 representing every hour in the day

getOccupantsAndTheirOccupancy(date, time)

* Returns Customers and the percentage of rooms occupied by them for every hotel in WolfVillas.

*getStaffByJobTitle(*JobTitle)

* Returns information on staff that have the given role/job title **for every hotel** in WolfVillas via a list. Returns staff ID, name, gender, job title, department, phoneNumber, address, hotelID **for every hotel** in a separate list.

*getStaffByDepartment(*Department)

* Returns information on staff that belong in the given department **for every hotel** in WolfVillas via a list. Returns staff ID, name, gender, job title, department, phoneNumber, address, hotelID **for every hotel** in a separate list.

getCustomersForStaff(staffID)

* Ensures the given staffID belongs to a catering staff or room service staff member
* If true, return a list of Customers that are serviced by this personnel or null otherwise

1. (80 points) Give short descriptions (no more than 50 words each) of the views of the data that correspond to the intended classes of users, one view per user class. Each view should reflect all and only the **data** (rather than operations) in the database that is relevant to all operations for the given class of users. For example, the hotel-receptionist view may reflect all check-in and check-out information about the guests, and may represent the entire database as just one table for guests, rooms, and check-ins. You do not need to give all the details in your descriptions. Be careful not to include users' operations (from Tasks and Operations in the [**narrative**](http://courses.ncsu.edu/csc440/lec/001/wrap/narrative.pdf)) in your views. Submit your descriptions, one per user class.

**Customer**

Customers have the ability to see their reservation information and bill for their stay. This view includes:

* The reservation: the hotel name, address, phone number, the nightly rate for their room, the maximum allowed occupancy, the room number, the category of the room, the check in and check out dates and their customer ID.
* Details regarding their bill, including a table for the list of charges on their reservation.

**Staff (Laundry, Catering and Room Service)**

The staff should be able to view and charge for the services they have control over and see their own information. For example, catering staff can view and charge only for restaurant services and not laundry services.

* Each reservation’s table of charges, but they can see only charges they have control over for security reasons.
* View personal information
* List of customers they are responsible for providing services (only catering and room service staff)

**Front Desk Representative (FDR)**

FDR’s have the ability to view data relevant to the room, reservation and billing information as they bill customers for their stay. The FDR can view:

* Regular staff viewas described above.
* Rooms in the hotel, their category, occupancy, rates and availability. If not available, states which customers are assigned to them.
* Reservations table: includes customer billing information, services availed during current reservation, and payment information.
* Customers table: personal information, past and current reservations, services availed so far.

**Manager**

In addition to the staff and FDR view, Managers can access data related to staff. The manager view includes:

* Regular staff and front desk representatives views as described above - reservations, customers’ billing account information.
* Staff Table: information about staff ID, name, age, gender, job title, department and contact information.

**WolfVillas Company Management**

Company management can access all of the hotels individually and together, staff information, Customer reservations, etc. Specifically, they can access:

* Hotels: ID, manager, staff information, other information (name, address, phone number), and occupancy
* Staff: staff ID, personal and contact information, hotel and role they currently serve, and customers they are responsible for.
* Rooms: room information (room number, category, max occupancy, rate, availability), occupancy according to room type
* Customers: personal information (ID, name, address, contact information, SSN)
* Reservations: information on a particular reservation as well as a group of them belonging to a particular customer, room type, or a period, payment information

1. (160 points) Construct local E/R diagrams, one for each view in item 6. In each diagram, you need to reflect all database information that is relevant to all operations for the given class of users. (See item 6 above and the [**narrative**](http://courses.ncsu.edu/csc440/lec/001/wrap/narrative.pdf)). Submit all the diagrams.

**[See End of Document]**

1. (40 points) Document the local E/R diagrams. Highlight any design decisions - specifically, describe why you have the entities and relationships that you do. Submit your documentation.

* Staff can be Manager, Front Desk Representative, or general staff
* The following is how we chose to identify each entity set:
  + Customers are identified by their customer ID. This ID is central to our design and multiple entity sets rely on it.
  + Reservation is a weak entity set that relies on multiple identifiers from other entities such as customer ID, room number and check-in date.
  + Room is identified by its room number. Multiple hotels may have the same room number in the franchise. Therefore, we differentiate it by coupling check-in date and room number as a key.
  + Staff is identified by their staff ID. This allows multiple staff with the same name to exist in the database.
  + Hotel is identified by its hotel ID. This allows management to have multiple hotels in any given area.
* Front desk representatives and managers . All other employees at the hotel are considered under the generic entity staff.
* Relationship types and constraints in the design:
  + Staff adds charges to the reservation
    - Multiple staff members can add charges to any and all reservations in the database
  + Front desk representative manages the reservation
    - Multiple front desk representatives can manage all reservations.
  + Customers have a reservation
    - Customers may have multiple reservations in the wolfvillas system, but each reservation may have only one customer.
  + Room is assigned to a reservation
    - A room may be assigned to multiple reservations, but each reservation may have only one room.
  + Manager manages a hotel
    - Only one manager may manage one hotel at a time.
  + A hotel leases its rooms
    - A hotel contains many rooms and each room can only be assigned to one hotel.
  + A hotel employs staff
    - A hotel may employ many staff, but each staff may only be employed by one hotel.

9. (80 points) Derive a local relational schema from each of the above local E/R diagrams (see item 7). Please do not be creative in your translations - points will be taken off if your translations from E/R diagrams to relation schemas cannot be done mechanically as explained in the textbook. Submit all the relational schemas - one per user class.

**Customer**

*customer*(customerId, name, SSN, email, address, phoneNumber, gender, creditCard, billingAddress)

*reservation*(customerID, roomNumber, checkInDate)

*room*(number, rate, category, availability, occupancy)

*hotel*(hotelId, name, address, phoneNumber)

**Staff**

*staff*(staffId, name, address, age, dept, phoneNumber, gender, jobTitle, SSN, hotelId, salary)

*charges*(customerID, staffID, description, date, roomNumber, checkInDate, amount)

*reservation*(roomNumber, customerID, checkInDate)

**Manager**

*staff*(staffId, name, address, age, department, phoneNumber, gender, jobTitle, SSN, salary)

*hotel*(hotelId, address, name, phoneNumber)

*manager*(staffId)

*managerManages*(staffID, hotelID, address, name, phone)

*frontDeskRepresentative*(staffId)

fdrManages(checkInDate, roomNumber, customerID)

*customer*(customerId, name, email, address, phoneNumber, gender, SSN)

*reservation*(roomNumber, customerID, checkInDate, checkoutDate, balance)

*rooms*(number, rate, availability, occupancy, category)

*presidentialSuite*(number, roomStaffId, cateringStaffId)

*charges*(customerId, roomNumber, checkInDate, description, date, staffID, amount)

**Front Desk Representative**

*staff*(staffID, name, address, age, dept, phoneNumber, gender, jobTitle, SSN, hotelId)

*frontDeskRepresentative*(staffID)

*manage*(staffID, roomNumber, customerID, checkInDate)

*customer*(customerId, name, email, address, phoneNumber, gender)

*reservation*(roomNumber, customerID, checkInDate, checkoutDate, balance)

*room*(number, rate, availability, occupancy, category)

*presidentialSuite*(number, roomStaffID, cateringStaffID)

*charge*(roomNumber, customerID, checkInDate, description, amount, currentDate, staffID)

**WolfVillas Company Management**

*hotel*(hotelID, address, name, phoneNumber, managerId)

*staff*(staffID, name, address, age, dept, phoneNumber, gender, jobTitle, SSN, hotelId)

*employs*(staffID, hotelID)

1. (40 points) Document the local relational schemas. Highlight any design decisions - specifically, explain why you decided to make the relations you did and how each entity and relationship in the E/R diagram is captured in the relational schema. Submit your documentation.

The E/R diagrams were translated into relational schemas by:

* Transferring each entity set into a relation
* We used the E/R approach in regards to subclasses.
  + For presidential suite, this allows us to quickly see which rooms staff are assigned to, and not have a lot of repeated staff information.
  + For Manager, it was not necessary to create another relation schema because there weren’t more attributes, just an extra relationship allowing them to manage the hotel information
  + For Front desk representative, again there was not a necessity to make another relation schema because the entity just had an extra relationship that allowed them to manage reservations.
* We transferred the “Add Charge” relationship because we thought it would be a helpful relation to have - for a customer to see a list of their charges. We used the IDs from both of the entities that it is a relationship between, as well as the attributes on the relationship.