Ranishree Anegundi
ID# 99795021

SWE 565 1 Advanced Database Systems
Hotel Management System (MongoDB)

Table of Contents

Ranishree Anegundi
ID# 99795021

1. Introduction

The Hotel Management System (HMS) is a comprehensive software solution designed to streamline and automate various aspects of hotel operations. This document outlines the database design for the HMS, focusing on MongoDB as the chosen database system and explaining the rationale behind this choice.

2. Database System

2.1 Database Management System (DBMS)

For the Hotel Management System, I have opted for a NoSQL database model, specifically MongoDB. MongoDB is a document-oriented database that offers flexibility, scalability, and efficiency for handling large volumes of unstructured or semi-structured data.

3. Database Design

3.1 Collection and Documents

In MongoDB, data is organized into collections, which are analogous to tables in a relational database. Each document within a collection represents a record and can have a flexible schema, allowing for dynamic and evolving data structures.

The key collections in the HMS MongoDB database include:

Staff: Documents with StaffID, FirstName, LastName, Designation, Email, and other details about hotel staff.
Rooms: Documents containing RoomNumber, RoomType, Rate, Status, and other details about the hotel rooms.
Guests: Documents with fields like GuestID, FirstName, LastName, ContactNumber, Email, and other guest-related information.

- **Reservations**: Documents with ReservationID, GuestID (linked to Guests collection), FromDate, ToDate, and other reservation-related information.

Bookings: Records of guest bookings, including check-in and check-out dates. It has Services as a sub-key.

- **Services**: Documents with ServiceID, ServiceName, Type, Amount, and other information related to hotel services.

Billing: Records of PaymentDate, TotalAmount and Status.

The relationships between these entities are modeled to represent the interactions within the hotel management system. For example, a reservation is associated with a specific guest and a particular room, creating links between the Guests, Rooms, and Reservations tables.

3.2 Schema Flexibility

MongoDB's schema flexibility allows for easy adaptation to changing business requirements. New fields can be added to documents without affecting existing data, making it suitable for a dynamic environment like a hotel management system.

4. Rationale for Database Choice

The selection of MongoDB as the database system for the Hotel Management System is based on several key considerations:

Scalability: MongoDB's horizontal scaling capabilities make it well-suited for managing large volumes of data and accommodating the potential growth of the hotel business.

Document-Oriented Model: The document-oriented approach simplifies representing complex relationships between entities, such as guests, rooms, and reservations, by storing related information in a single document.

JSON-Like Documents: MongoDB's use of JSON-like documents makes it intuitive for developers and facilitates seamless integration with modern web development frameworks and languages.

Rich Query Language: MongoDB's query language allows for powerful and expressive queries, enabling efficient retrieval of specific data patterns relevant to hotel management operations.

Ranishree Anegundi
ID# 99795021

5. Queries

5.1) Query items under a specific category.

5.1.1) Query rooms available based on availability status.

```
HotelManagementSystem> const availableRooms = db.Rooms.find({ Status: 'Available' }).toArray();

HotelManagementSystem> availableRooms.forEach(room => {
...   print(`Room Number: ${room._id}, Type: ${room.Type}, Capacity: ${room.Capacity}, PricePerDay: ${room.PricePerDay}`);
... });
Room Number: 101, Type: Standard, Capacity: 2, PricePerDay: 100
Room Number: 201, Type: Deluxe, Capacity: 3, PricePerDay: 150
Room Number: 202, Type: Standard, Capacity: 2, PricePerDay: 120
Room Number: 104, Type: Mountain View, Capacity: 3, PricePerDay: 160
Room Number: 105, Type: Executive, Capacity: 2, PricePerDay: 220
Room Number: 403, Type: Standard, Capacity: 2, PricePerDay: 110
Room Number: 106, Type: Suite, Capacity: 4, PricePerDay: 210
Room Number: 203, Type: Ocean View, Capacity: 2, PricePerDay: 190
Room Number: 204, Type: Mountain View, Capacity: 3, PricePerDay: 180
Room Number: 205, Type: Standard, Capacity: 2, PricePerDay: 110
Room Number: 206, Type: Suite, Capacity: 4, PricePerDay: 220
Room Number: 110, Type: Ocean View, Capacity: 2, PricePerDay: 190
Room Number: 111, Type: Mountain View, Capacity: 3, PricePerDay: 180
Room Number: 209, Type: Deluxe, Capacity: 3, PricePerDay: 170
Room Number: 112, Type: Standard, Capacity: 2, PricePerDay: 110
```

5.1.2) Query managers in the Staff table.

```
HotelManagementSystem> const managers = db.Staff.find({ Designation: 'Manager' }).toArray();

HotelManagementSystem> managers.forEach(manager => {
...   print(`StaffID: ${manager._id}, FirstName: ${manager.FirstName}, LastName: ${manager.LastName}, Designation: ${manager.Designat
ion}, Email: ${manager.Email}`);
... });
StaffID: 656e9a33d29717bc7784d3ed, FirstName: John, LastName: Doe, Designation: Manager, Email: john.doe@examplehotel.com
StaffID: 656e9a33d29717bc7784d3f1, FirstName: Michael, LastName: Brown, Designation: Manager, Email: michael.brown@downtownsuites.com
```

Ranishree Anegundi
ID# 99795021

5.1.3) Query to get the total number of guests, rooms booked on the specified date.

```
HotelManagementSystem> const result = db.Bookings.aggregate([
...   {
...     $match: {
...             CheckInDate: ISODate('2023-01-10')
...     }
...   },
...   {
...     $group: {
...       _id: null,
...       totalGuests: { $sum: '$NumOfGuests' },
...       totalRoomsBooked: { $sum: 1 }
...     }
...   },
...   {
...     $project: {
...       _id: 0,
...       totalGuests: 1,
...       totalRoomsBooked: 1
...     }
...   }
... ]).toArray();

HotelManagementSystem>

HotelManagementSystem> result.forEach(doc => {
...   print('totalGuests:' + doc.totalGuests),
...   print('totalRoomsBooked:' + doc.totalRoomsBooked)
...   });
totalGuests:4
totalRoomsBooked:2
```

Ranishree Anegundi
ID# 99795021

5.2) Query data between different dates.

5.2.1) Query Rooms Booked, Guest Details Between CheckIn and CheckOut Dates.

```
HotelManagementSystem> db.Bookings.aggregate([
...   {
...     $match: {
...       $or: [
...         {
...           $and: [
...             { CheckInDate: { $gte: checkInDate, $lte: checkOutDate } },
...             { CheckOutDate: { $ne: null, $gte: checkInDate, $lte: checkOutDate } }
...           ]
...         },
...         {
...           $and: [
...             { CheckInDate: { $lte: checkInDate } },
...             { $or: [{ CheckOutDate: { $gte: checkInDate } }, { CheckOutDate: null }] }
...           ]
...         },
...         {
...           $and: [
...             { CheckInDate: { $lte: checkOutDate } },
...             { $or: [{ CheckOutDate: { $gte: checkOutDate } }, { CheckOutDate: null }] }
...           ]
...         }
...       ]
...     }
...   },
...   {
...     $lookup: {
...       from: 'Rooms',
...       localField: 'RoomNum',
...       foreignField: 'RoomNum',
...       as: 'roomDetails'
...     }
...   },
...   {
...     $lookup: {
...       from: 'Guests',
...       localField: 'GuestID',
...       foreignField: 'GuestID',
...       as: 'guestDetails'
...     }
...   },
...   {
...     $project: {
...       _id: 0,
...       BookingID: 1,
...       RoomNum: 1,
...       CheckInDate: 1,
...       CheckOutDate: 1,
...       NumOfGuests: 1,
...       roomDetails: { $arrayElemAt: ['$roomDetails', 0] },
...       guestDetails: { $arrayElemAt: ['$guestDetails', 0] }
...     }
...   }
... ]);
```

Ranishree Anegundi
ID# 99795021

Output:

```
[
  {
    RoomNum: 101,
    CheckInDate: ISODate("2023-01-10T00:00:00.000Z"),
    NumOfGuests: 2
  },
  {
    RoomNum: 201,
    CheckInDate: ISODate("2023-01-10T00:00:00.000Z"),
    NumOfGuests: 2
  }
]
```

5.3) Query data for a specific item.

5.3.1) Query Room Details based on Room Number 101.

```
HotelManagementSystem> db.Rooms.findOne({ _id: 101 });
{
  _id: 101,
  Type: 'Standard',
  Capacity: 2,
  PricePerDay: 100,
  Status: 'Available'
}
HotelManagementSystem>
```

Ranishree Anegundi
ID# 99795021

5.3.2) Query Booking and Guest Details based on GuestID and RoomNumber.

```
HotelManagementSystem> db.Bookings.aggregate([
...   {
...     $match: {
...       $and: [
...         { GuestID: ObjectId('656ea2e8d29717bc7784d3fa') },
...         { RoomNum: 103 }
...       ]
...     }
...   },
...   {
...     $lookup: {
...       from: 'Guests',
...       localField: 'GuestID',
...       foreignField: '_id',
...       as: 'guestDetails'
...     }
...   },
...   {
...     $unwind: '$guestDetails'
...   },
...   {
...     $project: {
...       _id: 0,
...       RoomNum: 1,
...       CheckInDate: 1,
...       CheckOutDate: 1,
...       NumOfGuests: 1,
...       guestName: { $concat: ["$guestDetails.FirstName", " ", "$guestDetails.LastName"] },
...       guestPhone: '$guestDetails.Phone'
...     }
...   }
... ]);
```

Output:

```
[
  {
    RoomNum: 103,
    CheckInDate: ISODate("2023-03-01T00:00:00.000Z"),
    CheckOutDate: ISODate("2023-03-05T00:00:00.000Z"),
    NumOfGuests: 1,
    guestName: 'Olivia Johnson',
    guestPhone: '333-444-5555'
  }
]
```

Ranishree Anegundi
ID# 99795021

5.3.3) Query services taken by the guest in the specified room.

```
HotelManagementSystem> db.Bookings.aggregate([
...   {
...     $match: {
...       RoomNum: 102
...     }
...   },
...   {
...     $unwind: '$Services'
...   },
...   {
...     $project: {
...       _id: 0,
...       BookingID: 1,
...       RoomNum: 1,
...       Note: '$Services.Note',
...       Charges: '$Services.Charges'
...     }
...   }
... ]);
[
  { RoomNum: 102, Note: 'Room Service', Charges: 10 },
  { RoomNum: 102, Note: 'Laundry', Charges: 30 }
]
```

5.3.4) Query to get the number of staff members.

```
HotelManagementSystem> db.Staff.aggregate([
...   {
...     $group: {
...       _id: 1,
...       totalStaff: { $sum: 1 }
...     }
...   }
... ]);
[ { _id: 1, totalStaff: 7 } ]
HotelManagementSystem>
```

Ranishree Anegundi
ID# 99795021

5.4) Calculate the Total number of items in a period of time.

5.4.1) Query total payment by room 102 with services and number of days stayed with room rent.

```
db.Bookings.aggregate([
 {
  $match: {
   RoomNum: 102
  }
 },
 {
  $lookup: {
   from: 'Rooms',
   localField: 'RoomNum',
   foreignField: '_id',
   as: 'Room'
  }
 },
 {
  $unwind: '$Room'
 },
 {
  $unwind: '$Services'
 },
 {
  $project: {
   _id: 0,
```

```
    RoomNum: 1,
    CheckInDate: 1,
    CheckOutDate: 1,
    NumOfGuests: 1,
    RoomRent: {
     $multiply: [
       {
        $divide: [
          { $subtract: ["$CheckOutDate", "$CheckInDate"] },
          86400000
        ]
       },
       '$Room.PricePerDay'
     ]
    },
    servicesTotal: '$Services.Charges'
  }
},
{
  $group: {
    _id: {
      RoomNum: '$RoomNum',
      CheckInDate: '$CheckInDate',
      CheckOutDate: '$CheckOutDate',
      NumOfGuests: '$NumOfGuests'
    },
```

```
    RoomRent: { $first: '$RoomRent' },
    servicesTotal: { $sum: '$servicesTotal' }
   }
  },
  {
   $project: {
    _id: 0,
    RoomNum: '$_id.RoomNum',
    RoomRent: 1,
    ServicesTotal: 1,
    TotalPayment: {
     $sum: ['$RoomRent', '$servicesTotal']
    }
   }
  }
]);
```

Output:

```
... ]);
[ { RoomRent: 1000, RoomNum: 102, TotalPayment: 1040 } ]
HotelManagementSystem>
```

5.4.2) Query net rooms available on a particular date.

```
var fromDate = ISODate('2023-05-12');
var toDate = ISODate('2023-05-12');
var availableRooms = db.Rooms.countDocuments({ Status: 'Available' });
var bookingResult = db.Bookings.aggregate([
  {
   $match: {
```

Ranishree Anegundi
ID# 99795021

```
      $or: [
        { CheckInDate: { $lte: toDate }, CheckOutDate: { $gte: fromDate } },
        { CheckInDate: { $eq: fromDate } },
        { CheckOutDate: { $eq: toDate } }
      ]
    }
  },
  {
    $count: 'TotalBookings'
  }
]);

var totalBookings = 0;

bookingResult.forEach(function(doc) {
  totalBookings = doc.TotalBookings;
});

var reservationResult = db.Reservations.aggregate([
  {
    $match: {
      $or: [
        { FromDate: { $lte: toDate }, ToDate: { $gte: fromDate } },
        { FromDate: { $eq: fromDate } },
        { ToDate: { $eq: toDate } }
      ]
```

```
    }
  },
  {
    $count: 'TotalReservations'
  }
]);

var totalReservations = 0;

reservationResult.forEach(function(doc) {
  totalReservations = doc.TotalReservations;
});
var netAvailableRooms = availableRooms - (totalBookings + totalReservations);
print('Net Available Rooms on 2023-05-12: ' + netAvailableRooms);
```

Output:

```
HotelManagementSystem> print('Net Available Rooms on 2023-05-12: ' + netAvailableRooms);

Net Available Rooms on 2023-05-12: 14
```

6.Conclusion

The selection of MongoDB as the database system for the Hotel Management System reflects a strategic choice to leverage the advantages of a NoSQL, document-oriented database. MongoDB's flexibility, scalability, and ease of integration align with the dynamic nature of hotel management systems, providing a foundation for efficient data storage, retrieval, and adaptation to evolving business needs.