Group Project - Fall 2023

CPIS-382 Development of E-Systems and Interface Design

Coordinator(s) Dr. Abdulaziz Alsulami

11984

Student ID	
Student Name	
Section	

Total Obtained Marks ☐ out of 15

SO	Max	Obtained Marks for SO
2	15	

Group Project - Fall 2023

CPIS-382 Development of E-Systems and Interface Design

Coordinator(s) Dr. Abdulaziz Alsulami

11984

Student ID	
Student Name	
Section	

Total Obtained Marks [ ] out of 15

SO	Max	Obtained Marks for SO
2	15	

Group Project - Fall 2023

CPIS-382 Development of E-Systems and Interface Design

Coordinator(s) Dr. Abdulaziz Alsulami

11984

Student ID	
Student Name	
Section	

Total Obtained Marks out of 15

SO	Max	Obtained Marks for SO
2	15	

Computing Accreditation Commission

AIMS 4.0

Group Project - Fall 2023

CPIS-382 Development of E-Systems and Interface Design

Coordinator(s) Dr. Abdulaziz Alsulami

11984

Student ID	
Student Name	
Section	

Total Obtained Marks ______ out of 15

SO	Max	Obtained Marks for SO
2	15	

Printed by : Ms. Waad Alshehri Wednesday, November 8 2023

FACULTY OF COMPUTING
& INFORMATION TECHNOLOGY
KING ABDULAZIZ UNIVERSITY

كلية الحاسبات
وتقنية المعلومات
جامعة الملك عبدالعزيز

FCIT
KAU

CPIS382-Final Project
Hotel Booking System

Prepared by

Majd Alasmari	2206923
Layan alghamdi	2207091
Maria Alghamdi	2210867
Samaher Khan	2207268

Table of content

Abstract

This project uses the Resource Description Framework (RDF) and the Web Ontology Language (OWL) to create a semantic ontology for a hotel reservation and management system. The core components of hotel operations, including hotels, room categories, guests, staff, reservations, payments, services, amenities, locations, and customer reviews, are modeled by the ontology. This ontology addresses the limitations of traditional systems that lack semantic understanding and structured reasoning.

As a solution to this limitation, the project applies a structured methodology that includes defining classes and subclasses, modeling object and data properties, and creating individuals that represent real hotel entities. SPARQL queries are used to retrieve semantic information such as room availability, hotel amenities, reservation details, payment types, staff roles, and guest reviews. The resulting ontology enhances interoperability, supports semantic reasoning and intelligent querying, and provides a flexible foundation for building smart hotel management applications

Introduction

Domain Overview

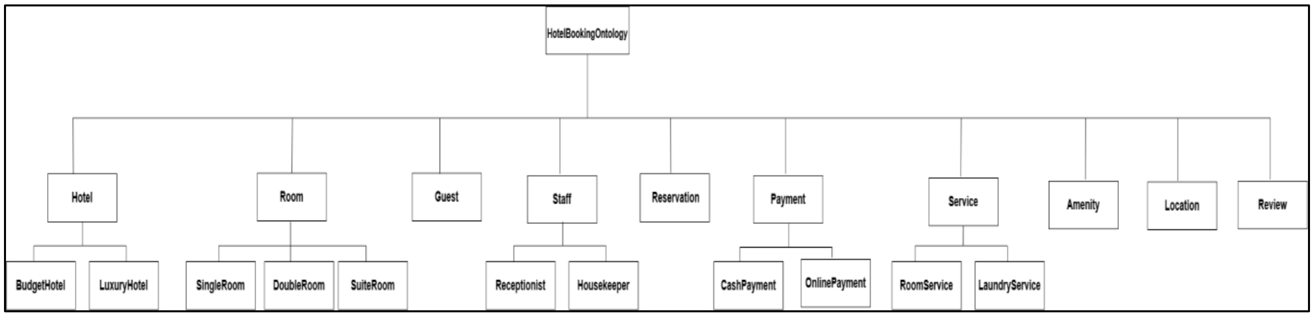
The hotel booking domain encompasses managing relationships between several interconnected entities, including hotels, guest accommodations, reservations, payments, staff, and customer reviews. Hotels offer various room types with diverse amenities and services, payments are processed in different ways, staff are hired systematically, and guests provide reviews based on their experiences in various ways. Representing these relationships requires high flexibility and intelligent query enablement by building an approach that goes beyond the data-saving approach found in traditional booking systems.

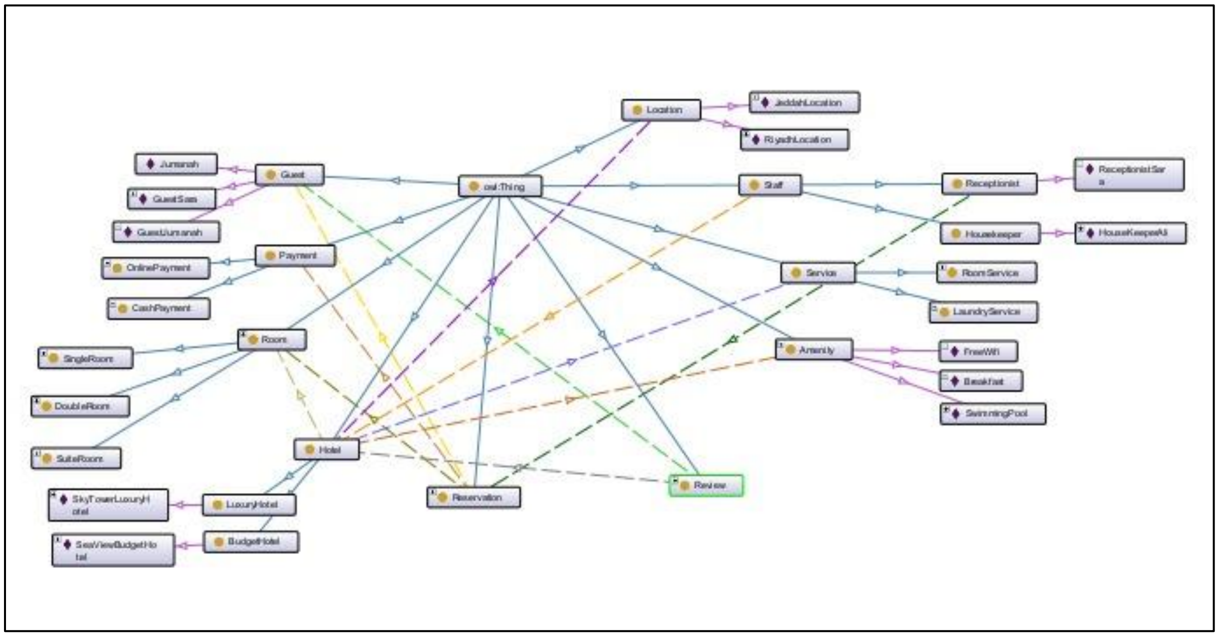
Application Overview

The ontology we have built in this project provides a flexible and intelligent representation of the hotel booking domain using Semantic Web standards. It models essential components through a hierarchical class structure, including two hotel types: Budget and Luxury, and three room categories: Single, Double, and Suite. It also includes two staff roles: Receptionist and Housekeeper. Guest information captures names and email addresses, while reservations track booking dates, prices, associated rooms, and payment details. Payment transactions are categorized as Cash or Online, with recorded amounts. Customer reviews include numerical ratings and text feedback. Services such as Room Service and Laundry are associated with hotels, and location information connects properties to cities like Jeddah and Riyadh. This semantic structure enables SPARQL queries to retrieve specific information, such as luxury hotels with amenities, reservations by specific guests, hotels exceeding rating thresholds, and staff assignments across properties.

Application Design

- Tree Diagram





- **Description of classes and subclasses**

Class	**Description**	**Hierarchy Position**
Hotel	Represents hotels in general.	Class.
LuxuryHotel	Represents high-end or luxury hotels.	Subclass of Hotel
BudgetHotel	Represents economical hotels.	Subclass of Hotel
Room	Represents the types of rooms available in a hotel.	Class
SingleRoom	Represents single occupancy rooms.	Subclass of Room
DoubleRoom	Represents double occupancy rooms.	Subclass of Room
SuiteRoom	Represents hotel suites.	Subclass of Room

Payment	Represents methods of payment for services.	Class
CashPayment	Represents payments made in cash.	Subclass of Payment
OnlinePayment	Represents payments made online.	Subclass of Payment
Staff	Represents employees who work at the hotel.	Class
Receptionist	Represents front desk staff.	Subclass of Staff
Housekeeper	Represents housekeeping staff.	Subclass of Staff
Service	Represents the services offered by the hotel.	Class
LaundryService	Represents laundry services.	Subclass of Service
RoomService	Represents in-room dining or service.	Subclass of Service

Guest	Represents the hotel customers.	Calss
Amenity	Represents facilities or extra features like breakfast and Free WIFI	Class
Reservation	Represents a booking reservation for a room.	Class
Review	Represents ratings and reviews of the hotel or service.	Class
Location	Represents the geographical location of the hotel.	Class

- **Description of object properties**

Object properties define relationships between classes, connecting different entities in the ontology.

Property	Domain	Range	Description
bookedBy	Reservation	Guest	Links a reservation to the guest who made the booking
handlesReservation	Receptionist	Reservation	Associates a receptionist with the reservations they process.
hasAmenity	Hotel	Amenity	Connects a hotel to the amenities it offers like WiFi and breakfast
hasRoom	Hotel	Room	Links a hotel to the rooms it contains
locatedIn	Hotel	Location	Specifies the geographical location (city/region) where a hotel is situated

offersService	Hotel	Service	Associates a hotel with the services it provides such as room service and laundry
paidBy	Reservation	Payment	Links a reservation to the payment transaction used
reservesRoom	Reservation	Room	Connects a reservation to the specific room that was booked
reviewsHotel	Review	Hotel	Links a review to the hotel being reviewed
worksAt	Staff	Hotel	Links a staff member to the hotel where they are employed
writtenBy	Review	Guest	Connects a review to the guest who wrote it

- Data Properties

Data properties define attributes that store literal values strings, numbers, dates for classes.

Property	**Domain**	**Range**	**Description**
checkInDate	Reservation	xsd:dateTime	The check-in date for the reservation
checkOutDate	Reservation	xsd:dateTime	The check-out date for the reservation
guestEmail	Guest	xsd:string	The email address of the guest
guestName	Guest	xsd:string	The name of the guest
hotelName	Hotel	xsd:string	The name of the hotel
hotelRating	Hotel	xsd:decimal	The rating of the hotel
paymentAmount	Payment	xsd:decimal	The amount paid in the payment transaction

reviewRating	Review	xsd:int	The numerical rating given in the review
reviewText	Review	xsd:string	The actual text or comment of the review
roomNumber	Room	xsd:string	The number identifying the room
totalPrice	Reservation	xsd:decimal	The total cost of the reservation

- Description of SPARQL Queries

SPARQL queries enable intelligent retrieval and analysis of ontology data.

Query01: Retrieve All Hotels with Their Ratings

Description
This query retrieves all hotels in the ontology along with their names and ratings, sorted from highest-rated to lowest-rated.
How it works
It selects individuals that belong to either BudgetHotel or LuxuryHotel and retrieves the hotel's name using hotelName. Then it gets the rating value via the hotelRating data property and returns the hotel identifier, hotel name, and rating value.

Query02: Retrieve Detailed Reservation Information

Description
This query retrieves complete reservation details, showing which guest made each reservation, the room number, check-in and check-out dates, and the total price.
How it works
It identifies individuals from the Reservation class and retrieves details such as reservation ID, guest name, room number, check-in date, check-out date, and total price.

Query03: Retrieve Hotels and All Their Room Numbers

Description
This query retrieves all hotels Budget and Luxury together with all the room numbers available in each hotel.
How it works
It selects hotels from the two subclasses: Budget and Luxury hotels. Then it retrieves the hotel name from the hotelName property and uses the hasRoom object property to connect each hotel to its rooms. For each room, it retrieves its room number and returns the hotel name and room number.

Query04: Retrieve Guest Reviews for Hotels Rating ≥ 4

Description

This query retrieves all guest reviews with a rating of 4 or higher, along with the guest name and the hotel being reviewed.

How it works

It identifies individuals in the Review class, retrieves the rating using reviewRating, links each review to the guest via writtenBy, and retrieves the hotel using reviewsHotel. The FILTER keeps only results where rating ≥ 4. Finally, it returns the guest name, hotel name, and review rating.

Query05: Retrieve All Staff with Their Roles and Hotel

Description

This query retrieves all staff members working in the hotel booking system, along with their job roles and the hotel they are assigned to.

How it works

It finds individuals that belong to the Staff class. Then, for each staff member, it checks their role and retrieves the hotel they work at using the worksAt object property. A FILTER is used to include only staff from the two role subclasses (Receptionist and Housekeeper). Finally, it returns the staff name, role type, and hotel name they work at.

Query06: Count the Number of Reservations per Guest

Description

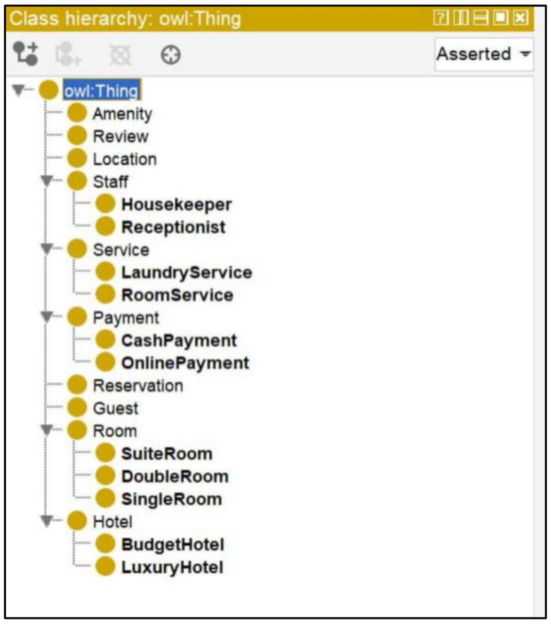
This query counts how many reservations each guest has made in the hotel booking system.

How it works

It selects individuals from the Reservation class, uses the bookedBy property to find the guest who made each reservation, groups results by guest name, and computes the number of reservations using the COUNT function. Finally, it returns the guest name and reservation count.

Screenshots

- Classes and Subclasses

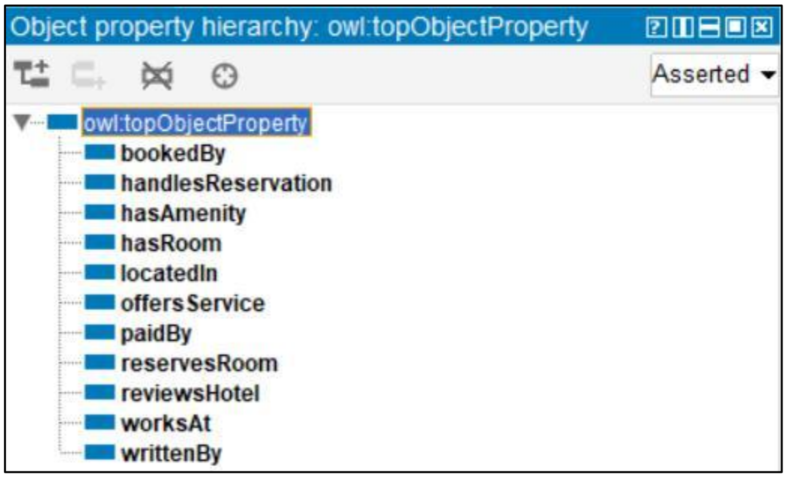


Class hierarchy: owl:Thing

Asserted

- owl:Thing
  - Amenity
  - Review
  - Location
  - Staff
    - Housekeeper
    - Receptionist
  - Service
    - LaundryService
    - RoomService
  - Payment
    - CashPayment
    - OnlinePayment
  - Reservation
  - Guest
  - Room
    - SuiteRoom
    - DoubleRoom
    - SingleRoom
  - Hotel
    - BudgetHotel
    - LuxuryHotel

- Object properties

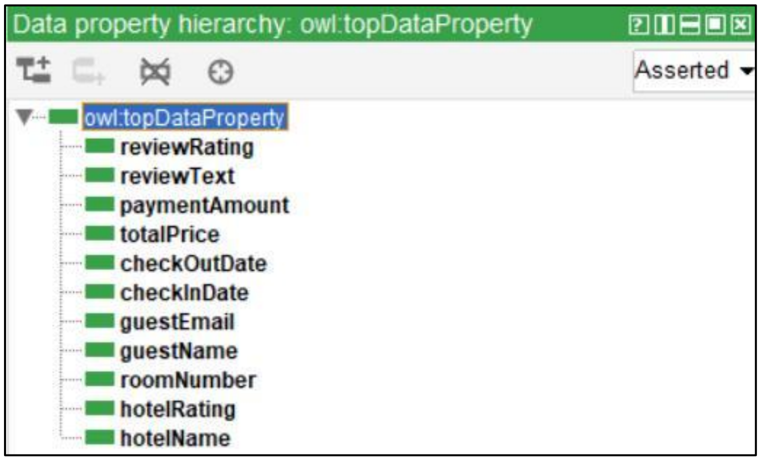


Object property hierarchy: owl:topObjectProperty

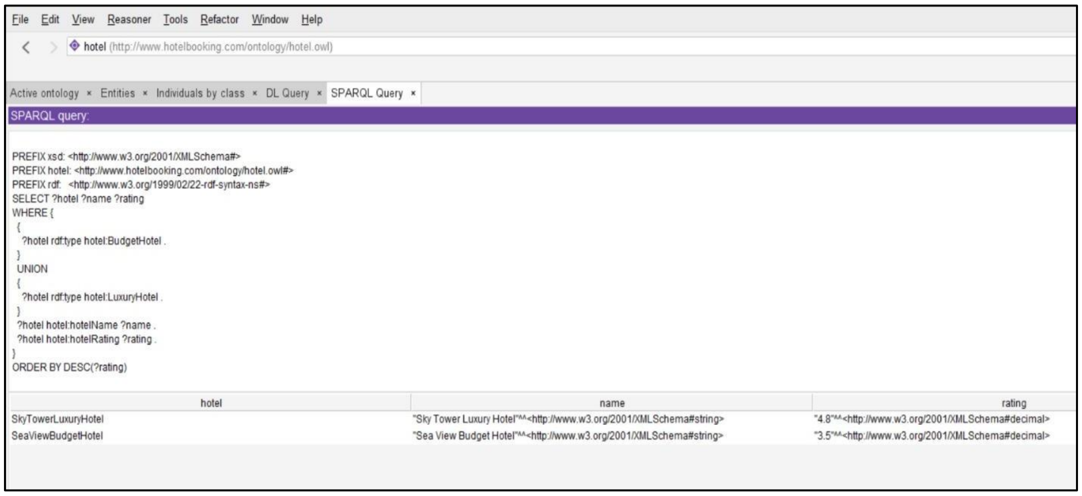
Asserted

- owl:topObjectProperty
  - bookedBy
  - handlesReservation
  - hasAmenity
  - hasRoom
  - locatedIn
  - offersService
  - paidBy
  - reservesRoom
  - reviewsHotel
  - worksAt
  - writtenBy

- Data properties



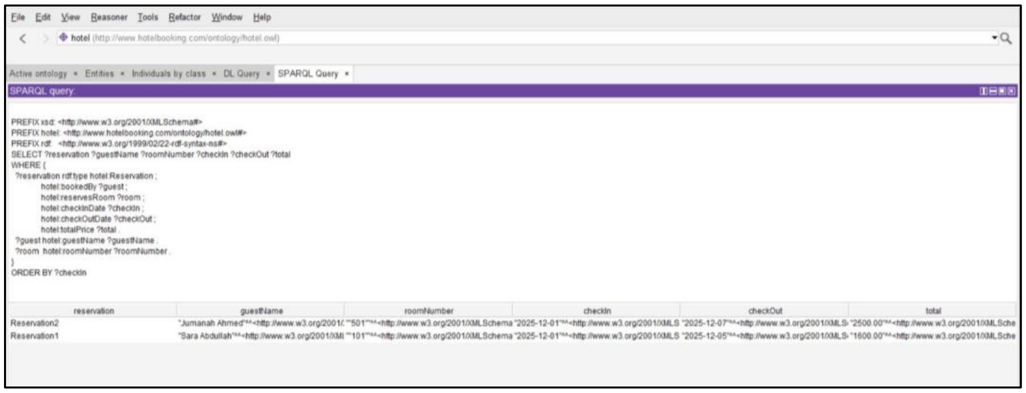
- SPARQL Queries


File Edit View Reasoner Tools Refactor Window Help
hotel (http://www.hotelbooking.com/ontology/hotel.owl)
Active ontology
Entities
Individuals by class
DL Query
SPARQL Query
SPARQL query:

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX hotel: <http://www.hotelbooking.com/ontology/hotel.owl#>
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?hotel ?name ?rating
WHERE {
 {
  ?hotel rdf:type hotel:BudgetHotel .
 }
 UNION
 {
  ?hotel rdf:type hotel:LuxuryHotel .
 }
 ?hotel hotel:hotelName ?name .
 ?hotel hotel:hotelRating ?rating .
}
ORDER BY DESC(?rating)
```

hotel	name	rating
SkyTowerLuxuryHotel	"Sky Tower Luxury Hotel"^^<http://www.w3.org/2001/XMLSchema#string>	"4.8"^^<http://www.w3.org/2001/XMLSchema#decimal>
SeaViewBudgetHotel	"Sea View Budget Hotel"^^<http://www.w3.org/2001/XMLSchema#string>	"3.5"^^<http://www.w3.org/2001/XMLSchema#decimal>

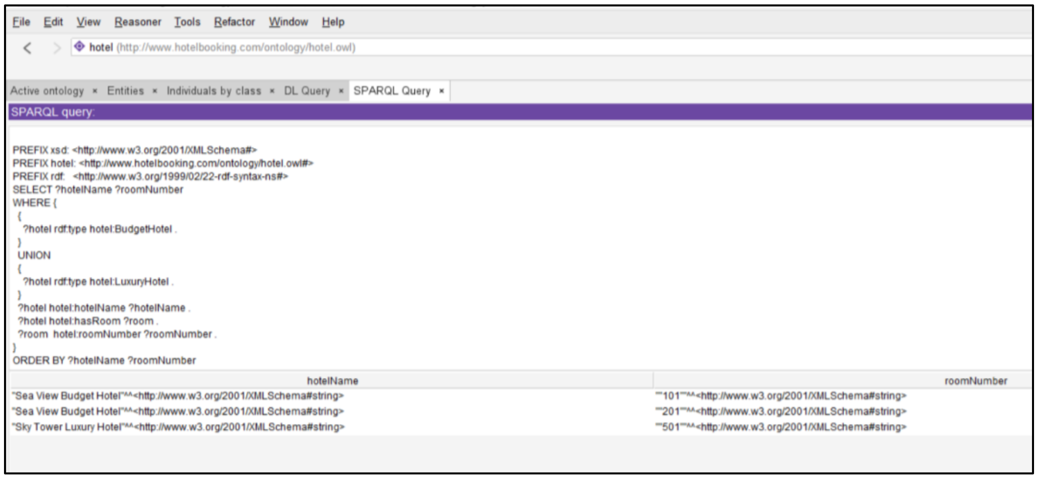
Query01: Retrieve All Hotels with Their Ratings


File Edit View Reasoner Tools Refactor Window Help
hotel (http://www.hotelbooking.com/ontology/hotel.owl)
Active ontology × Entities × Individuals by class × DL Query × SPARQL Query ×
SPARQL query:

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX hotel: <http://www.hotelbooking.com/ontology/hotel.owl#>
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?reservation ?guestName ?roomNumber ?checkIn ?checkOut ?total
WHERE {
 ?reservation rdf:type hotel:Reservation ;
          hotel:bookedBy ?guest ;
          hotel:reservesRoom ?room ;
          hotel:checkInDate ?checkIn ;
          hotel:checkOutDate ?checkOut ;
          hotel:totalPrice ?total .
 ?guest hotel:guestName ?guestName .
 ?room  hotel:roomNumber ?roomNumber .
}
ORDER BY ?checkIn
```

```
reservation    guestName                                      roomNumber                                      checkIn                                        checkOut                                       total
Reservation2   "Jumanah Ahmed"^^<http://www.w3.org/2001/.  ""501""^^<http://www.w3.org/2001/XMLSchema  "2025-12-01"^^<http://www.w3.org/2001/XMLS  "2025-12-07"^^<http://www.w3.org/2001/XMLS  "2500.00"^^<http://www.w3.org/2001/XMLSche
Reservation1   "Sara Abdullah"^^<http://www.w3.org/2001/XMl  ""101""^^<http://www.w3.org/2001/XMLSchema  "2025-12-01"^^<http://www.w3.org/2001/XMLS  "2025-12-05"^^<http://www.w3.org/2001/XMLS  "1600.00"^^<http://www.w3.org/2001/XMLSche
```

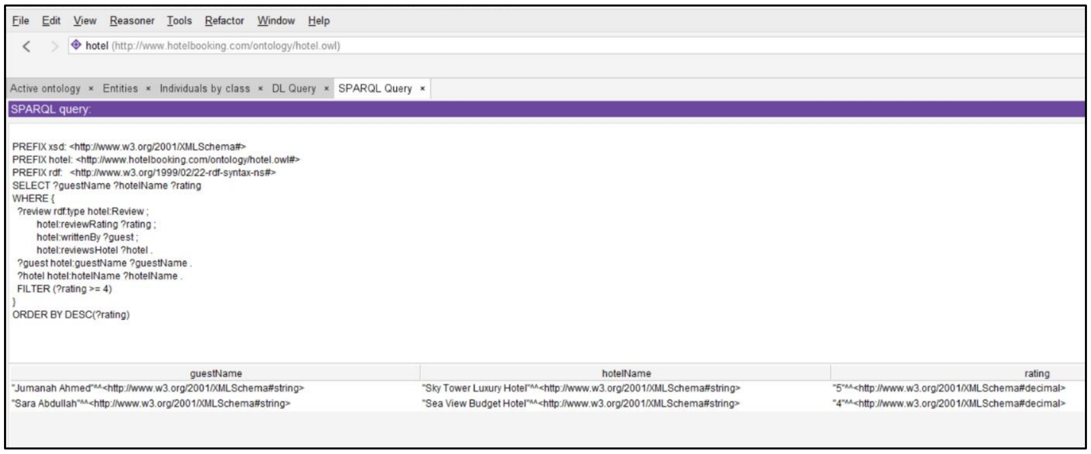
Query02: Retrieve Detailed Reservation Information


File Edit View Reasoner Tools Refactor Window Help
hotel (http://www.hotelbooking.com/ontology/hotel.owl)
Active ontology × Entities × Individuals by class × DL Query × SPARQL Query ×
SPARQL query:

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX hotel: <http://www.hotelbooking.com/ontology/hotel.owl#>
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?hotelName ?roomNumber
WHERE {
 {
  ?hotel rdf:type hotel:BudgetHotel .
 }
 UNION
 {
  ?hotel rdf:type hotel:LuxuryHotel .
 }
 ?hotel hotel:hotelName ?hotelName .
 ?hotel hotel:hasRoom ?room .
 ?room  hotel:roomNumber ?roomNumber .
}
ORDER BY ?hotelName ?roomNumber
```

```
hotelName                                                               roomNumber
"Sea View Budget Hotel"^^<http://www.w3.org/2001/XMLSchema#string>      ""101""^^<http://www.w3.org/2001/XMLSchema#string>
"Sea View Budget Hotel"^^<http://www.w3.org/2001/XMLSchema#string>      ""201""^^<http://www.w3.org/2001/XMLSchema#string>
"Sky Tower Luxury Hotel"^^<http://www.w3.org/2001/XMLSchema#string>     ""501""^^<http://www.w3.org/2001/XMLSchema#string>
```

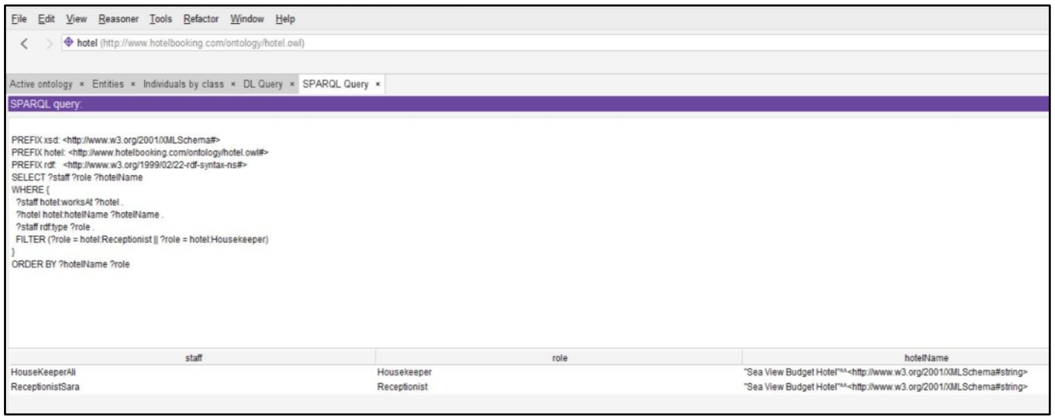
Query03: Retrieve Hotels and All Their Room Numbers


File Edit View Reasoner Tools Refactor Window Help
hotel (http://www.hotelbooking.com/ontology/hotel.owl)
Active ontology × Entities × Individuals by class × DL Query × SPARQL Query ×
SPARQL query:

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX hotel: <http://www.hotelbooking.com/ontology/hotel.owl#>
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?guestName ?hotelName ?rating
WHERE {
 ?review rdf:type hotel:Review ;
       hotel:reviewRating ?rating ;
       hotel:writtenBy ?guest ;
       hotel:reviewsHotel ?hotel .
 ?guest hotel:guestName ?guestName .
 ?hotel hotel:hotelName ?hotelName .
 FILTER (?rating >= 4)
}
ORDER BY DESC(?rating)
```

```
guestName                                                        hotelName                                                              rating
"Jumanah Ahmed"^^<http://www.w3.org/2001/XMLSchema#string>      "Sky Tower Luxury Hotel"^^<http://www.w3.org/2001/XMLSchema#string>   "5"^^<http://www.w3.org/2001/XMLSchema#decimal>
"Sara Abdullah"^^<http://www.w3.org/2001/XMLSchema#string>      "Sea View Budget Hotel"^^<http://www.w3.org/2001/XMLSchema#string>    "4"^^<http://www.w3.org/2001/XMLSchema#decimal>
```

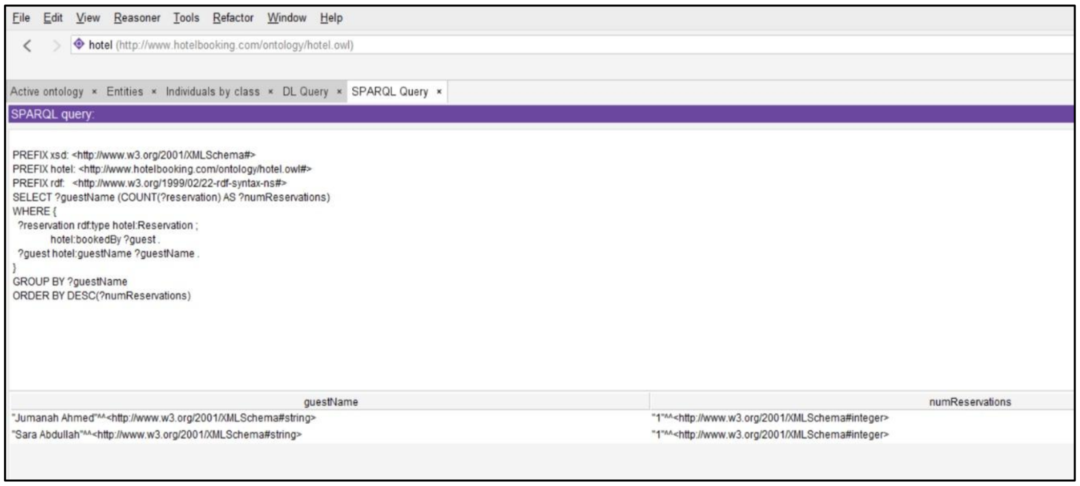
Query04: Retrieve Guest Reviews for Hotels Rating ≥ 4



```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX hotel: <http://www.hotelbooking.com/ontology/hotel.owl#>
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?staff ?role ?hotelName
WHERE {
 ?staff hotel:worksAt ?hotel .
 ?hotel hotel:hotelName ?hotelName .
 ?staff rdf:type ?role .
 FILTER (?role = hotel:Receptionist || ?role = hotel:Housekeeper)
}
ORDER BY ?hotelName ?role
```

staff	role	hotelName
HouseKeeperAli	Housekeeper	"Sea View Budget Hotel"^^<http://www.w3.org/2001/XMLSchema#string>
ReceptionistSara	Receptionist	"Sea View Budget Hotel"^^<http://www.w3.org/2001/XMLSchema#string>

Query05: Retrieve All Staff with Their Roles and Hotel



```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX hotel: <http://www.hotelbooking.com/ontology/hotel.owl#>
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?guestName (COUNT(?reservation) AS ?numReservations)
WHERE {
 ?reservation rdf:type hotel:Reservation ;
          hotel:bookedBy ?guest .
 ?guest hotel:guestName ?guestName .
}
GROUP BY ?guestName
ORDER BY DESC(?numReservations)
```

guestName	numReservations
"Jumanah Ahmed"^^<http://www.w3.org/2001/XMLSchema#string>	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
"Sara Abdullah"^^<http://www.w3.org/2001/XMLSchema#string>	"1"^^<http://www.w3.org/2001/XMLSchema#integer>

Query06: Count the Number of Reservations per Guest

Conclusion

In this project, we developed a semantic ontology for a hotel reservation and management system using RDF and OWL to address the limitations found in traditional booking platforms. By modeling hotels, rooms, guests, staff, reservations, payments, services, amenities, locations, and reviews as interconnected semantic entities, the ontology provides a structured and machine-interpretable representation of the hotel booking domain. This semantic structure enables intelligent querying through SPARQL, allowing users and systems to retrieve meaningful information such as room types, guest reservations, hotel ratings, available services, and staff roles with greater accuracy and flexibility.

The ontology enhances interoperability, supports semantic reasoning, and improves data integration across different components of the system. Through this approach, hotel management can benefit from more efficient data handling, streamlined operations, and smarter decision support. Overall, the developed ontology lays a solid foundation for future extensions, such as integrating recommendation systems, automated reasoning tools, or advanced analytics to further improve the effectiveness and intelligence of hotel management applications.