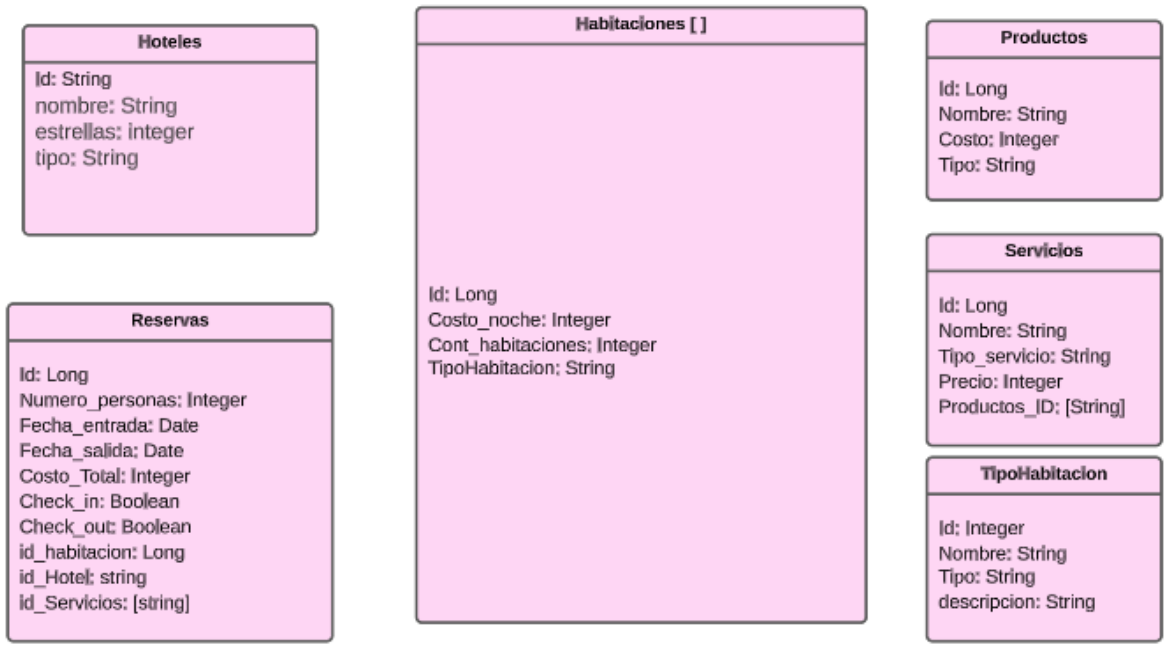
Iteración 3 proyecto Sistrans C6

María Fernanda de la Hoz

Juan David Obando

Julián Contreras

2. (7%) Análisis y modelo conceptual a. Proponga un modelo conceptual en UML o E/R que describa las entidades del modelo de datos para la aplicación que se quiere desarrollar.



a-Identifiquen las entidades y sus atributos

Hotel: 20

Id: String

Nombre: String

Tipo: String

Estrellas: Integer

TipoHabitacion: 20

Id: Integer

Nombre: String

Descripcion: String

Tipo: String

Habitacion: 200

Id: Long

Tipo_habitacion: String

Costo_noche: Integer

Cont_habitaciones: Integer

Reserva:17.000

Id: Long

Numero_personas: Integer

Fecha_entrada: Date

Fecha_salida: Date

Costo_Total: Integer

Check_in: Boolean

Check_out: Boolean

id_habitacion: Long

id_Hotel: string

id_Servicios: [string]

Producto: 40

Id: Long

Nombre: String

Costo: Integer

Tipo: String

Servicio: 10

Id: Long

Nombre: String

Tipo_servicio: String

Precio: Integer

Productos_ID: [String]

c. Analicen las operaciones de lectura y escritura para cada entidad. Para ello utilicen una tabla como la del ejemplo del anexo A. Recuerden que este análisis sirve para saber que información se accederá de manera conjunta.

Entidades	Operaciones	Información necesaria	Tipo
Hotel	Fetch	Hotel details + estrellas	Read
Reserva	Fetch	Reserva details	Read
Servicio	Add servicio	Servicio details	Write
Producto	Add producto	Producto details	Write

Habitacion	Add habitacion	Habitacion details	Write
Plan de consumo	Add plan de consumo	Plan de consumo details	Write
Caracteristicas Habitación	Add caracteristicas habitacion	Caracteristicas habitacion details	Write
Reserva	Update	Reserva details	Write

d. Cuantifiquen las operaciones de lectura y escritura para cada entidad. Para ello utilicen una tabla como la del ejemplo del anexo B.

Entidades	Operaciones	Información necesaria	Tipo	Rate
Hotel	Fetch	Hotel details + estrellas	Read	1/10 min
Reserva	Fetch	Reserva details	Read	100/min
Servicio	Add servicio	Servicio details	Write	all/min
Producto	Add producto	Producto details	Write	all/min
Habitacion	Add habitacion	Habitacion details	Write	all/min
Caracteristicas Habitación	Add caracteristicas habitacion	Caracteristicas habitacion details	Write	all/min
Reserva	Update	Reserva details	Write	1000/min

b. (15%) Describan las entidades de datos y las relaciones entre ellas que corresponden al modelo conceptual UML propuesto. Para ello, presenten lo siguiente:

a. La lista de entidades con la descripción de cada una de ellas.

Hotel: Esta entidad, representa a casa hotel sobre el cual opera la aplicación.

CaracteristicaHabitacion: Detalla los atributos específicos que tiene cada habitación.

Habitacion: Representa cada una de las habitaciones presentes dentro de un hotel.

Reserva: Contiene la información de los huespedes y de su estancia dentro del hotel.

Producto: Cualquier tipo de consumo ofrecido dentro de un servicio.

Servicio: Actividades extra que ofrece un hotel.

b. Las relaciones entre entidades y su cardinalidad (uno a uno, uno a muchos, o muchos a muchos).

Hotel-Habitación (1-*): Una habitación tiene un solo hotel, pero un hotel puede tener muchas habitaciones.

Hotel-Reserva (1-*): Una reserva tiene un solo hotel, pero un hotel puede tener muchas reservas.

Hotel-Servicio (1-*): Un servicio tiene un solo hotel, pero un hotel puede tener muchos servicios.

Habitación-CaracterísticaHabitación (* - *): Una habitación puede tener muchas características al igual que una característica puede estar presenta en muchas habitaciones.

Reserva-Servicio(*-*): Una reserva puede tener muchos servicios, al igual que un servicio puede tener muchas reservas.

c. El análisis de selección de esquema de asociación (referenciado o embebido) para cada relación entre entidades. Para ello use la tabla de análisis vista en clase, la cual se retoma en el anexo C, junto con los resultados del análisis de la carga de trabajo (workload), descrita antes.

Relación	Embeber	Referenciar	Factores de Decisión
Hotel - Habitación (1-*)	Sí	No	Incrustar "Habitaciones" en "Hotel" simplifica el modelo ya que las operaciones de consulta y actualización de la información de las habitaciones suelen ser atómicas con respecto al hotel (Simplicidad, Atomicidad de Consulta, Complejidad de Actualización). Sin embargo, si el número de habitaciones es muy grande y cambia con frecuencia, se podría considerar referenciar para evitar tamaños grandes de documentos (Tamaño del Documento).
Hotel - Reserva (1-*)	No	Sí	Las reservas típicamente se hacen y actualizan de manera independiente de la entidad del hotel, y puede haber un gran número de reservas con el tiempo (Cardinalidad, Crecimiento del Documento).
Hotel - Servicio (1-*)	No	Sí	Los servicios pueden actualizarse independientemente del hotel, y un hotel puede ofrecer una amplia gama de servicios (Individualidad, Cardinalidad).
Habitación - Característica Habitación (-)	No	Sí	Dado que existe una relación muchos-a-muchos, se prefiere la referencia. Una característica puede estar en muchas habitaciones, y una habitación puede tener muchas características (Cardinalidad, Duplicación de Datos).
Reserva - Servicio (-)	No	Sí	Con una relación muchos-a-muchos, la referencia es apropiada. Una reserva podría incluir múltiples servicios y un servicio podría ser parte de muchas reservas (Cardinalidad, Complejidad de Actualización).

d. Una descripción gráfica usando Json de cada relación entre entidades en donde presente un ejemplo de datos junto con el esquema de asociación usado (referenciado o embebido). En el anexo D se muestra un ejemplo de lo que se requiere.

Hotel - Habitación (1 a muchos)
Se utiliza una referencia del Hotel en el documento Habitación.

Hotel:

{"Id": 1,

"Nombre": "Hotel Sunshine",

"Tipo": "Resort",

"Estrellas": 5}

Habitación:

{"Id": 101,

"Tipo_habitacion": "Suite",

"Costo_noche": 150,

"Cont_habitaciones": 10,

"Hotel_Id": 1 // Referencia al ID del Hotel}

Habitación - CaracterísticaHabitación (muchos a muchos)
Se utiliza un array de referencias a CaracterísticaHabitación en el documento Habitación.

```
db.habitaciones.insertOne({
  "_id": 101,
  "costo_noche": 150,
  "cont_habitaciones": 10,
  "tipoHabitacion": [{
    "_id": 201,
    "nombre": "A2",
    "descripcion": "Vista al mar",
    "tipo": "suite"
  }]
})
```

tipoHabitación: {"Id": 201,

"nombre": "A2",

"descripcion":: "Vista al mar",

"tipo": "suite",

}

Hotel - Reserva (1 a muchos)

Se utiliza una referencia del Hotel en el documento Reserva.

Reserva: {"Id": 301,

"Numero_personas": 2,

"Fecha_entrada": "2023-07-01T00:00:00Z",

"Fecha_salida": "2023-07-05T00:00:00Z",

"Costo_Total": 600,

"Check_in": true,

"Check_out": false,

"Hotel_Id": 1 // Referencia al ID del Hotel }

Hotel - Servicio (1 a muchos)

Se utiliza una referencia del Hotel en el documento Servicio.

Servicio: {"Id": 401,

"Nombre": "Limpieza diaria",

"Tipo_servicio": "Limpieza",

"Precio": 30,

"Hotel_Id": 1 // Referencia al ID del Hotel}

Reserva - Servicio (muchos a muchos)

Se utilizan arrays de referencias tanto en Reserva como en Servicio para establecer la relación muchos a muchos.

Reserva: {"Id": 301,

"Numero_personas": 2,

"Fecha_entrada": "2023-07-01T00:00:00Z",

"Fecha_salida": "2023-07-05T00:00:00Z",

"Costo_Total": 600,

"Check_in": true,

"Check_out": false,

"Servicios": [401, 402] // Referencias a los IDs de Servicios}

Servicio: {"Id": 401,

"Nombre": "Limpieza diaria",

"Tipo_servicio": "Limpieza",

"Precio": 30,

"Reservas": [301, 302] // Referencias a los IDs de Reservas}

c. (10%) Cree en MongoDB las colecciones principales de su base, así como la validación de los esquemas. Puede usar Compass o Mongo Shell (Mongosh) para realizar este proceso. Guarde lo hecho en un archivo. Anexe a los entregables los archivos con los scripts utilizados.

```
// Colección de hoteles
db.createCollection("hoteles");
db.hoteles.insertOne({
  Id: 1,
  Nombre: "Hotel Andes",
  Tipo: "Resort",
  Estrellas: 4
});
```

```
// Colección de características de habitaciones
db.createCollection("caracteristicasHabitaciones");
db.caracteristicasHabitaciones.insertOne({
  Id: 1,
  Nombre: "Vista al mar"
});

// Colección de habitaciones
db.createCollection("habitaciones");
db.habitaciones.insertOne({
  Id: 1,
  Tipo_habitacion: "Suite",
  Costo_noche: 200,
  Cont_habitaciones: 10,
  Caracteristicas: [1] // Array de IDs de características
});

// Colección de reservas
db.createCollection("reservas");
db.reservas.insertOne({
  Id: 1,
  Numero_personas: 2,
  Fecha_entrada: new Date("2023-01-01"),
  Fecha_salida: new Date("2023-01-05"),
  Costo_Total: 1000,
  Check_in: true,
  Check_out: false,
  Servicios: []

});
```

```
// Colección de productos
db.createCollection("productos");
db.productos.insertOne({
  Id: 1,
  Nombre: "Botella de Agua",
  Costo: 2,
  Tipo: "Consumible"
});

// Colección de servicios
db.createCollection("servicios");
db.servicios.insertOne({
  Id: 1,
  Nombre: "Limpieza",
  Tipo_servicio: "Diario",
  Precio: 20
});

// Para relacionar las colecciones, se utilizan referencias a los IDs.
db.habitaciones.updateOne(
  { Id: 1 },
  { $set: { Hotel_Id: 1 } } // Referencia al ID del hotel
);

// Una reserva pertenece a un hotel:
db.reservas.updateOne(
  { Id: 1 },
  { $set: { Hotel_Id: 1 } } // Referencia al ID del hotel
);

// Un servicio pertenece a un hotel:
```

```
db.servicios.updateOne(
  { Id: 1 },
  { $set: { Hotel_Id: 1 } } // Referencia al ID del hotel
);

// Para la relación muchos a muchos entre habitaciones y características:
db.habitaciones.updateOne(
  { Id: 1 },
  { $set: { Caracteristicas: [1, 2] } } // Referencia a los IDs de las características
);

// Para la relación muchos a muchos entre reservas y servicios:
db.reservas.updateOne(
  { Id: 1 },
  { $set: { Servicios: [1, 2] } } // Referencia a los IDs de los servicios
);
```