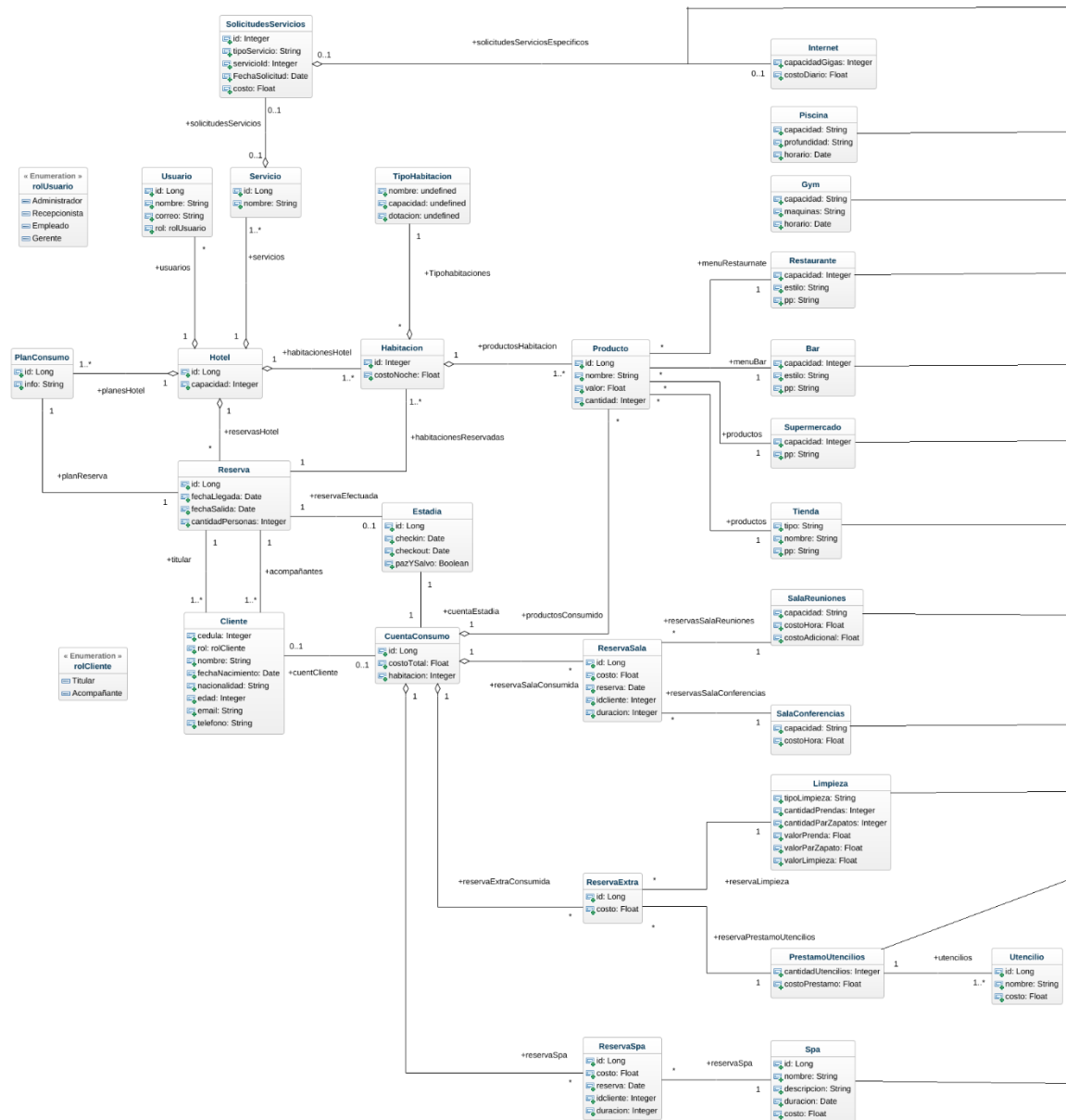
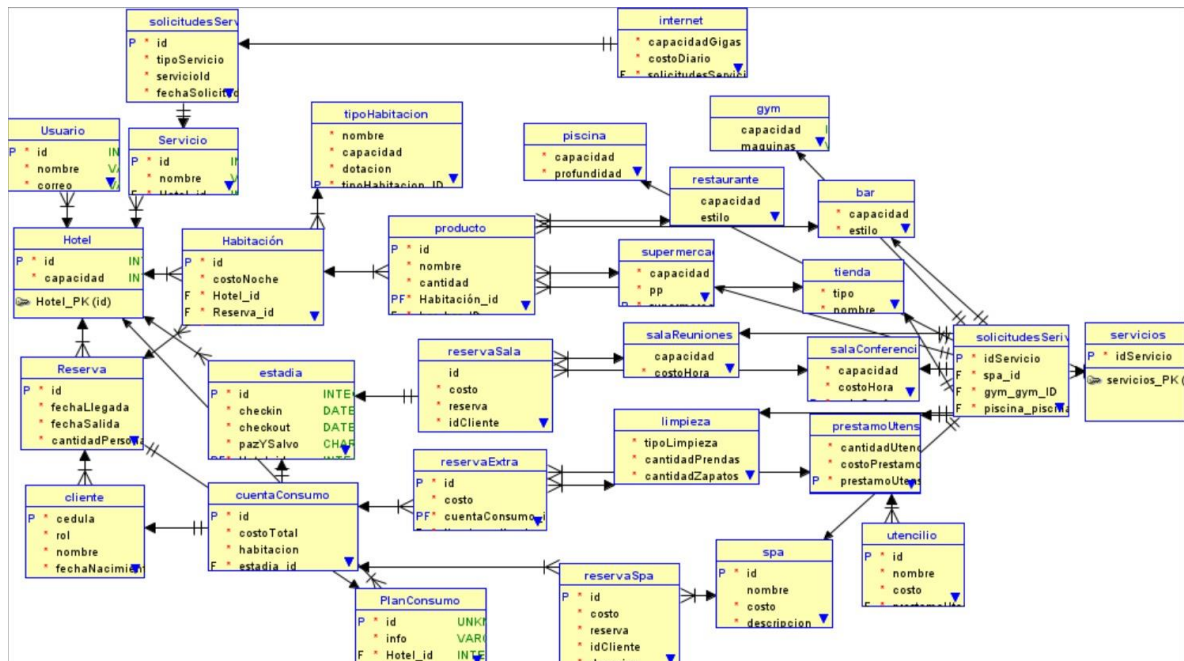
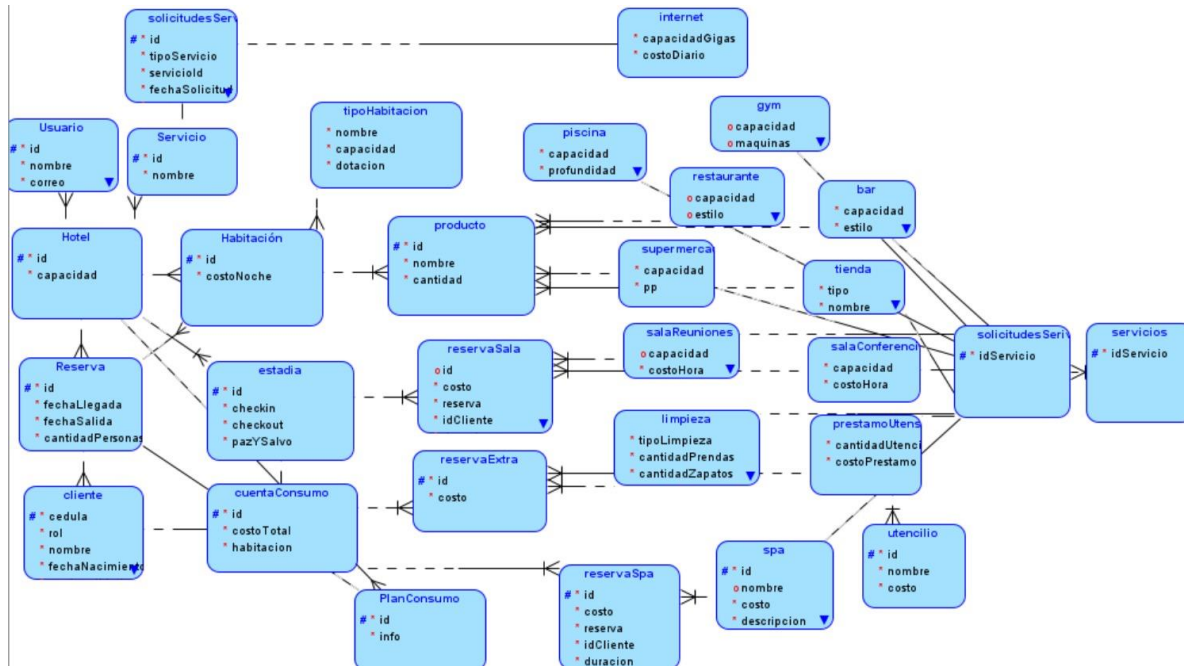


# Documentacion Proyecto 3 SISTRANS

## 1. UML



## 2. E/R



### **3. Elección de embebido vs referenciado:**

#### **1. Tipo de Habitación (embebido en Habitación):**

- ¿Por qué embebido? Un Tipo de Habitación (como 'normal', 'suite', etc.) es una característica intrínseca de la Habitación y raramente cambia. Al embeber el Tipo de Habitación directamente en el documento de Habitación, se facilita la recuperación de toda la información de la habitación en una sola consulta, sin necesidad de realizar joins o referencias adicionales. Esto es eficiente para operaciones de lectura.

- Ventajas: Rápido acceso a datos relacionados sin necesidad de consultas adicionales. Reducción de la complejidad en las consultas y en la lógica de la aplicación.

- Consideraciones: Si los detalles del Tipo de Habitación cambian (lo cual es raro), se deben actualizar todos los documentos de Habitación que contienen esa información. Esto no es óptimo para datos que cambian con frecuencia.

#### **2. Reserva (referenciando Tipo de Habitación y Habitación):**

- ¿Por qué referenciado? Las reservas son transacciones que ocurren en un momento específico y tienen una relación directa tanto con el Tipo de Habitación como con la Habitación específica. Referenciar permite mantener la integridad de los datos sin duplicar información. Si una habitación o su tipo se actualizan, no hay necesidad de actualizar cada reserva; simplemente se sigue la referencia para obtener la información actual.

- Ventajas: Mantiene la consistencia de los datos y evita la duplicación. Facilita la actualización de los datos de las habitaciones y sus tipos, ya que no se requiere actualizar múltiples reservas.

- Consideraciones: Las consultas pueden ser ligeramente más complejas, ya que requieren seguir referencias para obtener información completa.

#### **3. Estadías (referenciando Reserva):**

- ¿Por qué referenciado? Las estancias son eventos que se derivan directamente de una reserva. Referenciar la reserva en una estancia permite mantener una clara relación entre la estancia y todos los detalles de la reserva (incluyendo la habitación y el tipo de habitación). Esto garantiza que cualquier cambio en la reserva se refleje automáticamente en la información de la estancia.

- Ventajas: Se mantiene un enlace claro entre la estancia y su reserva correspondiente. Las actualizaciones en la reserva se reflejan en la estancia sin duplicación de datos.

- Consideraciones: Al igual que con las reservas, las consultas para recuperar información completa de estancias pueden requerir seguir múltiples referencias.

CuentaServicio (referenciando Reserva):

- ¿Por qué referenciado? Las estancias son eventos que se derivan directamente de una cuenta. Referenciar la reserva en una cuenta permite mantener una clara relación entre la cuenta y todos los detalles de la reserva (incluyendo la habitación y el tipo de habitación). Esto garantiza que cualquier cambio en la reserva se refleje automáticamente en la información de la estancia.

- Ventajas: Se mantiene un enlace claro entre la cuenta y su reserva correspondiente. Las actualizaciones en la reserva se reflejan en la estancia sin duplicación de datos.

- Consideraciones: Al igual que con las reservas, las consultas para recuperar información completa de estancias pueden requerir seguir múltiples referencias.

#### 4. Ejmplo documentos:

```
db.tipoHabitacion.insertOne({
  "_id": "1",
  "nombre": "Suite Presidencial",
  "capacidad": 4,
  "dotacion": "Cama King, Baño con Jacuzzi, Vista al mar"
})
```

```
db.habitacion.insertOne({
  "_id": "1",
  "costo": 250,
  "tipoHabitacion": db.tipoHabitacion.findOne({ "_id": "1" })
})
```

```
db.reserva.insertOne({
  "_id": "3",
  "fechaLlegada": ISODate("2023-12-24T00:00:00Z"),
  "fechaSalida": ISODate("2023-12-31T00:00:00Z"),
  "tipoHabitacion": db.tipoHabitacion.findOne({ "_id": "1" })
  "titular": "Juan Pérez",
  "habitacion": db.habitacion.findOne({ "_id": "3" })
})
```

```
db.estadia.insertOne({
  "_id": "1",
  "fechaCheckIn": ISODate("2024-01-01T00:00:00Z"),
  "fechaCheckOut": ISODate("2024-01-07T00:00:00Z"),
  "checkInRealizado": false,
  "checkOutRealizado": false,
  "reserva": db.reserva.findOne({ "_id": "3" })
})
```

##### 5. Creación de las colecciones:

```
db.createCollection("tipoHabitacion")
db.createCollection("habitacion")
db.createCollection("reserva")
db.createCollection("estadia")
```

```
db.createCollection("cuentaServicio", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["descripcion", "fecha", "reservaId", "servicioId"],
      properties: {
        descripcion: {
          bsonType: "string"
        },
        fecha: {
          bsonType: "date"
        },
        reservaId: {
          bsonType: "objectId",
          description: "Referencia a la reserva"
        },
        servicioId: {
          bsonType: "objectId",
          description: "Referencia al servicio o producto"
        }
      }
    }
  }
});
```

```

db.createCollection("hotel", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["nombre", "habitaciones", "serviciosProductos"],
      properties: {
        nombre: {
          bsonType: "string",
          description: "Nombre del hotel"
        },
        habitaciones: {
          bsonType: "array",
          description: "Lista de habitaciones en el hotel",
          items: {
            bsonType: "object",
            required: ["id", "descripcion", "tipoHabitacionId"],
            properties: {
              id: {
                bsonType: "int",
                description: "ID único de la habitación"
              },
              descripcion: {
                bsonType: "string",
                description: "Descripción de la habitación"
              },
              tipoHabitacionId: {
                bsonType: "objectId",
                description: "Referencia al tipo de habitación"
              }
            }
          }
        },
        serviciosProductos: {
          bsonType: "array",
          items: {
            bsonType: "objectId",
            description: "Referencia a servicios o productos"
          }
        }
      }
    }
  }
});

```

## 6. Validaciones esquema

```

const tipoHabitacionSchema = new mongoose.Schema({
  _id: String,
  nombre: String,
  capacidad: Number,
  dotacion: String,
});

const TipoHabitacion = mongoose.model('TipoHabitacion', tipoHabitacionSchema);

```

```
const habitacionSchema = new mongoose.Schema({
  _id: String,
  costo: Number,
  tipoHabitacion: {
    type: String,
    ref: 'TipoHabitacion',
  },
});

const Habitacion = mongoose.model('Habitacion', habitacionSchema);
```

```
db.createCollection("hotel", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["nombre", "habitaciones", "serviciosProductos"],
      properties: {
        nombre: {
          bsonType: "string",
          description: "Nombre del hotel"
        },
        habitaciones: {
          bsonType: "array",
          description: "Lista de habitaciones en el hotel",
          items: {
            bsonType: "object",
            required: ["id", "descripcion", "tipoHabitacionId"],
            properties: {
              id: {
                bsonType: "int",
                description: "ID único de la habitación"
              },
              descripcion: {
                bsonType: "string",
                description: "Descripción de la habitación"
              },
              tipoHabitacionId: {
                bsonType: "objectId",
                description: "Referencia al tipo de habitación"
              }
            }
          }
        },
        serviciosProductos: {
          bsonType: "array",
          items: {
            bsonType: "objectId",
            description: "Referencia a servicios o productos"
          }
        }
      }
    }
  }
});
```



```
db.createCollection("cuentaServicio", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["descripcion", "fecha", "reservaId", "servicioId"],
      properties: {
        descripcion: {
          bsonType: "string"
        },
        fecha: {
          bsonType: "date"
        },
        reservaId: {
          bsonType: "objectId",
          description: "Referencia a la reserva"
        },
        servicioId: {
          bsonType: "objectId",
          description: "Referencia al servicio o producto"
        }
      }
    }
  }
});
```

```
const reservaSchema = new mongoose.Schema({
  _id: String,
  fechaLlegada: Date,
  fechaSalida: Date,
  tipoHabitacion: {
    type: String,
    ref: 'TipoHabitacion',
  },
  titular: String,
  habitacion: {
    type: String,
    ref: 'Habitacion',
  },
});

const Reserva = mongoose.model('Reserva', reservaSchema);
```

```
const estadiaSchema = new mongoose.Schema({
  _id: String,
  fechaCheckIn: Date,
  fechaCheckOut: Date,
  checkInRealizado: Boolean,
  checkOutRealizado: Boolean,
  reserva: {
    type: String,
    ref: 'Reserva',
  },
});

const Estadia = mongoose.model('Estadia', estadiaSchema);
```

```
module.exports = {
  TipoHabitacion,
  Habitacion,
  Reserva,
  Estadia,
};
```

## ISIS2304B08202320.cuentaServicio

[Documents](#)[Aggregations](#)[Schema](#)[Indexes](#)[Validation](#)

Validation Action ⓘ

Error ▼

Validation Level ⓘ

Strict ▼

```
1  {
2    $jsonSchema: {
3      bsonType: 'object',
4      required: [
5        'descripcion',
6        'fecha',
7        'reservaId',
8        'servicioId'
9      ],
10   properties: {
11     descripcion: {
12       bsonType: 'string'
13     },
14     fecha: {
15       bsonType: 'date'
16     },
17     reservaId: {
18       bsonType: 'objectId',
19       description: 'Referencia a la reserva'
20     },
21     servicioId: {
22       bsonType: 'objectId',
23       description: 'Referencia al servicio o producto'
24     }
25   }
26 }
27 }
```

### 7. Escenarios de prueba para todos los req

Req 1:

### Eliminar y mostrar

### Tipos de Habitaciones

Crear Nuevo Tipo Habitación

Doble

Capacidad: 2

Dotación: Bar, Tv y Cocina

Eliminar

Editar

Sencilla

Capacidad: 1

Dotación: Cama individual, TV, Baño todo

Eliminar

Editar

### editar:

### Editar Tipo de Habitación

Nombre:

Doble

Capacidad:

2

Dotación:

Bar, Tv y Cocina

Actualizar

### crear

### Crear Nuevo Tipo de Habitación

Nombre:

Capacidad:

0

Dotación:

Crear

Req 2:

Eliminar y mostrar:

### Habitaciones

Crear Nueva Habitación

3

Costo: 100.0

Detalles del Tipo de Habitación

Tipo Habitación: Sencilla  
Capacidad: 1  
Dotación: Cama individual, TV, Baño todo

Eliminar

Editar

4

Crear:

### Crear Nueva Habitación

Costo:

0,0

Tipo de Habitación:

Doble

Crear

Editar:

Editar Habitación

Costo:

100,0

Tipo de Habitación:

Doble

Actualizar

Req 3:

Servicios y Productos

Nombre	Precio	Es Producto	Duración
Arroz con pollo	57844	Sí	0
Arroz con verduras	66374	Sí	0
Takoyaki	17523	Sí	0
Arroz con mariscos y carne	96201	Sí	0
Facial Treatment	67478	Sí	51
Facial Care	10171	Sí	18
	0	No	0
jsakdnksja	1	Sí	69

Crear Nuevo Servicio o Producto

Req 4 y 7 estadía:  
Mostrar eliminar

### Estadias

Crear Nueva Estadia

<div>Estadia ID: 1</div> <div>Fecha de Check-In: 2024-11-10</div> <div>Fecha de Check-Out: 2024-12-12</div> <div>Check-In Realizado: Si</div> <div>Check-Out Realizado: No</div> <div>ID de Reserva: 4</div> <div><div>Editar</div><div>Eliminar</div></div>
<div>Estadia ID: 2</div> <div>Fecha de Check-In: 2023-02-11</div> <div>Fecha de Check-Out: 2023-05-15</div> <div>Check-In Realizado: No</div> <div>Check-Out Realizado: No</div>

### Crear:

### Crear Nueva Estadia

Fecha de Check-In:

dd/mm/aaaa

Fecha de Check-Out:

dd/mm/aaaa

☐ Check-In Realizado

☐ Check-Out Realizado

Reserva:

4

Crear

Editar:

Hotel Andes

Tipos de HabitaciónHabitacionesReservasEstadías

### Editar Estadía

Fecha de Check-In:

dd/mm/aaaa

Fecha de Check-Out:

dd/mm/aaaa

☒ Check-In Realizado

☐ Check-Out Realizado

Reserva:

4

Actualizar

Req 6:

Hotel Andes

Tipos de HabitaciónHabitacionesReservasEstadíasCuenta ServicioServicios Productos

Cuentas de Servicio

Crear Nueva Cuenta de Servicio

ID	Descripción	Fecha	Reserva ID	Servicio ID
656d4e5533aaed0f1d371f95	Reserva de habitación doble	2023-01-15	656d4e5533aaed0f1d371f81	656d4e5533aaed0f1d371f82
656d4e5533aaed0f1d371f96	Alquiler de bicicletas	2023-02-20	656d4e5533aaed0f1d371f83	656d4e5533aaed0f1d371f84
656d4e5533aaed0f1d371f97	Tour guiado por la ciudad	2023-03-05	656d4e5533aaed0f1d371f85	656d4e5533aaed0f1d371f86

Req 2 consultas:

Hotel Andes

Tipos de HabitaciónHabitacionesReservasEstadías

Ocupación de Habitaciones

ID de Habitación	Porcentaje de Ocupación (%)
	0.0

8. Escenarios de prueba



- a. Arriba están escenarios de que se puede realizar todo el CRUD y en la sustentación se demostrara
- b. Para el siguiente documento incumple entonces sale este error:  
E11000 error collection: ISIS2304B08202320.tipoHabitacion

```
db.tipoHabitacion.insertOne({  
  "_id": "2",  
  "nombre": "Habitación Estándar"  
  // Falta 'capacidad' y 'dotacion'  
})
```

```
db.reserva.insertOne({  
  "_id": "4",  
  "fechaLlegada": ISODate("2023-12-15T00:00:00Z"),  
  "fechaSalida": ISODate("2023-12-20T00:00:00Z"),  
  "tipoHabitacion": db.tipoHabitacion.findOne({ "_id": "1" }),  
  "titular": "Ana Gómez",  
  "habitacion": db.habitacion.findOne({ "_id": "99" }) // Esta habitación no existe  
})
```

```
db.estadia.insertOne({  
  "_id": "2",  
  "fechaCheckIn": ISODate("2024-02-01T00:00:00Z"),  
  "fechaCheckOut": ISODate("2024-02-07T00:00:00Z"),  
  "checkInRealizado": false,  
  "checkOutRealizado": false,  
  "reserva": db.reserva.findOne({ "_id": "99" }) // Esta reserva no existe  
})
```

Asi para cada una de las colecciones

C. script utilizado

```

db.reserva.aggregate([
  {
    $match: {
      fechaLlegada: { $gte: ISODate("2023-01-01T00:00:00Z") },
      fechaSalida: { $lte: ISODate("2023-12-31T23:59:59Z") }
    }
  },
  {
    $project: {
      habitacion: "$habitacion.$id",
      duracionReserva: {
        $divide: [
          { $subtract: ["$fechaSalida", "$fechaLlegada"] },
          1000 * 60 * 60 * 24
        ]
      }
    }
  },
  {
    $group: {
      _id: "$habitacion",
      totalDiasOcupados: { $sum: "$duracionReserva" }
    }
  },
  {
    $project: {
      _id: 0,
      habitacion: "$_id",
      ocupacion: {
        $multiply: [
          { $divide: ["$totalDiasOcupados", 365] },
          100
        ]
      }
    }
  }
]);

```