PARKING LOT SHARING

Make money from your free driveways

Jakrapop Akaranee (st121075)
Tran Hoang Minh (st120761)
Kunlanit Korsamphan (st121422)
Phway Thant Thant Soe Lin (st121494)

PROJECT SUMMARY

PROBLEM AND SOLUTION:

Too many cars, too little parking lots.

PROBLEMS

Make money from your free driveways. Save money for cheaper parking lot.

SOLUTION

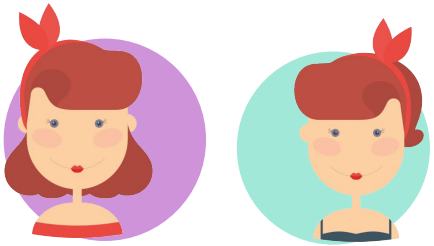
HOW DOES IT WORK?

Search available spaces by the area

Book it!

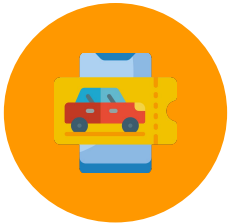
Share your experience to others

SCOPES:

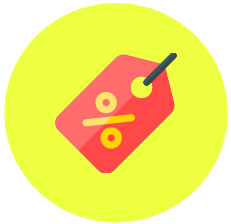


Users: Providers & Renters

Parking Lots: Owned/ Favorited/Reviewed



Booking: Short-term/ Long-term Parking



Discount: Receive and use or share to others

FEATURES:

Search by area / filter by preferences

Display on the map

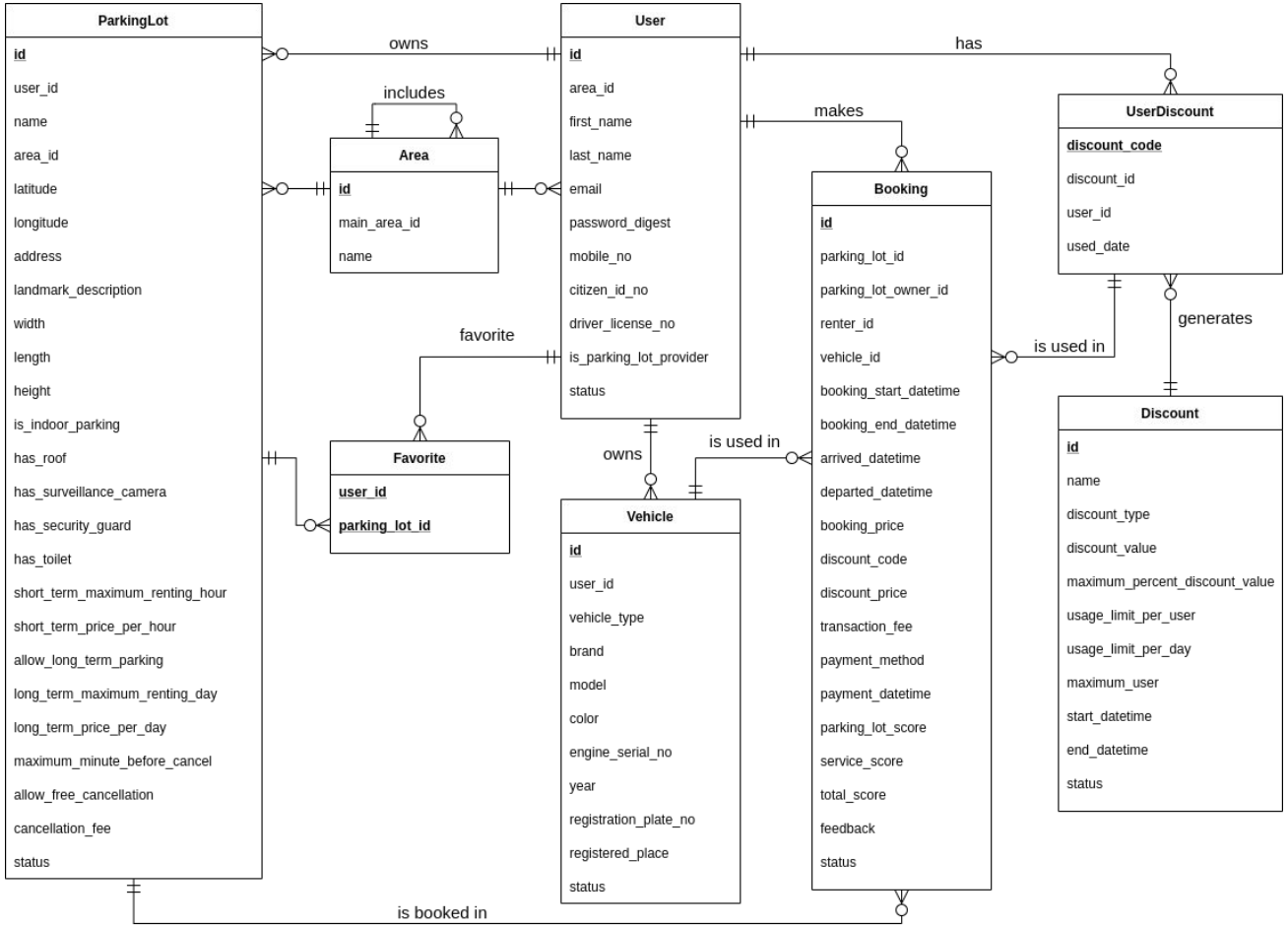
Save to your favorite list

Book instantly or make an advance booking

Share your discount code with friends and customers

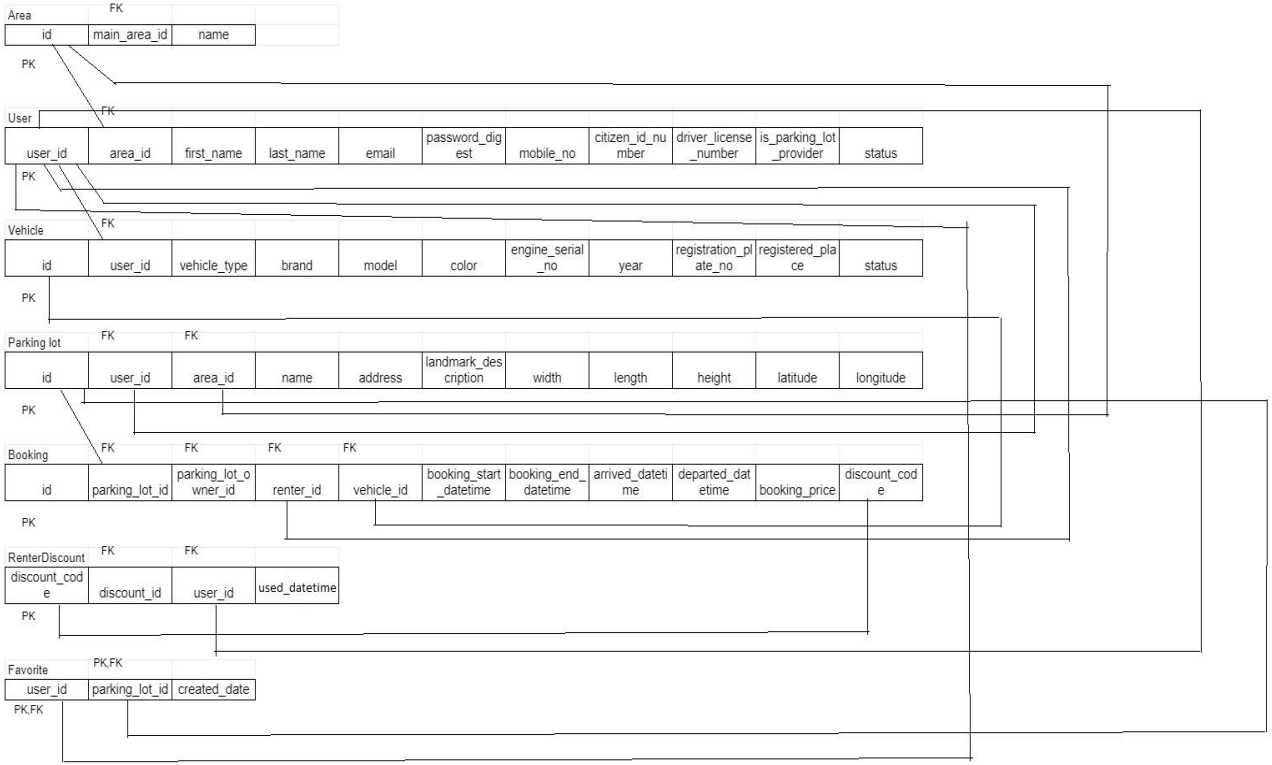
Share your opinion and experience with everyone

ER DIAGRAM


ParkingLot
id
user_id
name
area_id
latitude
longitude
address
landmark_description
width
length
height
is_indoor_parking
has_roof
has_surveillance_camera
has_security_guard
has_toilet
short_term_maximum_renting_hour
short_term_price_per_hour
allow_long_term_parking
long_term_maximum_renting_day
long_term_price_per_day
maximum_minute_before_cancel
allow_free_cancellation
cancellation_fee
status
owns
includes
Area
id
main_area_id
name
User
id
area_id
first_name
last_name
email
password_digest
mobile_no
citizen_id_no
driver_license_no
is_parking_lot_provider
status
has
makes
UserDiscount
discount_code
discount_id
user_id
used_date
Booking
id
parking_lot_id
parking_lot_owner_id
renter_id
vehicle_id
booking_start_datetime
booking_end_datetime
arrived_datetime
departed_datetime
booking_price
discount_code
discount_price
transaction_fee
payment_method
payment_datetime
parking_lot_score
service_score
total_score
feedback
status
is used in
generates
Discount
id
name
discount_type
discount_value
maximum_percent_discount_value
usage_limit_per_user
usage_limit_per_day
maximum_user
start_datetime
end_datetime
status
favorite
Favorite
user_id
parking_lot_id
owns
is used in
Vehicle
id
user_id
vehicle_type
brand
model
color
engine_serial_no
year
registration_plate_no
registered_place
status
is booked in

SQL DB MODELLING RECAP

REFERENTIAL INTEGRITY CONSTRAINTS



DATA OPERATION AND INQUIRY

- Normal CRUD Operation for each models
- Search available parking lot which match require conditions.
- View parking lot comment/review and rating
- View user's booking history
- View favorite parking lot list
- Report of provider's total income
- Report of the popular area
- Report of the top 5 best review parking lot
- Report of average price (short term and long term) per area
- Report of total income from all parking lots in each area in Bangkok

DOCUMENT DB MODELLING

COLLECTION: AREA

```
{
        id: {{object_id}},
        name: {{string}},
        main_area_id: {{object_id}}
}
```

```
{
        id: "5f8ef4a55202b4b6cb63b1d7",
        name: "Bangkok",
        main_area_id: "5f8ef4a5dbd37598ff6aba92"
}
```

COLLECTION: USERS

```
{
        id: {{object_id}},
        first_name: {{string}},
        last_name: {{string}},
        email: {{string}},
        mobile_no: {{string}},
        citizen_id_no: {{string}},
        driver_license_no: {{string}},
        Is_parking_lot_provider: {{bool}},
        area_id: {{object_id}},
        favorites: [ {string} ],
        status,
        ...
}
```

```
{
        id: "5f8ef4a55202b4b6cb63b1d7",
        first_name: "Jane",
        last_name: "Doe",
        email: "jane.d@mail.com",
        mobile_no: "0812345678",
        citizen_id_no: "1234567890987",
        driver_license_no: "1234567890",
        Is_parking_lot_provider: false,
        area_id: "5f8ef4a5dbd37598ff6aba92",
        favorites: ["5f8ef4a5dbd37598ff6aba92"
                "5f8ef4a5dbd37598ff6aba92"],
        status: "Active"
        ...
}
```

COLLECTION: USERS

```
{
        ...
        vehicle: [
                {
                        id:{{object_id}},
                        vehicle_type:{{string}},
                        brand:{{string}},
                        model:{{string}},
                        color:{{string}},
                        engine_serial_no:{{string}},
                        year:{{string]},
                        registration_plate_no:{{string}},
                        registered_place:{{string}},
                        status: {{string}}
                }
        ],
        ...
}
```

```
{
        ...
        vehicle: [
                {
                        id: “5f8ef4a55202b4b6cb63b1d7”,
                        vehicle_type: “suv”,
                        brand:”Toyota”,
                        model:”belta”,
                        color:”white”,
                        engine_serial_no: “5Be3c6”,
                        year:”2018”,
                        registration_plate:”ABC111”,
                        registered_place:”Bangkok”,
                        status:”Active”
                }
        ],
        ...
}
```

COLLECTION: PARKING LOTS

```
{
        id: {{object_id}},
        owner_id: {{object_id}},
        name: {{string}},
        latitude: {{string}},
        longitude: {{string}},
        address: {{string}},
        landmark_description: {{string}},
        width: {{integer}},
        length: {{integer}},
        height: {{integer}},
        ...
}
```

```
{
        id: "5f8ef4a55202b4b6cb63b1d7",
        owner_id: "5f8ef4a55202b4b6cb63b1d7",
        name: "My Parking Lot",
        latitude: "13.567098",
        longitude: "100.345678",
        address: "2365/10 Sukhumvit 77 Rd.",
        landmark_description: "Near Onnut BTS",
        width: 2,
        length: 3,
        height: 4,
        ...
}
```

COLLECTION: PARKING LOTS

```
{
        ...
        facilities: [{{string}}],
        short_term_max_rent_hour: {{integer}},
        short_term_price_per_hour: {{integer}},
        allow_long_term: {{bool}},
        long_term_max_rent_day: {{integer}},
        long_term_price_per_day: {{integer}},
        allow_cancel: {{bool}},
        cancel_fee: {{float}},
        area: {
                id: {{object_id}},
                name: {{string}}
        }
}
```

```
{
        ...
        facilities: ["has_roof", "has_camera", "has_guard"],
        short_term_max_rent_hour: 8,
        short_term_price_per_hour: 5,
        allow_long_term: true,
        long_term_max_rent_day: 120,
        long_term_price_per_day: 40,
        allow_cancel: true,
        cancel_fee: 0.0,
        area: {
                id: "5f8ef4a55202b4b6cb63b1d7",
                name: "Onnut"
        }
}
```

COLLECTION: BOOKINGS

```
{
        id: {{object_id}},
        booking_start_datetime: {{date_time}},
        booking_end_datetime: {{date_time}},
        arrival_datetime: {{date_time}},
        departure_datetime: {{date_time}},
        booking_price: {{float}},
        discount_code: {{string}},
        discount_price: {{float}},
        transaction_fee: {{float}},
        payment_method: {{string}},
        payment_date_time: {{date_time}},
        parking_lot_score: {{float}},
        ...
}
```

```
{
        id: "5f8ef4a55202b4b6cb63b1d7",
        booking_start_datetime: "09-10-2020 10:00:00",
        booking_end_datetime: "09-10-2020 12:00:00",
        arrival_datetime: "09-10-2020 10:01:00",
        departure_datetime: "09-10-2020 11:50:00",
        booking_price: 12.00,
        discount_code: null,
        discount_price: 0.00,
        transaction_fee: 1.8,
        payment_method: "cash",
        payment_date_time: "09-10-2020 11:51:03",
        parking_lot_score: 5,
        ...
}
```

COLLECTION: BOOKINGS

```
{
        ...
        service_score: {{float}},
        total_score: {{float}},
        feedback: {{string}},
        status: {{string}},
        parking_lot: {
                id: {{object_id}},
                owner_id: {{object_id}},
                area_id: {{object_id}},
                name: {{string}},
                facilities: [{{string}}],
        },
        ...
}
```

```
{
        ...
        service_score: 5,
        total_score: 5,
        feedback: "Clean",
        status: "completed",
        parking_lot: {
                id: "5f8ef4a55202b4b6cb63b1d7",
                owner_id: "5f8ef4a55202b4b6cb63b1d7",
                area_id: "5f8ef4a55202b4b6cb63b1d7",
                name: "My Parking Lot",
                facilities: ["has_roof", "has_camera",
                "has_guard"],
        },
        ...
}
```

COLLECTION: BOOKINGS

```
{
        ...
        renter: {
                id: {{object_id}},
                first_name: {{string}},
                last_name: {{string}},
                mobile_no: {{string}},
                vehicle_id: {{object_id}},
                vehicle_brand: {{string}},
                vehicle_model: {{string}},
                vehicle_color: {{string}},
                vehicle_registration_plate_no: {{string}},
                vehicle_registered_place: {{string}},
        }
}
```

```
{
        ...
        renter: {
                id: “5f8ef4a55202b4b6cb63b1d7”,
                first_name: “Jane”,
                last_name: “Doe”,
                mobile_no: “0812345678”,
                vehicle_id: “5f8ef4a55202b4b6cb63b1d7”,
                vehicle_brand: “Honda”,
                vehicle_model: “Jazz”,
                vehicle_color: “red”,
                vehicle_registration_plate_no: “AA9999”,
                vehicle_registered_place: “Bangkok”,
        }
}
```

COLLECTION: USER DISCOUNTS

```
{
        id: {{object_id}},
        discount_id: {{object_id}},
        code: {{string}},
        owner_id: {{object_id}},
        user_id: {{object_id}},
        used_date: {{date_time]}
}
```

```
{
        id: “5f8ef4a5545t7jf6cb63b1d7”,
        discount_id: “5f8ef4a5545t7jf6cb63b1d7”,
        code: “BT56GG”,
        owner_id: “5f8ef4a5545t7jf6cb63b1d7”,
        user_id: “5f8ef4a5545t7jf6cb63b1d7”,
        used_date: “09-10-2020”,
}
```

COLLECTION: DISCOUNTS

```
{
        id: {{object_id}},
        name: {{string}},
        discout_type: {{string}},
        discount_value: {{Int32}},
        maximum_percent_discount_rate: {{Int32}},
        usage_limit_per_user: {{Int32}},
        usage_limit_per_day: {{Int32}},
        maximum_user: {{Int32}},
        start_datetime: {{date_time}},
        end_datetime: {{date_time}},
}
```

```
{
        id: "5f8ef4a55202b4b6cb63b1d8",
        name: "Discount 5 percent",
        discout_type: "percent",
        discount_value: 5,
        maximum_percent_discount_rate: 20,
        usage_limit_per_user: 2,
        usage_limit_per_day: 10,
        maximum_user: 50,
        start_datetime: '20-10-2020',
        end_datetime: '25-10-2020',
}
```

DOCUMENT DB MODELLING: SUMMARY

- Vehicles is embedded in Users collection as we see that the number of the data per users is small and rarely change. It's also essential to get the data at the same time with user information when they want to create a booking.
- Favorites is a field. We use reference method to link between users and their favorite parking lots by keeping the ID of parking lots in array type data.
- Parking Lot data and Renter data is partially embedded in Booking collection. Both data are needed to display the detail of the booking.

DATA OPERATION & INQUIRY

BASIC OPERATIONS:

- Create an area
- Update an area
- Add a renter and vehicles
- Update a renter's vehicles
- Delete a user
- Add a renter's favorite parking lot
- A renter's favorite parking lot list
- Create a parking lot
- Update a parking lot
- View a provider's parking lots
- Delete a parking lot
- Search parking lots by conditions
- Create a booking
- Cancel a booking
- Update a booking's review
- View a user's booking history
- Create a discount
- Update a discount
- Search active discounts
- Give a discount to a user

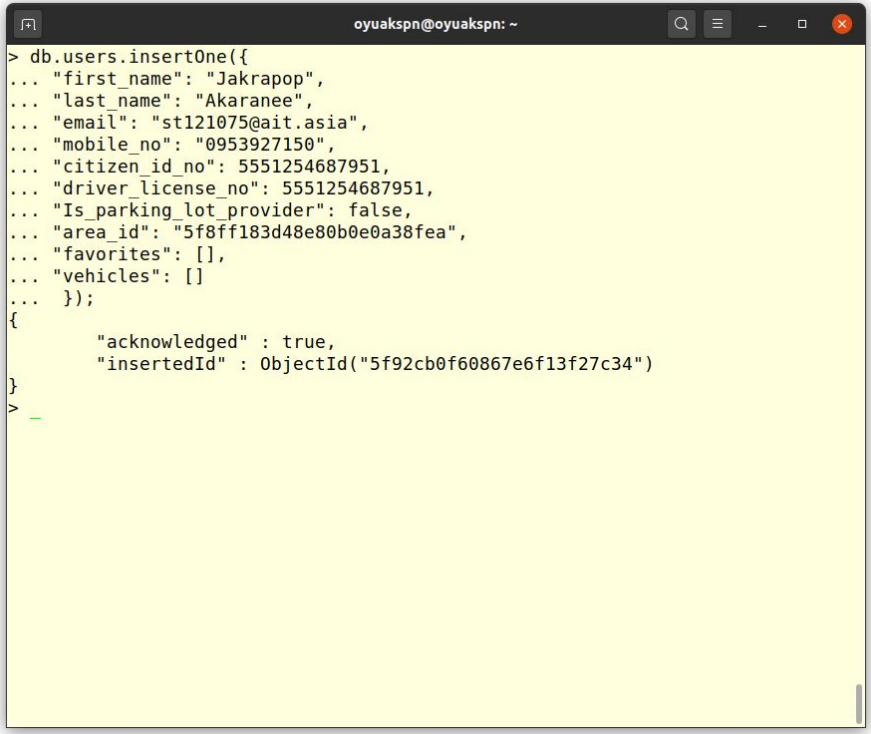
BASIC OPERATIONS:
CREATE AN AREA


```
db.area.insertOne(
  {
      main_area_id: null,
      name: "Chiang Mai"
  }
)
```

BASIC OPERATIONS:
UPDATE AN AREA

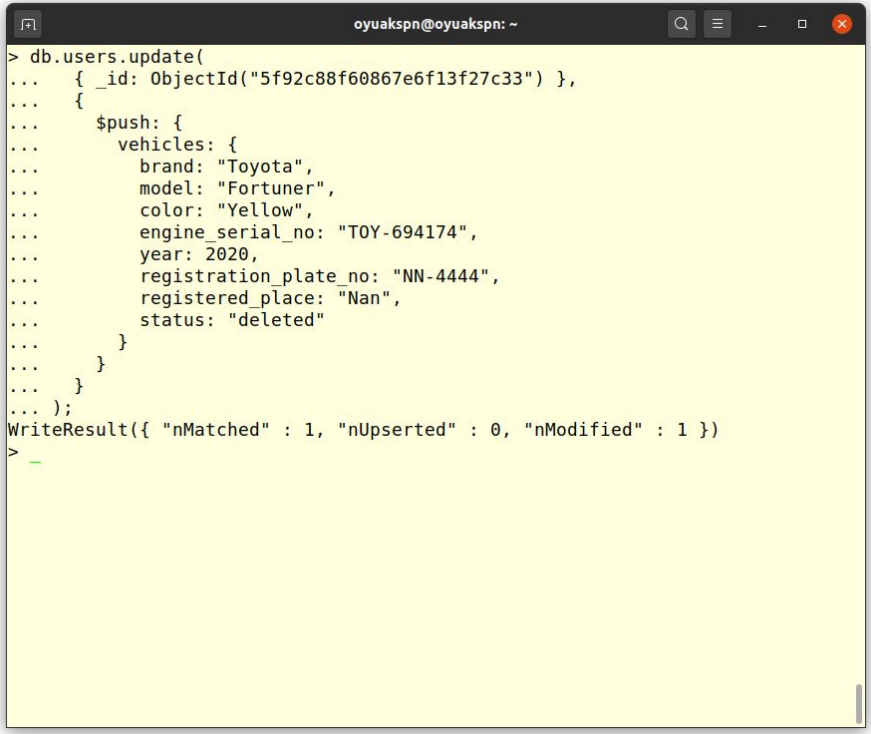

```
db.areas.update(
  { _id: ObjectId("5f8fe32ee6322b1ee92b6d10") },
  { $set: { status: "Bangkok" } }
)
```

BASIC OPERATIONS:
ADD A RENTER AND VEHICLES




```
db.users.insertOne({
        "first_name": "Jakrapop",
        "last_name": "Akaranee",
        "email": "st121075@ait.asia",
        "mobile_no": "0953927150",
        "citizen_id_no": 5551254687951,
        "driver_license_no": 5551254687951,
        "Is_parking_lot_provider": false,
        "area_id": "5f8ff183d48e80b0e0a38fea",
        "favorites": [],
        "vehicles": []
})
```

BASIC OPERATIONS:
UPDATE A RENTER'S VEHICLES




```
db.users.update(
 { _id: ObjectId("5f92c88f60867e6f13f27c33") },
 {
  $push: {
   vehicles: {
    brand: "Toyota",
    model: "Fortuner",
    color: "Yellow",
    engine_serial_no: "TOY-694174",
    year: 2020,
    registration_plate_no: "NN-4444",
    registered_place: "Nan",
    status: "deleted"
   }
  }
 }
);
```

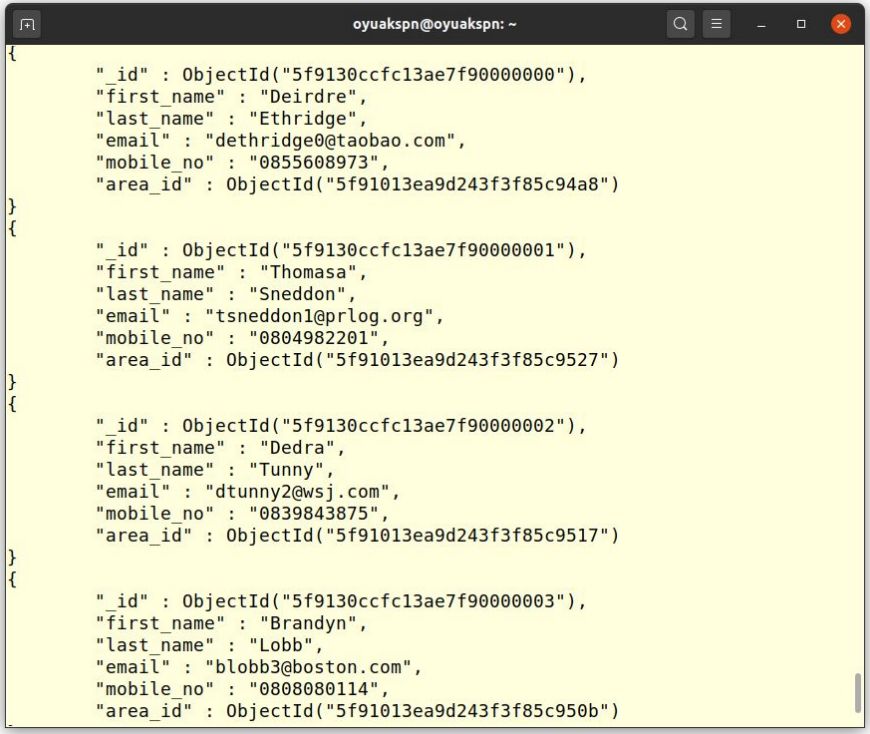
BASIC OPERATIONS:
DELETE A USER

db.users.deleteOne({ id_:ObjectID("5f9130ccfc13ae7f900000ff") })

BASIC OPERATIONS:
ADD A FAVORITE PARKING LOTS

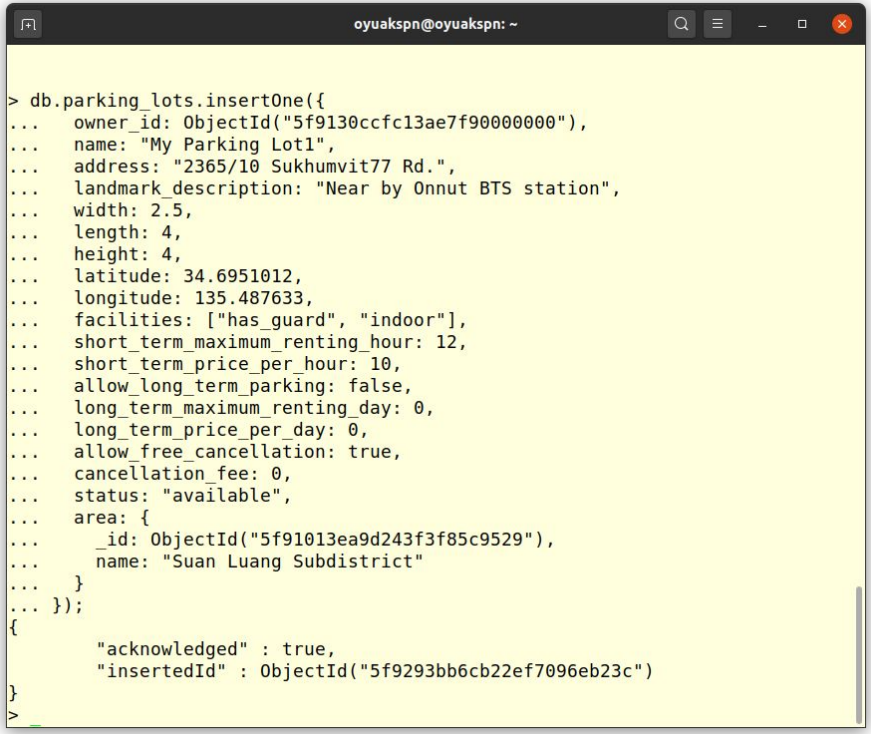

```
db.users.update(
  { _id:ObjectId("5f92881ed2e45fd07b6c48b4")},
  {
    $push: { "favorites" : ObjectId("5f914c48fc13ae1528000008") }
  }
);
```

BASIC OPERATIONS:
A RENTER'S LIST



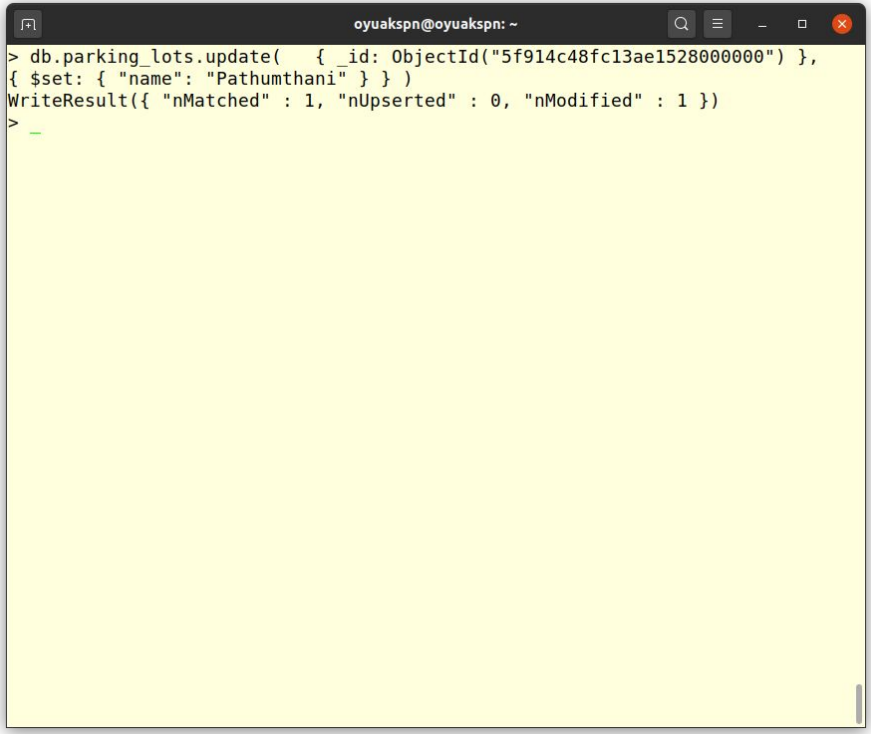

```
db.users.aggregate[{
   $match: {
      is_parking_lot_provider: true
   }
}, {
   $project: {
      first_name: 1,
      second_name: 1,
      mobile_no: 1,
      email: 1,
      area_id: 1
   }
}]
```

BASIC OPERATIONS:
CREATE A PARKING LOT



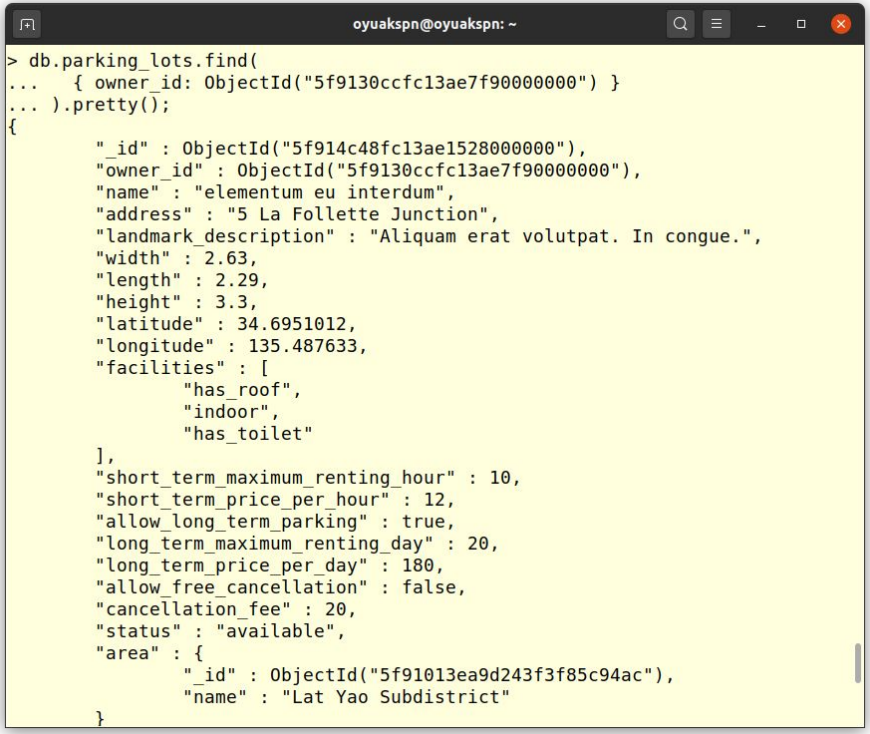

```
db.parking_lots.insertOne(
  {
        owner_id: ObjectId("5f9130ccfc13ae7f90000000"),
        name: "My Parking Lot1",
        address: "2365/10 Sukhumvit77 Rd.",
        landmark_description: "Near by Onnut BTS station",
        width: 2.5,
        length: 4,
        height: 4,
        latitude: 34.6951012,
        longitude: 135.487633,
        facilities: ["has_guard", "indoor"],
        short_term_maximum_renting_hour: 12,
        short_term_price_per_hour: 10,
        allow_long_term_parking: false,
        long_term_maximum_renting_day: 0,
        long_term_price_per_day: 0,
        allow_free_cancellation: true,
        cancellation_fee: 0,
        status: "available",
        area: {
           _id: ObjectId("5f91013ea9d243f3f85c9529"),
           name: "Suan Luang Subdistrict"
    }
})
```

BASIC OPERATIONS:
UPDATE A PARKING LOT



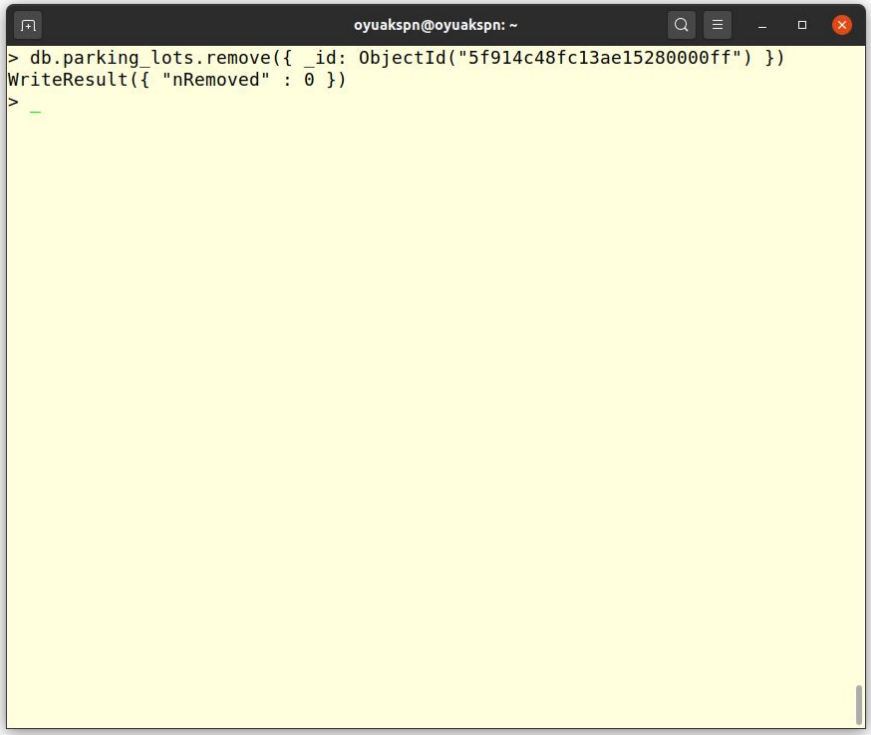

```
db.parking_lots.update(
  { _id: ObjectId("5f914c48fc13ae1528000000") },
  { $set: { "name": "Pathumthani" } }
)
```

BASIC OPERATIONS:
VIEW A PROVIDER'S PARKING LOTS



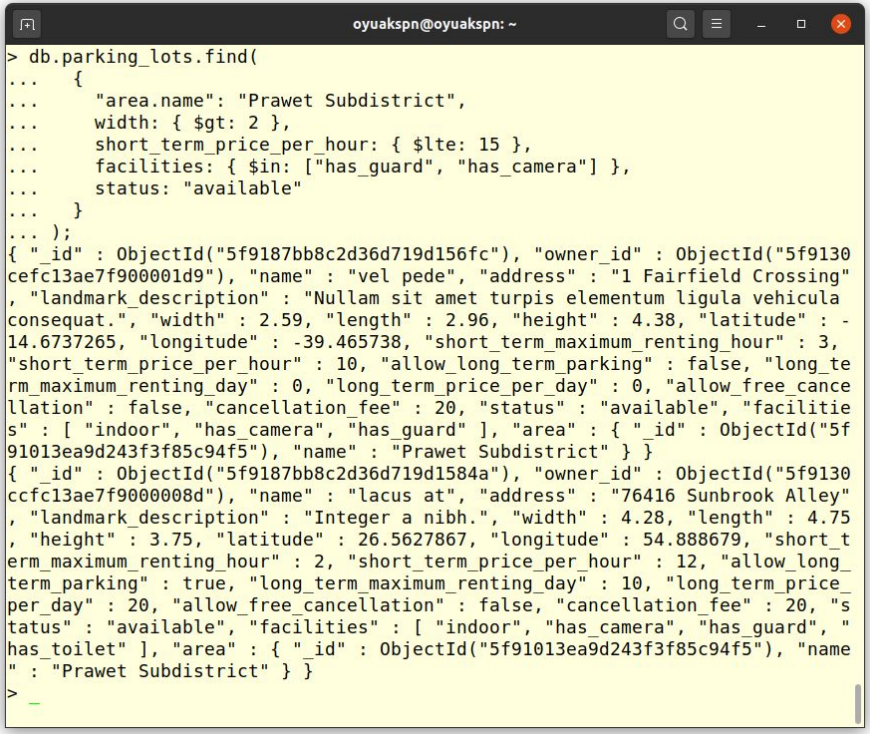

```
db.parking_lots.find(
  {
        owner_id: ObjectId("5f9130ccfc13ae7f90000000")
  }
)
```

BASIC OPERATIONS:
DELETE A PARKING LOT



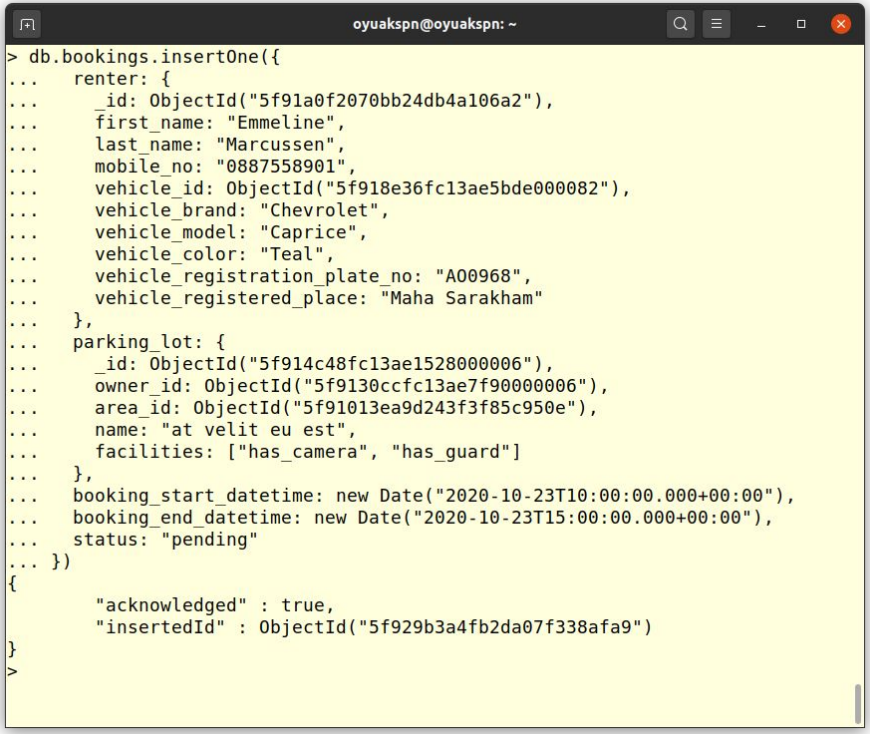

```
db.parking_lots.remove({ _id: ObjectId("5f914c48fc13ae1528000000")
})
```

BASIC OPERATIONS:
SEARCH PARKING LOTS




```
db.parking_lots.find(
  {
        "area.name": "Prawet Subdistrict",
        width: { $gt: 2 },
        short_term_price_per_hour: { $lte: 15 },
        facilities: { $in: ["has_guard","has_camera"] },
        status: "available"
  }
);
```

BASIC OPERATIONS:
CREATE A BOOKING




```
db.bookings.insertOne({
  renter: {
    _id: ObjectId("5f91a0f2070bb24db4a106a2"),
    first_name: "Emmeline",
    last_name: "Marcussen",
    mobile_no: "0887558901",
    vehicle_id: ObjectId("5f918e36fc13ae5bde000082"),
    vehicle_brand: "Chevrolet",
    vehicle_model: "Caprice",
    vehicle_color: "Teal",
    vehicle_registration_plate_no: "AO0968",
    vehicle_registered_place: "Maha Sarakham"
  },
  parking_lot: {
    _id: ObjectId("5f914c48fc13ae1528000006"),
    owner_id: ObjectId("5f9130ccfc13ae7f90000006"),
    area_id: ObjectId("5f91013ea9d243f3f85c950e"),
    name: "at velit eu est",
    facilities: ["has_camera", "has_guard"]
  },
  booking_start_datetime: new
Date("2020-10-23T10:00:00.000+00:00"),
  booking_end_datetime: new
Date("2020-10-23T15:00:00.000+00:00"),
  status: "pending"
})
```

BASIC OPERATIONS:
CANCEL A BOOKING


```
db.bookings.update(
  { _id: ObjectId("5f9230d1070bb24db4a10ec4") },
  { $set: { "status" : "cancelled" } }
)
```

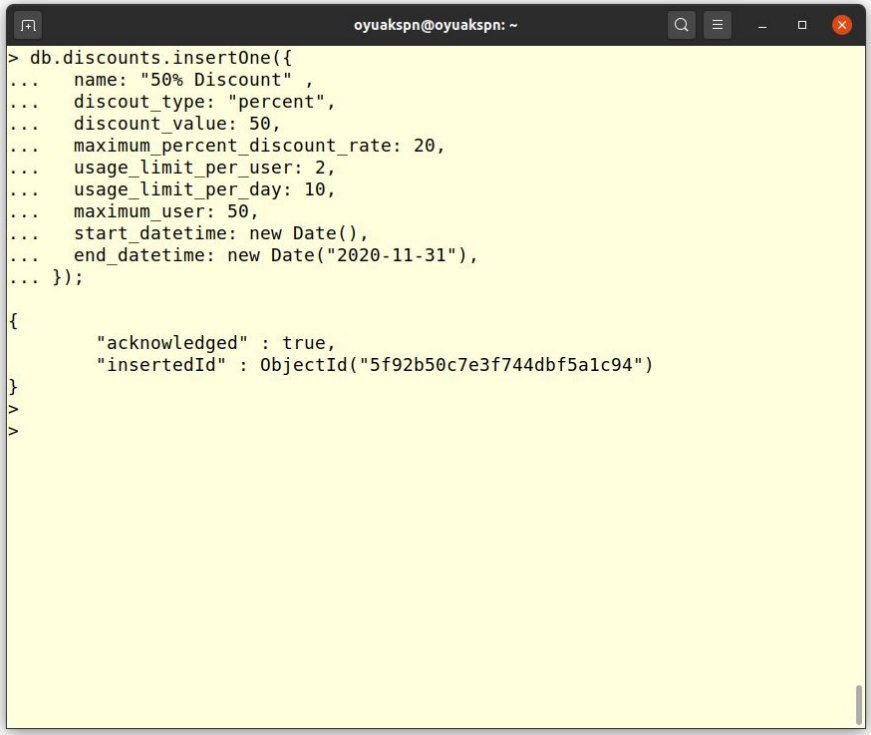
BASIC OPERATIONS:
UPDATE A BOOKING'S REVIEW


```
db.bookings.update(
  { _id: ObjectId("5f914c48fc13ae1528000006") },
  { $set: {
      "feedback": "very good",
      "parking_lot_score": 5,
      "service_score": 5,
      "total_score" : 5
  } }
)
```

BASIC OPERATIONS:
VIEW A USER'S BOOKING HISTORY

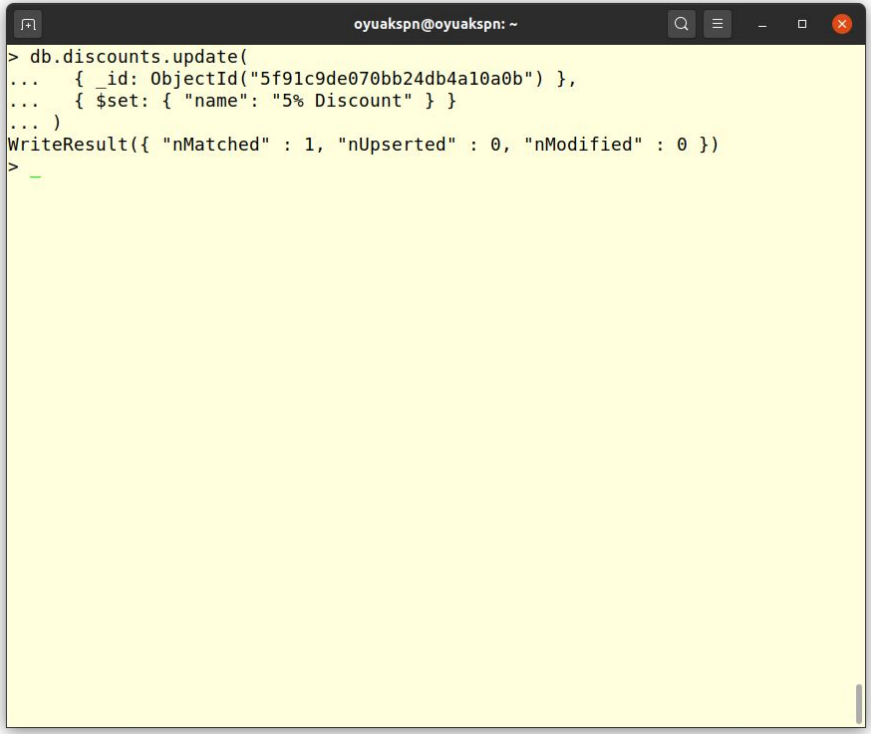

```
db.bookings.aggregate([
 { $match: { "renter._id": ObjectId('5f91a0f2070bb24db4a106a2') } },
 {
  $project: {
     "renter.first_name": 1,
     "renter.last_name": 1,
     "parking_lot._id": 1,
     "renter.vehicle_id": 1,
     booking_start_datetime: 1,
     booking_end_datetime: 1,
     discount_code: 1,
     transaction_fee: 1,
     payment_method: 1,
     payment_datetime: 1
  }
 }
])
```

BASIC OPERATIONS:
CREATE A DISCOUNT



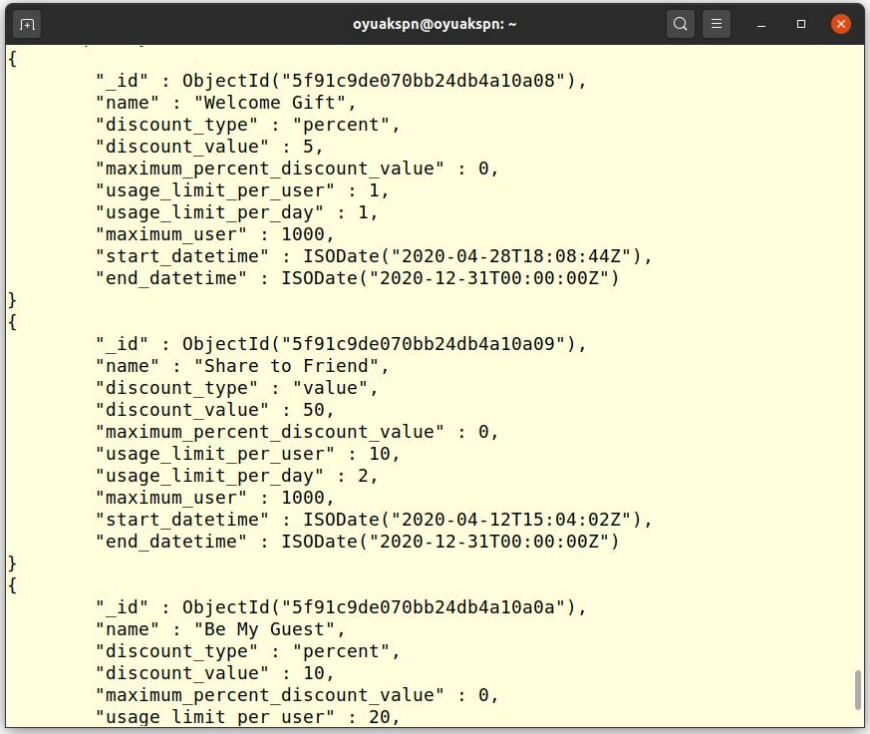

```
db.discounts.insertOne({
  name: "50% Discount" ,
  discout_type: "percent",
  discount_value: 50,
  maximum_percent_discount_rate: 20,
  usage_limit_per_user: 2,
  usage_limit_per_day: 10,
  maximum_user: 50,
  start_datetime: new Date(),
  end_datetime: new Date("2020-11-31"),
});
```

BASIC OPERATIONS:
UPDATE A DISCOUNT



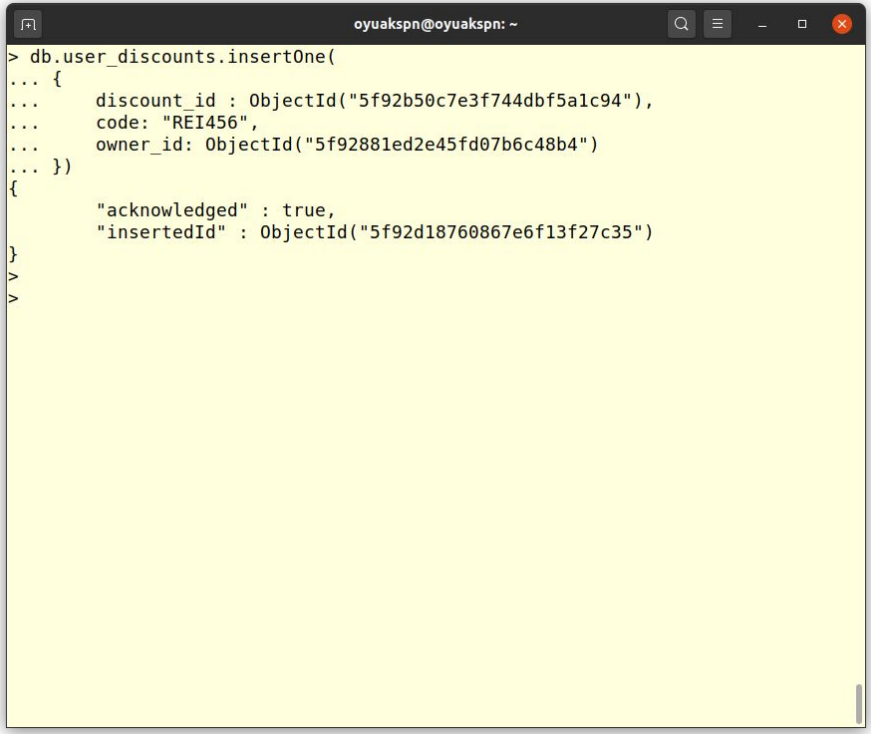

```
db.discounts.update(
  { _id: ObjectId("5f91c9de070bb24db4a10a0b") },
  { $set: { "name": "5% Discount" } }
)
```

BASIC OPERATIONS:
SEARCH ACTIVE DISCOUNTS




```
db.discounts.find(
  { $and: [
    { end_datetime: { $gte: ISODate("2020-11-01T00:00:00.000Z") } },
    { start_datetime:{ $lt: ISODate("2020-10-01T00:00:00.000Z") } }
  ]}
).pretty()
```

BASIC OPERATIONS:
GIVE A DISCOUNT TO USER

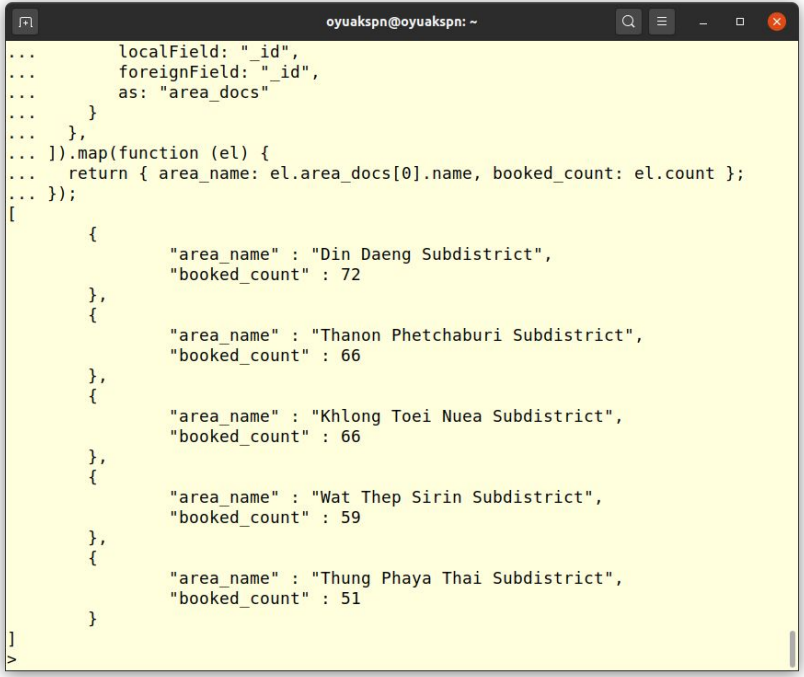



```
db.user_discounts.insertOne(
{
   discount_id : ObjectId("5f92b50c7e3f744dbf5a1c94"),
   code: "REI456",
   owner_id: ObjectId("5f92881ed2e45fd07b6c48b4")
})
```

INTERESTING INQUIRIES:

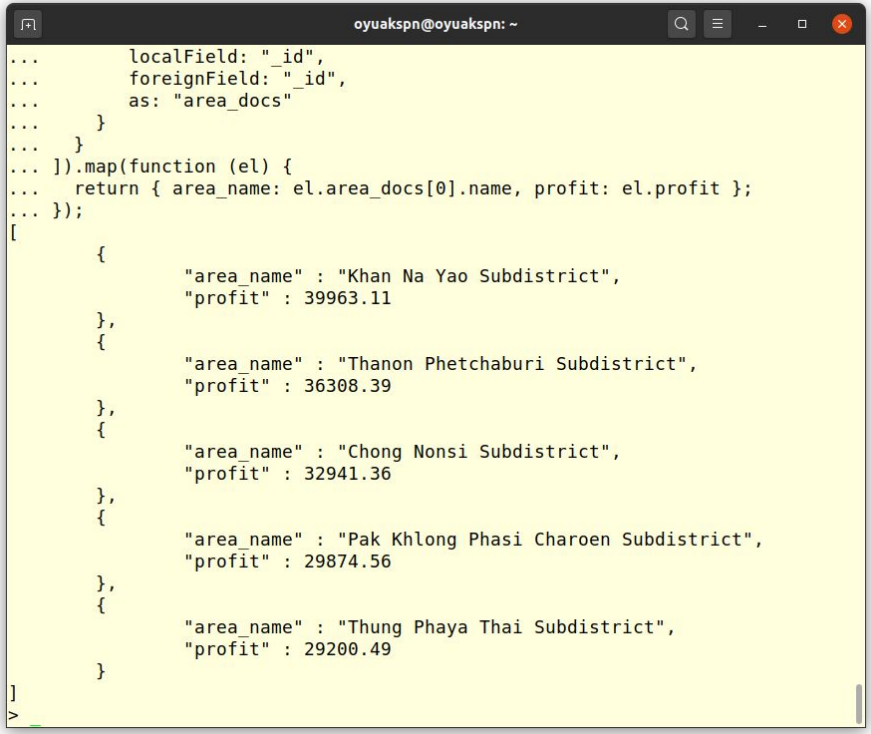
- Top 5 Most Popular Parking Areas
- Top 5 Most Profitable Areas (Provider)
- Average Short-term and Long-term Price per Area
- Average Parking Time per Area (Hours)
- Parking Time Preference Per Area (Day/ Night)
- Top 5 Most Spending Renters
- Top 5 Most Profitable Parking Lot
- Top 10 Popular Vehicle Brands
- The Best Reviewed Parking Lot
- Report of Provider's Total Income
- Report of Parking Lots with Low Facilities Score
- Report of Late Checkout Renters
- Report of One-Year None Active Users
- Report of Most Likely Popular Parking Lot Facilities
- Report of No. of Users Using Discounts
- Report of Top 5 Most-Cancelled Parking Lot
- Report of Free-Cancellation Parking Lot
- Report of Single-Booked Renters
- Report of No. of Users per Area
- Report of No. of Providers

INTERESTING INQUIRIES:
TOP 5 MOST POPULAR PARKING AREAS



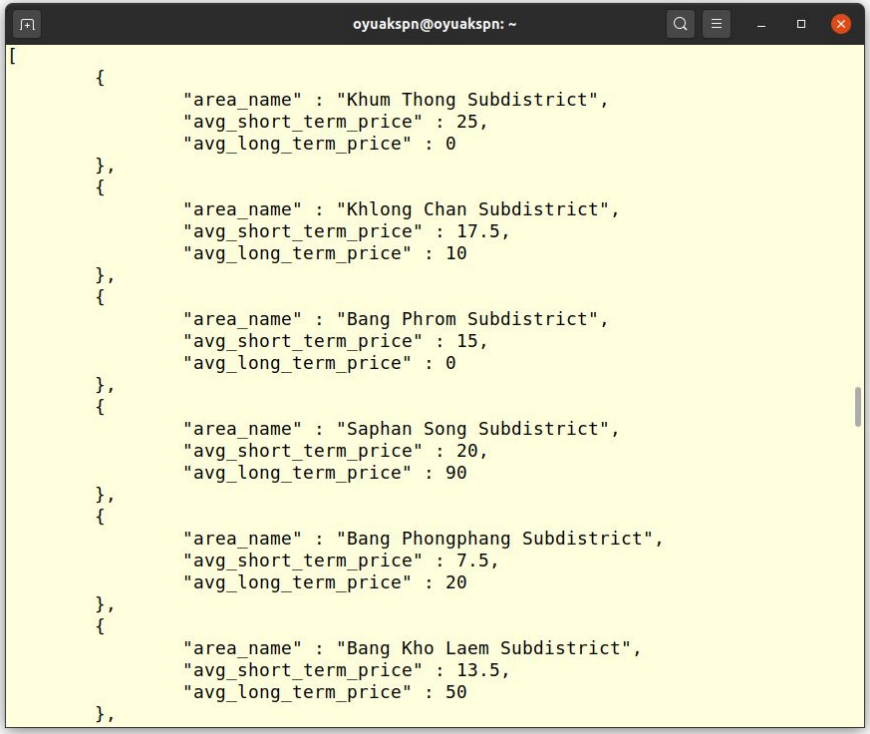

```
db.bookings.aggregate([
 {
  $group: {
   _id: "$parking_lot.area_id",
   count: { $sum: 1 }
  }
 },
 {
  $sort: { count: -1 }
 },
 {
  $limit : 5
 },
 {
  $lookup: {
   from: "areas",
   localField: "_id",
   foreignField: "_id",
   as: "area_docs"
  }
 },
]).map(function (el) {
 return { area_name: el.area_docs[0].name, booked_count: el.count };
});
```

INTERESTING INQUIRIES:
TOP 5 MOST PROFITABLE AREAS



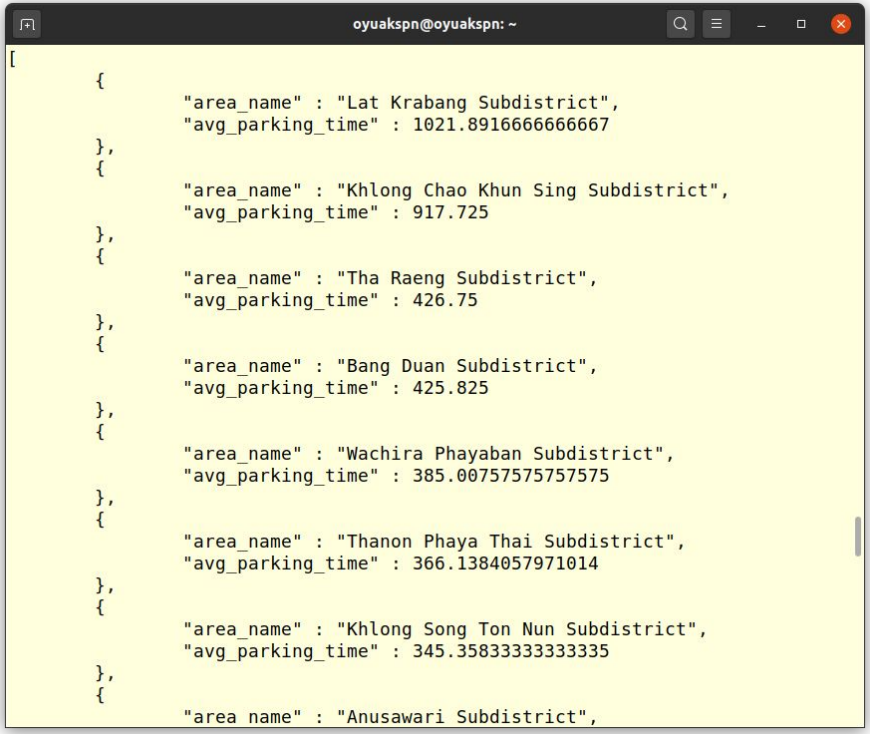

```
db.bookings.aggregate([
 { $match: { status: "completed" } },
 {
  $project: {
   parking_lot: 1,
   net_profit: { $subtract: ["$booking_price", "$transaction_fee"] }
  }
 },
 {
  $group: {
   _id: "$parking_lot.area_id",
   profit: { $sum: "$net_profit"}
  }
 },
 {
  $sort: { profit: -1 }
 },
 {
  $limit : 5
 },
 {
  $lookup: {
   from: "areas",
   localField: "_id",
   foreignField: "_id",
   as: "area_docs"
  }
 }
]).map(function (el) {
 return { area_name: el.area_docs[0].name, profit: el.profit };
});
```

INTERESTING INQUIRIES:
AVERAGE PRICES PER AREA



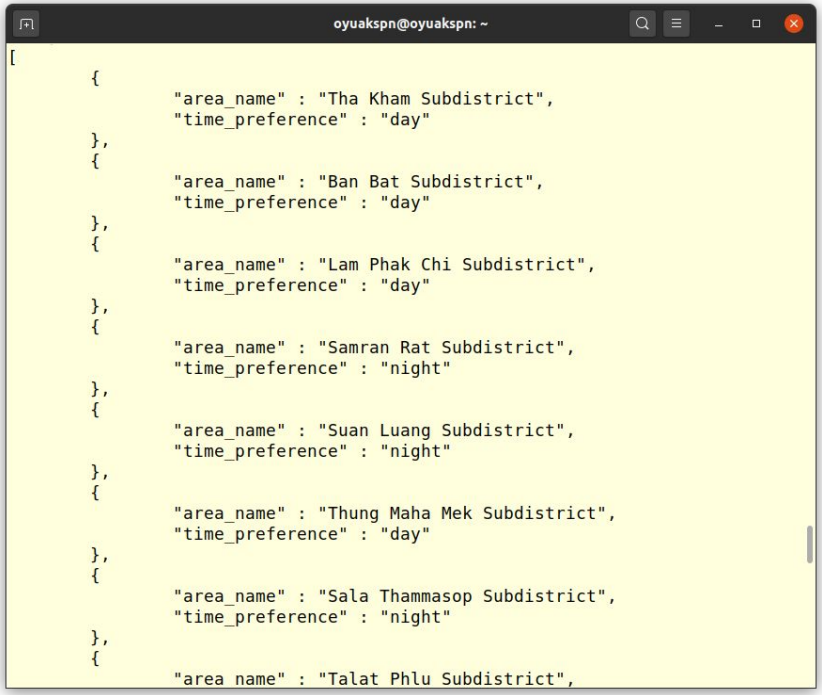

```
db.parking_lots.aggregate([
 {
  $group: {
   _id: "$area._id",
   avg_short_term_price: { $avg: "$short_term_price_per_hour"},
   avg_long_term_price: { $avg: "$long_term_price_per_day"},
  }
 },
 {
  $lookup: {
   from: "areas",
   localField: "_id",
   foreignField: "_id",
   as: "area_docs"
  }
 }
]).map(function (el) {
  return { area_name: el.area_docs[0].name, avg_short_term_price:
el.avg_short_term_price, avg_long_term_price: el.avg_long_term_price };
});
```

INTERESTING INQUIRIES:
AVERAGE PARKING TIME PER AREA



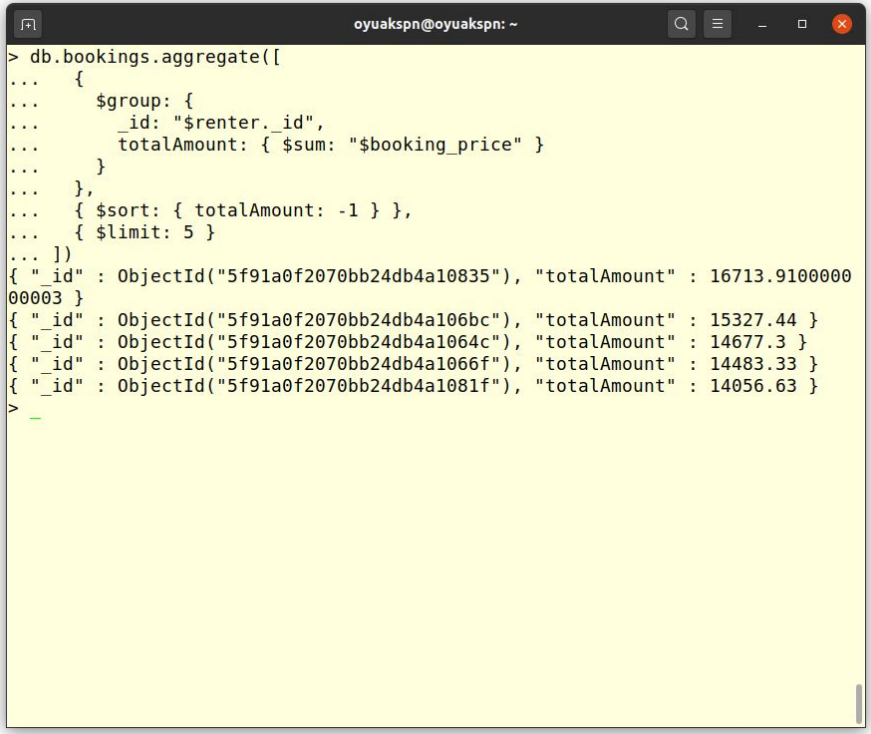

```
db.bookings.aggregate([
 { $match: { status: "completed" } },
 {
  $project: {
   parking_lot: 1,
   parking_time: {
    $divide: [{ $subtract: ["$departure_datetime", "$arrival_datetime"] },
     60 * 1000 * 60
    ]
   }
  }
 },
 {
  $group: {
   _id: "$parking_lot.area_id", avg_parking_time: { $avg: "$parking_time"},
  }
 },
 {
  $sort: { avg_parking_time: -1 }
 },
 {
  $lookup: {
   from: "areas", localField: "_id", foreignField: "_id", as: "area_docs"
  }
 }
]).map(function (el) {
 return { area_name: el.area_docs[0].name, avg_parking_time: el.avg_parking_time };
});
```

INTERESTING INQUIRIES: PARKING TIME PREFERENCE PER AREA



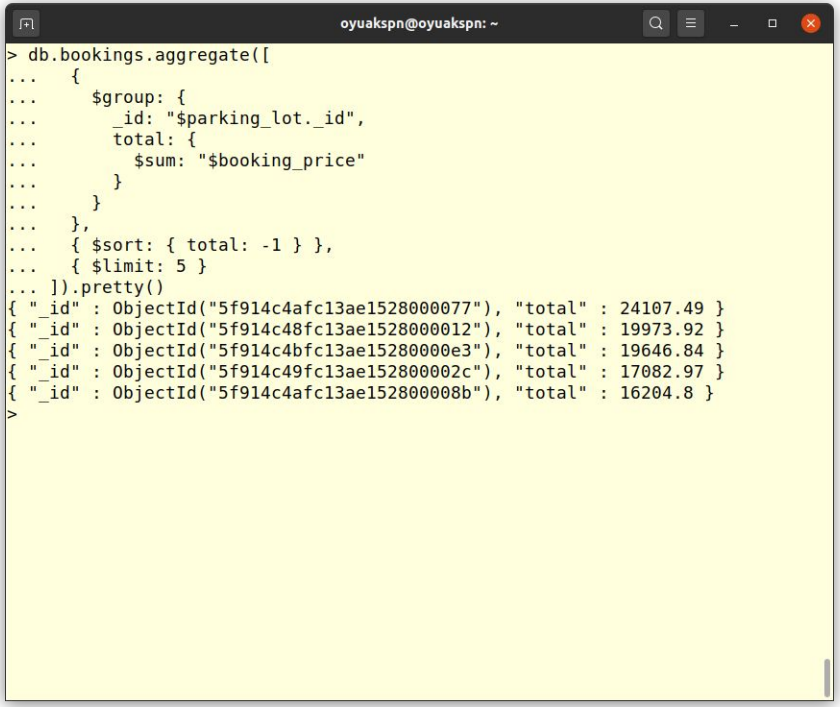

```
db.bookings.aggregate([
 { $match: { status: "completed" } },
 { $project: { parking_lot: 1, hr: { $hour: "$arrival_datetime" } } },
 { $project: { parking_lot: 1, time: { $cond: {
       if: { $and: [{ $gte: [ "$hr", 6 ] }, { $lt: [ "$hr", 18 ] }] },
       then: "day", else: "night" }
 } } },
 { $group: { _id: "$parking_lot.area_id",
    day: { $sum: { $cond: [{ $eq: ["$time", "day"]}, 1, 0] } },
    night: { $sum: { $cond: [{ $eq: ["$time", "night"]}, 1, 0] } }
 } },
 { $project: { _id: 1, time_pref: { $cond: {
       if: { $eq: ["$day", "$night"] }, then: "both",
       else: {
         $cond: { if: { $gt: ["$day", "$night"] }, then: "day", else: "night" }
 } } } } },
 {
  $lookup: {
    from: "areas", localField: "_id", foreignField: "_id", as: "area_docs"
  }
 }
]).map(function (el) {
 return { area_name: el.area_docs[0].name, time_preference: el.time_pref
};
});
```

INTERESTING INQUIRIES:
TOP 5 MOST SPENT RENTERS




```
db.bookings.aggregate([
  {
    $group: {
      _id: "$renter._id",
      totalAmount: { $sum: "$booking_price" }
    }
  },
  { $sort: { totalAmount: -1 } },
  { $limit: 5 }
])
```

INTERESTING INQUIRIES:
TOP 5 MOST PROFITABLE PARKING LOTS

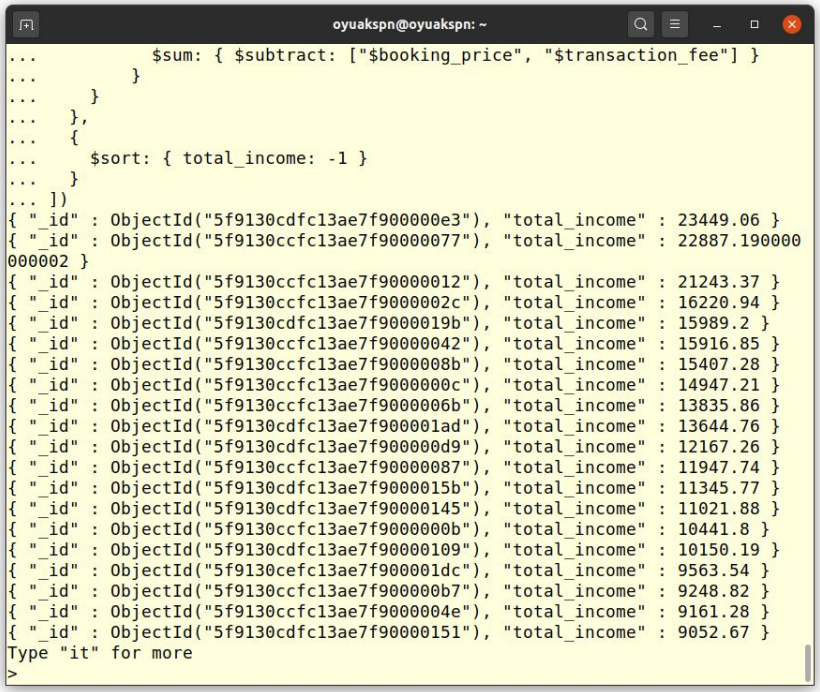



```
db.bookings.aggregate([
 {
  $group: {
   _id: "$parking_lot._id",
   total: {
    $sum: "$booking_price"
   }
  }
 },
 { $sort: { total: -1 } },
 { $limit: 5 }
])
```

INTERESTING INQUIRIES:
TOP 10 MOST POPULAR VEHICLE BRANDS


```
db.users.aggregate([
 {
  $unwind: {
            path: "$vehicles",
            includeArrayIndex: 'string',
            preserveNullAndEmptyArrays: false
  }
 },
 { $match: { "vehicles.status" : "active" } },
 {
  $group: {
   _id: "$vehicles.brand",
   "count": { $sum: 1 }
  }
 },
 { $sort: { "count": -1 } },
 { $limit: 10 }
])
```

INTERESTING INQUIRIES:
REPORT OF PROVIDER'S TOTAL INCOME

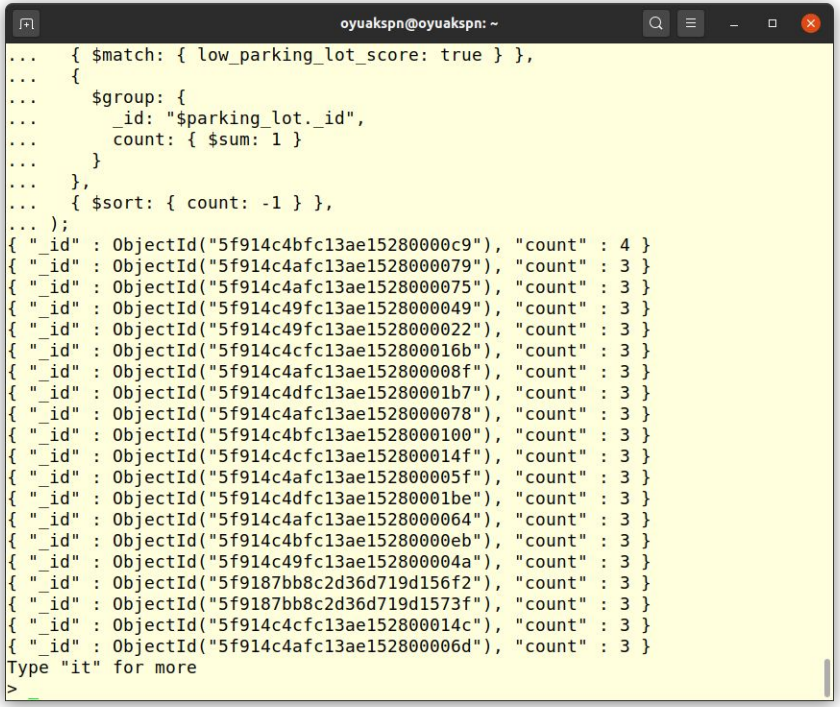



```
db.bookings.aggregate([
 { $match: { status: "completed" } },
 {
  $group: {
     _id: "$parking_lot.owner_id",
     total_income: {
       $sum: { $subtract: ["$booking_price", "$transaction_fee"] }
     }
  }
 },
 {
  $sort: { total_income: -1 }
 }
])
```

INTERESTING INQUIRIES:
THE BEST REVIEWED PARKING LOT


```
db.bookings.aggregate([
 {
  $match: { parking_lot_score: 5, service_score: 5, total_score: 5 }
 },
 { $limit: 5 },
 {
  $project: {
     parking_lot: 1,
     parking_lot_score: 1,
     service_score: 1,
     total_score: 1
  }
 }
])
```

INTERESTING INQUIRIES:
PARKING LOTS WITH LOW FACILITIES SCORE

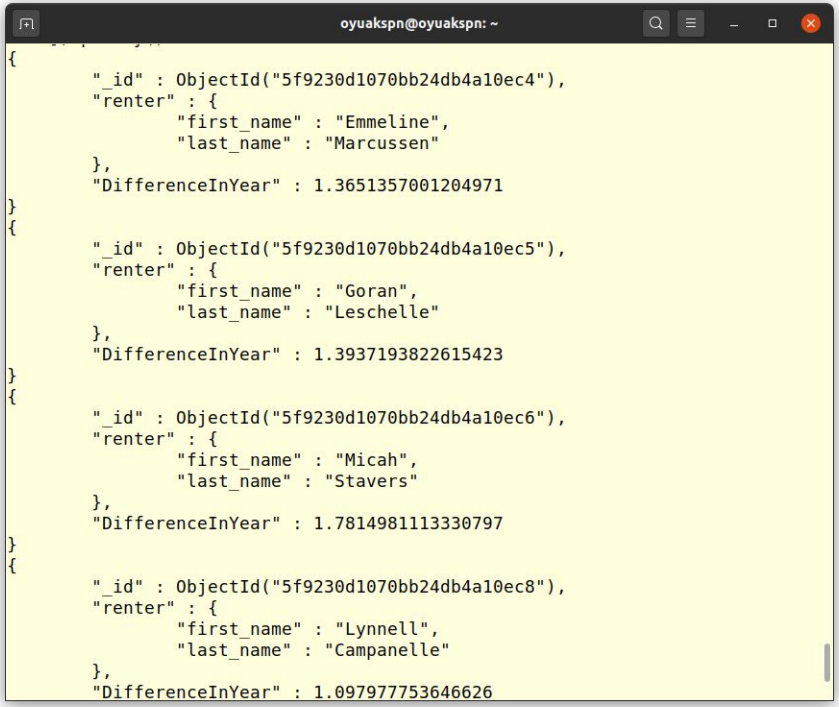



```
db.bookings.aggregate(
 { $match: { status: "completed" } },
 {
  $project: {
   parking_lot: 1,
   low_parking_lot_score: { $lt: [ "$parking_lot_score", 3 ] }
  }
 },
 { $match: { low_parking_lot_score: true } },
 {
  $group: {
   _id: "$parking_lot._id",
   count: { $sum: 1 }
  }
 },
 { $sort: { count: -1 } },
);
```

INTERESTING INQUIRIES:
REPORT OF LATE CHECKOUT RENTERS

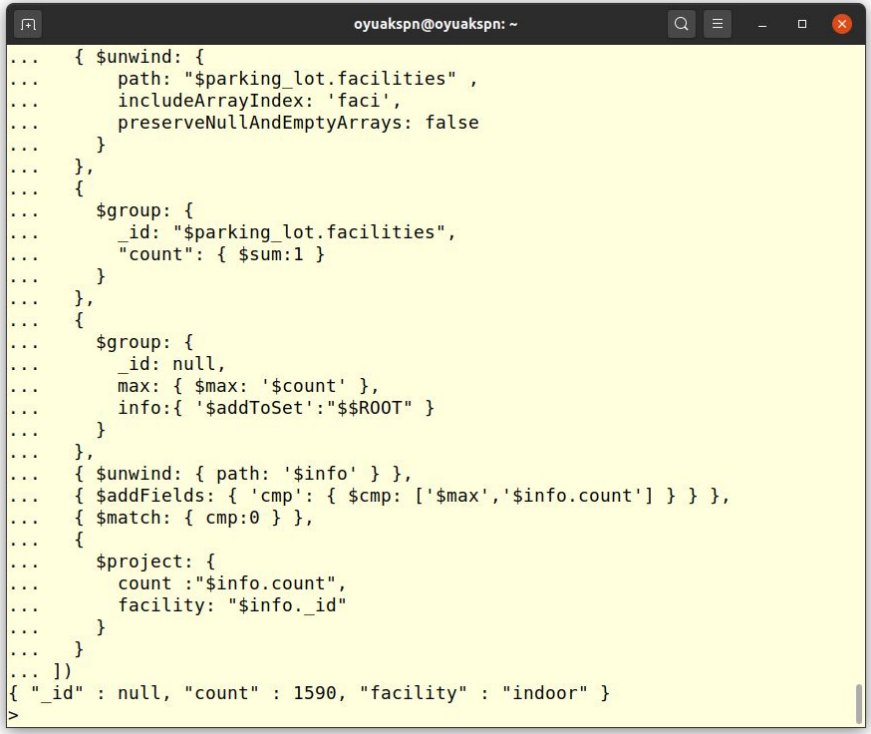

```
db.bookings.aggregate([
 {
  $project: {
   "renter.first_name": 1,
   "renter.last_name": 1,
    DifferenceInHours: {
     $divide: [
      {$subtract: ["$departure_datetime","$booking_end_datetime"] },
60000
      ]
    }
  }
 },
 { $match: { DifferenceInHours: { $gt: 30 } } }
])
```

INTERESTING INQUIRIES:
REPORT OF ONE-YEAR NONE ACTIVE USERS




```
db.bookings.aggregate([
 {
  $project: {
   "renter.first_name": 1,
   "renter.last_name": 1,
    DifferenceInYear: {
     $divide: [
      { $subtract: [ISODate(), "$booking_end_datetime"] },
31536000000]
     }
        }
  },
  { $match: { DifferenceInYear: { $gte:1 } } }
])
```

INTERESTING INQUIRIES:
MOST LIKELY POPULAR FACILITIES

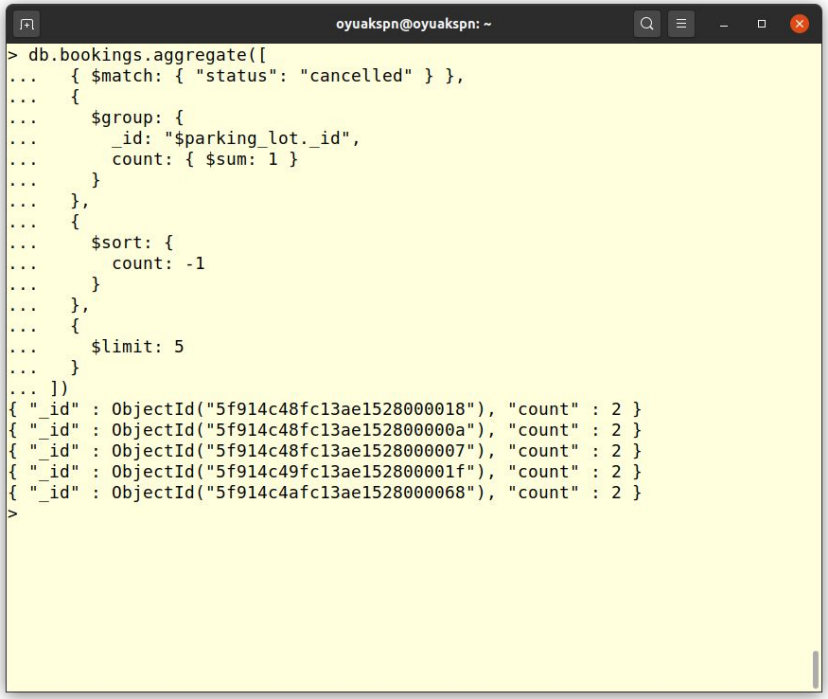



```
db.bookings.aggregate([
 { $unwind: {
    path: "$parking_lot.facilities" ,
    includeArrayIndex: 'faci',
    preserveNullAndEmptyArrays: false
  }
 },
 {
  $group: {
   _id: "$parking_lot.facilities",
   "count": { $sum:1 }
  }
 },
 {
  $group: {
   _id: null,
   max: { $max: '$count' },
   info:{ '$addToSet':"$$ROOT" }
  }
 },
 { $unwind: { path: '$info' } },
 { $addFields: { 'cmp': { $cmp: ['$max','$info.count'] } } },
 { $match: { cmp:0 } },
 {
  $project: {
   count :"$info.count",
   facility: "$info._id"
  }
 }
])
```

INTERESTING INQUIRIES:
NO. OF USERS USING DISCOUNTS


```
db.user_discounts.aggregate([
 {
  $project: {
   discount_id: 1,
   used: { $ne: [ "used_datetime", null ] }
  }
 },
 {
  $group: {
   _id: "$discount_id",
   count: { $sum: 1 }
  }
 }
]);
```

INTERESTING INQUIRIES:
TOP 5 MOST-CANCELLED PARKING LOT

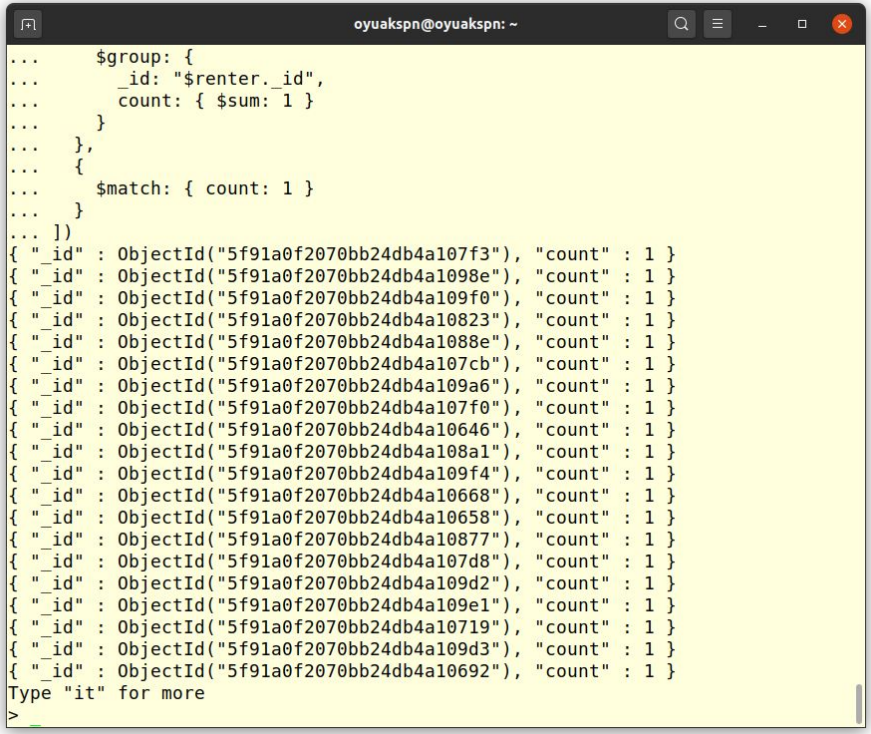



```
db.bookings.aggregate([
  { $match: { "status": "cancelled" } },
  {
    $group: {
      _id: "$parking_lot._id",
      count: { $sum: 1 }
    }
  },
  {
    $sort: {
      count: -1
    }
  },
  {
    $limit: 5
  }
])
```

INTERESTING INQUIRIES:
FREE-CANCELLATION PARKING LOT

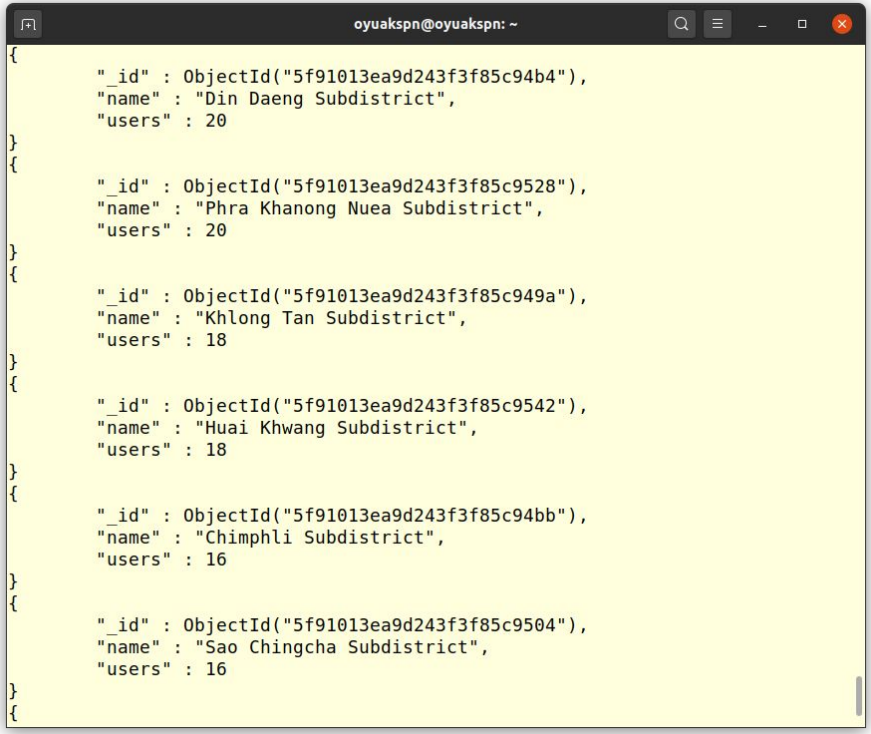
db.parking_lots.find({ "allow_free_cancelation": true })

INTERESTING INQUIRIES:
SINGLE-BOOKED RENTERS



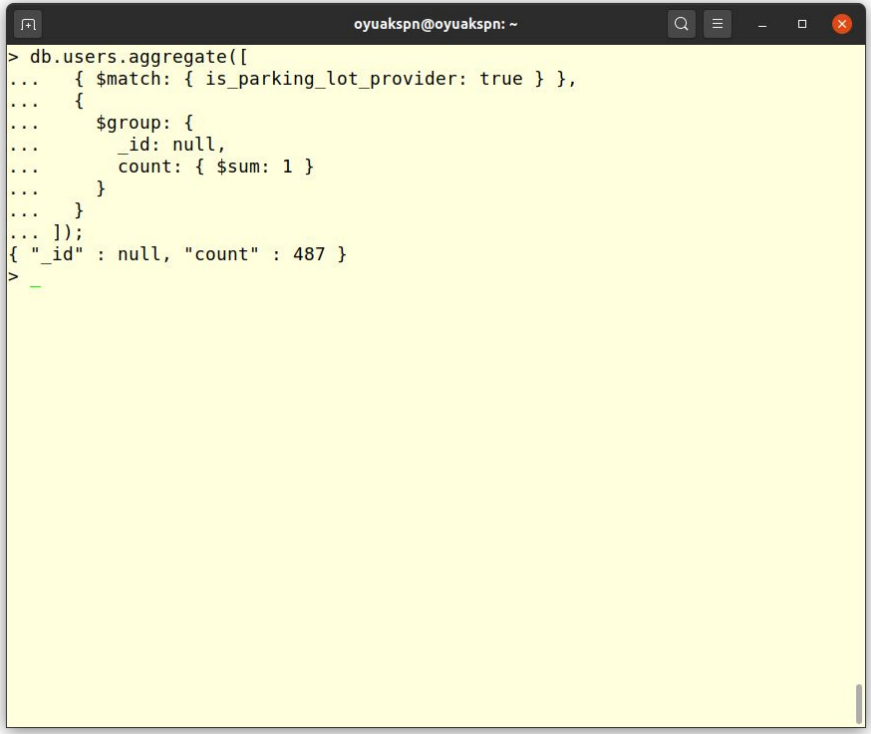

```
db.bookings.aggregate([
  { $match: { status: "completed" } },
  {
    $group: {
      _id: "$renter._id",
      count: { $sum: 1 }
    }
  },
  {
    $match: { count: 1 }
  }
]);
```

INTERESTING QUERIES:
NO. USERS PER AREA




```
db.users.aggregate([
 {
  $group: {
   _id: "$area_id",
   count: { $sum: 1 }
  }
 },
 { $sort: { count: -1 } },
 {
  $lookup: {
   from: "areas", localField: "_id", foreignField: "_id", as: "area_docs"
  }
 },
 {
  $unwind: {
            path: "$area_docs",
            includeArrayIndex: 'string',
            preserveNullAndEmptyArrays: false
  }
 },
 {
  $project: {
   name: "$area_docs.name",
   users: "$count"
  }
 }
])
```

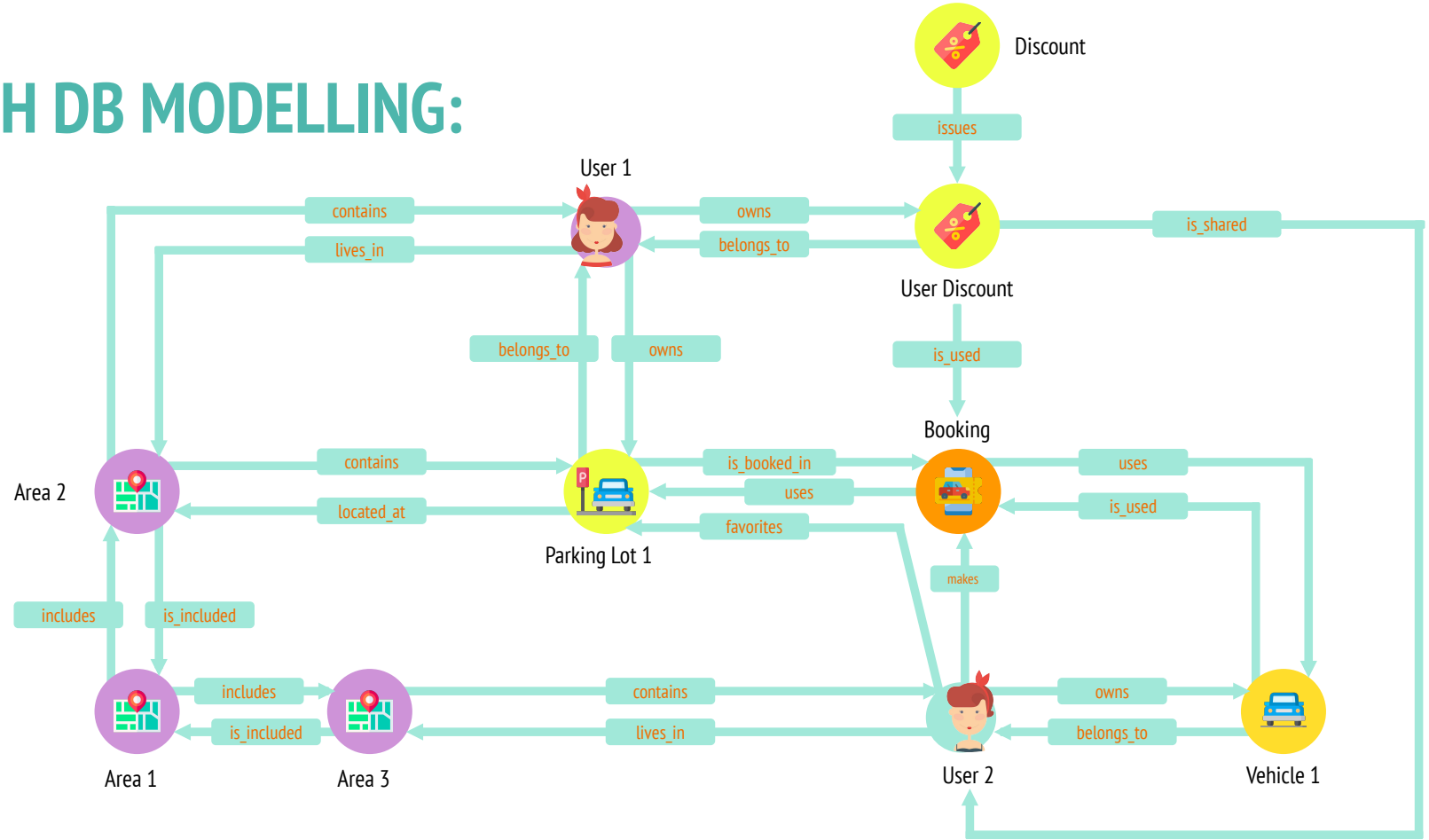
INTERESTING INQUIRIES:
NO. OF PROVIDERS




```
db.users.aggregate([
  { $match: { is_parking_lot_provider: true } },
  {
    $group: {
      _id: null,
      count: { $sum: 1 }
    }
  }
]);
```

GRAPH DB MODELLING

GRAPH DB MODELLING:

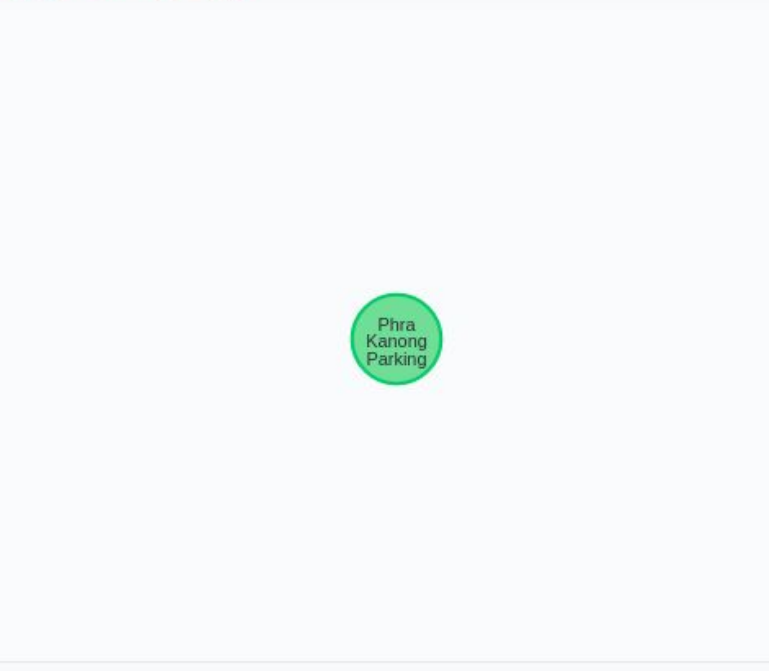


DATA OPERATION & INQUIRY

BASIC OPERATIONS:

- Create a parking lot
- Create a parking lot ownership
- Update a parking lot
- Search a parking lot
- Create a user
- Update a user
- Create a user's vehicles
- Update a user's vehicles
- Get a renter list
- Create a booking
- Give feedback to the parking lot
- Create an area with subareas
- Update an area
- Create a resident of an area
- Add a user's favorite parking lot
- Remove a user's favorite parking lot

BASIC OPERATION:
CREATE A PARKING LOT




```
CREATE ( pk:ParkingLot {name: "Phra Kanong Parking 1",
address: "344 Sukhumvit 71 Rd.",
landmark_description:"Aliquam erat volutpat In congue",
latitude: 13.567098,
longitude: 100.345678,
short_term_maximum_renting_hour: 8,
short_term_price_per_hour: 5,
allow_long_term_parking: true,
long_term_maximum_renting_day: 120,
long_term_price_per_day: 40,
allow_free_cancellation: false,
cancellation_fee: 20,
status: "available",
has_roof: true})
RETURN pk
```

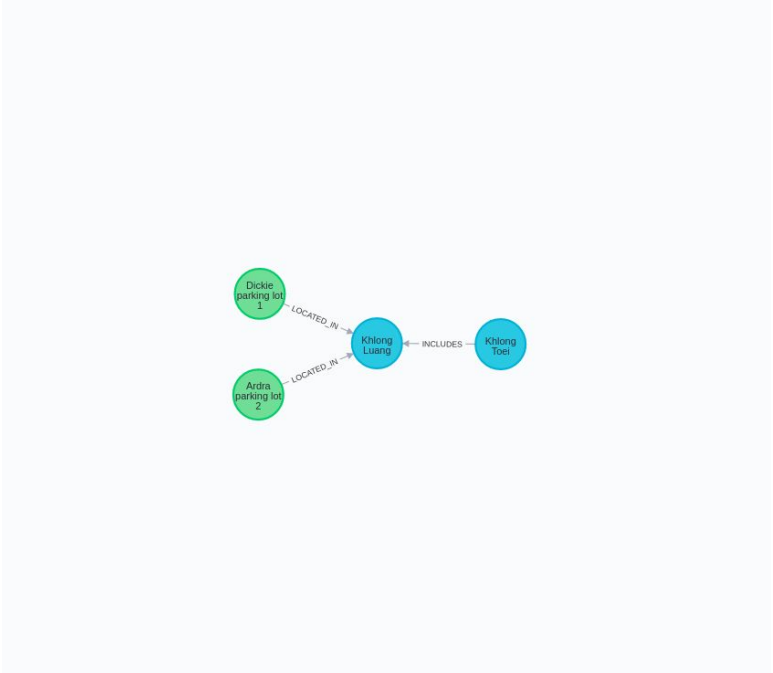
BASIC OPERATION:
CREATE A PARKING LOT OWNERSHIP


```
MATCH (u:User), (pk1:ParkingLot), (pk2:ParkingLot)
WHERE u.email = "hpolley8@ucoz.com"
AND pk1.name = "Hagen parking lot 1"
AND pk2.name = "Hagen parking lot 2"
MERGE (u)-[:OWNS]->(pk1),
(u)-[:OWNS]->(pk2),
(pk1)-[:BELONGS_TO]->(u),
(pk2)-[:BELONGS_TO]->(u)
RETURN u, pk1, pk2
```

BASIC OPERATION:
UPDATE A PARKING LOT

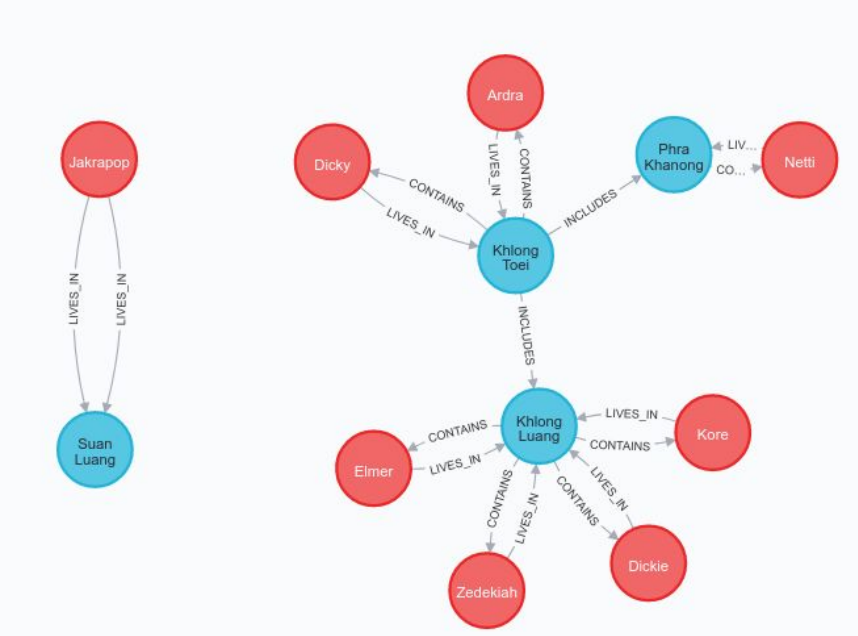

```
MATCH (pk:ParkingLot)
WHERE pk.name = "Phra Kanong Parking 1"
SET pk.name = "Pridi Alley 1", pk.address = "10 Sukhumvit 71"
RETURN pk
```

BASIC OPERATION:
SEARCH A PARKING LOT




```
MATCH (a:Area)-[:INCLUDES]-(b)<-[:LOCATED_IN]-(pk:ParkingLot)
WHERE pk.short_term_price_per_hour <= 30
AND a.name = "Khlong Toei"
RETURN a, b, pk
```

BASIC OPERATION:
CREATE A USER



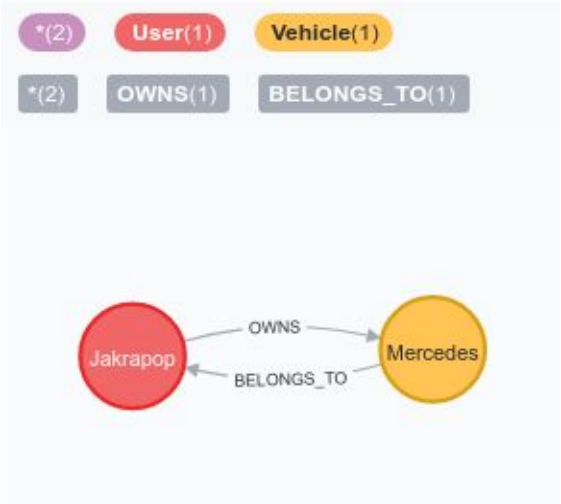

```
CREATE(user21:User {id: 21,
first_name: 'Jakrapop',
last_name: 'Akaranee',
email: 'st121075@ait.asia',
mobile_no: '0992415544',
citizen_id_no: '2812603604089',
driver_license_no: '2385656799479',
is_parking_lot_provider: false,
status: 'active' });

MATCH (ar:Area), (jk:User)
WHERE ar.name = "Suan Luang" AND jk.id = 21
CREATE (jk)-[r:LIVES_IN]->(ar) return jk, ar, r
```

BASIC OPERATION:
UPDATE A USER


```
{
  "identity": 509,
  "labels": [
    "User"
  ],
  "properties": {
"favorites": 64,
"citizen_id_no": "2812603604089",
"driver_license_no": "2385656799479",
"mobile_no": 980233465,
"last_name": "Fitzsimmons",
"id": 1,
"first_name": "Ardra",
"email": "adgoirtae@cscart.com",
"status": "Active",
"is_parking_lot_provider": true
  }
}
```


```
MATCH (u:User)
WHERE u.id = 1
SET u.mobile_no = 980233465, u.email = "adgoirtae@cscart.com"
RETURN u
```

BASIC OPERATION:
CREATE A USER'S VEHICLES



Step 1 Create a vehicle

```
MERGE (vehicle21:Vehicle {id: 21, vehicle_type: 'SUV',
brand: 'Mecedesk', model: 'E300',
color: 'Green', engine_serial_no: '5894158933316',
year: 2020, registration_plate: 'NN-1212',
registerd_place: 'Bangkok', status: 'Active' });
```

Step 2 Create a relationship of a vehicle and a user

```
MATCH (u21:User), (v21:Vehicle)
WHERE u21.id = 21 AND v21.id = 21
MERGE (u21)-[r:OWNS]->(v21 )
RETURN r ;
```

```
MATCH (u21:User), (v21:Vehicle)
WHERE u21.id = 21 AND v21.id = 21
MERGE (v21 )-[r:BELONGS_TO]->(u21)
RETURN r ;
```

Result

```
MATCH (u1:User)-[r:OWNS]->(cars)
WHERE u1.id =21
RETURN u1, cars
```

BASIC OPERATION:
UPDATE A USER'S VEHICLE

cars


```
  "identity": 621,
  "labels": [
    "Vehicle"
  ],
  "properties": {
"engine_serial_no": "5894158933316",
"color": "White",
"registerd_place": "Nan",
"year": 2020,
"vehicle_type": "SUV",
"model": "E300",
"id": 21,
"brand": "Mecedesk",
"status": "Active",
"registration_plate": "NN-1223"
  }
```

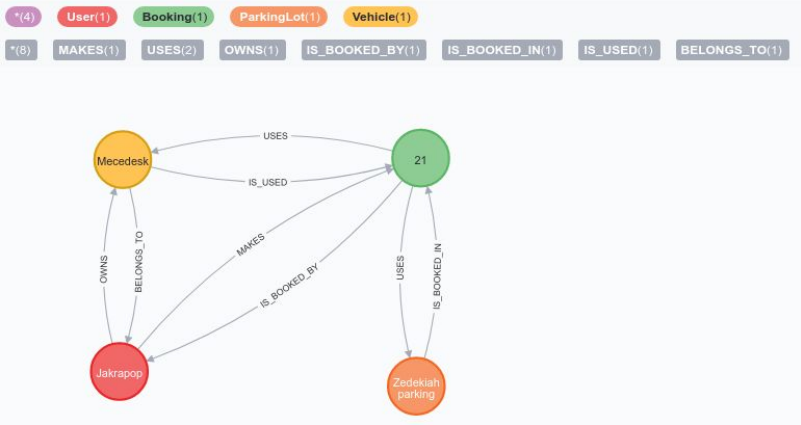

```
MATCH (jk:User)-[:OWNS]->(cars)
WHERE jk.first_name="Jakrapop" and cars.id=21
SET cars.color="White", cars.registration_plate="NN-1223",
cars.registerd_place="Nan"
RETURN jk,cars
```

BASIC OPERATION:
GET A RENTER LIST

"ID"	"FirstName"	"LastName"	"CitizenId"	"MobileNo"	"Email"
2	"Dicky"	"Mallam"	"3114909171600"	980233464	"adgoirtae@cs.com"
3	"Kyle"	"Colmer"	"4454682969732"	"970250155"	"kcolmer2@sciencedirect.com"
5	"Elmer"	"Moen"	"4218527625227"	"987612276"	"emoen4@yellowbook.com"
6	"Kathye"	"Partridge"	"8436934144386"	"988008842"	"kpartridge5@g.co"
8	"Neel"	"Lukasen"	"9040136210735"	"942095877"	"nlukasen7@skyrock.com"
11	"Kore"	"Noble"	"1063954565317"	"991288411"	"knoblek@hubpages.com"
12	"Kandace"	"Kleinfeld"	"8084385237301"	"991569014"	"kkleinfeldl@youtube.com"
16	"Dari"	"Tomasian"	"6781831668388"	"981667587"	"dtomasianp@istockphoto.com"
18	"Frankie"	"Chattey"	"3230939371716"	"924173386"	"fchatteyr@dailymail.co.uk"
19	"Legra"	"Malkin"	"9921751512367"	"961468712"	"lmalkins@cmu.edu"
20	"Bryana"	"Afield"	"3964736713610"	"994863363"	"bafieldt@desdev.cn"

```
MATCH (u:User)
WHERE u.is_parking_lot_provider = false
RETURN u.id AS ID,
u.first_name AS FirstName,
u.last_name AS LastName,
u.citizen_id_no AS CitizenId,
u.mobile_no AS MobileNo,
u.email AS Email
```

BASIC OPERATION:
CREATE A BOOKING



Step 1 Create a new Booking

```
CREATE (b:Booking {booking_start_datetime: datetime('2020-10-30T08:00:01'),
booking_end_datetime: datetime('2020-10-30T17:00:01'),
arrived_datetime: datetime('0000-01-01T00:00:00'),
departed_datetime: datetime('0000-01-01T00:00:00'),
booking_price: 0.0, discount_price: 0.0, transaction_fee: 0.0, payment_method: 'cash',
payment_datetime: datetime('0000-01-01T00:00:00'), parking_lot_score: 0.0, service_score: 0.0,
total_score: 0.0, feedback: 'Curabitur convallis.', status: 'booked'})
```

Step 2 Create booking and user relationships

```
MATCH (u:User), (b:Booking)  WHERE u.id = 21 AND u.id = 21
MERGE (u)-[r1:MAKES]->(b), (b)-[r2:IS_BOOKED_BY]->(u)
RETURN r1, r2 ;
```

Step 3 Create booking and parking lot relationships

```
MATCH (b:Booking), (p:ParkingLot)  WHERE b.id = 21 AND p.id = 14
MERGE (b)-[r1:USES]->(p), (p)-[r2:IS_BOOKED_IN]->(b)
RETURN r1, r2 ;
```

Step 4 Create a booking and vehicle relationships

```
MATCH (b:Booking), (v:Vehicle)  WHERE b.id=21 AND v.id=21
MERGE (b)-[r1:USES]->(v), (v)-[r2:IS_USED]->(b)
RETURN r1, r2 ;
```

Result

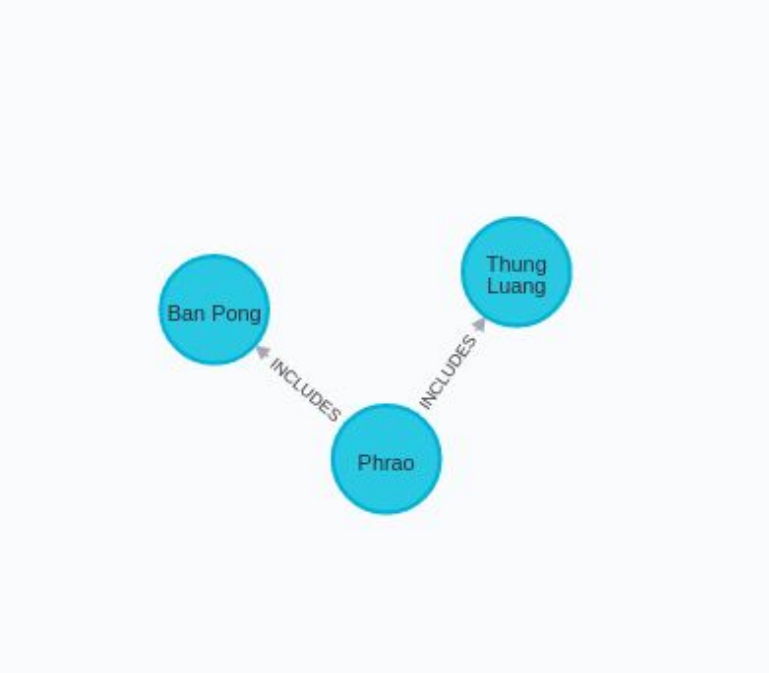
```
MATCH (jk:User)-[r:MAKES]->(bk:Booking),(bk)-[r2:USES]->(pl:ParkingLot),
(bk)-[r3:USES]->(car:Vehicle)  WHERE jk.id = 21
RETURN jk, r, bk, r2, pl, r3, car
```

BASIC OPERATION: GIVE FEEDBACK TO A PARKING LOT

"Customer_ID"	"ParkingLot_Name"	"Feedback"
11	"Zedekiah parking lot 2"	"Very good and clean"

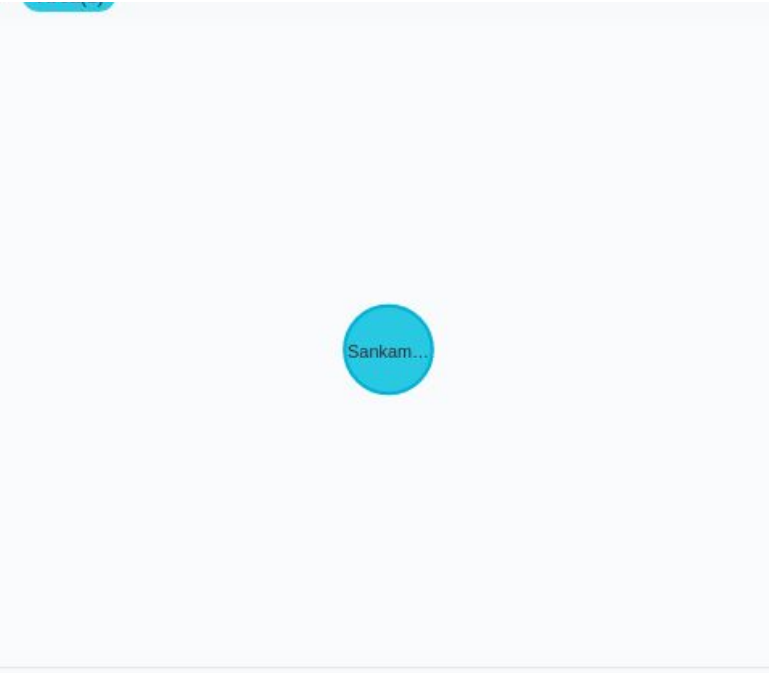
```
MATCH (n:User)-[:MAKES]-(b:Booking)-[:USES]-(p:ParkingLot) WHERE b.id=1 SET
b.feedback="Very Good and Clean"
RETURN n.id AS Customer_ID, p.name AS ParkingLot_Name,b.feedback AS Feedback
```

BASIC OPERATION:
CREATE AN AREA WITH SUBAREAS



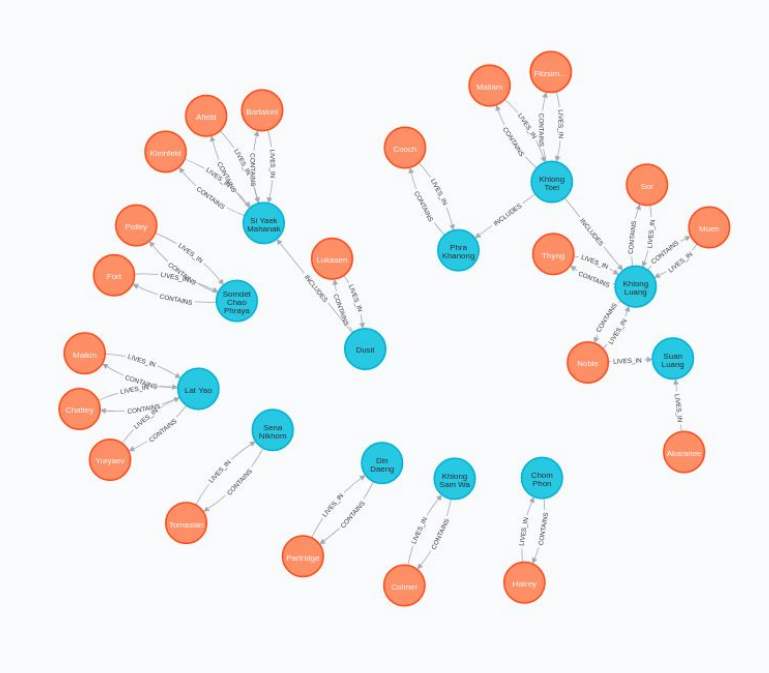

```
CREATE (pr:Area {name: "Phrao"}),
(tl:Area {name: "Thung Luang"}),
(bp:Area {name: "Ban Pong"}),
(pr)-[:INCLUDES]->(tl),
(pr)-[:INCLUDES]->(bp)
RETURN pr, tl, bp
```

BASIC OPERATION:
UPDATE AN AREA



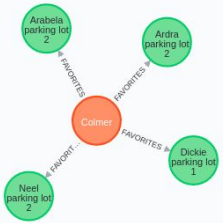

```
MATCH (a:Area)
WHERE a.name = "Sankamphaeng"
SET a.name = "Sankampang"
RETURN a
```

BASIC OPERATION:
GET ALL AREAS WITH RESIDENTS



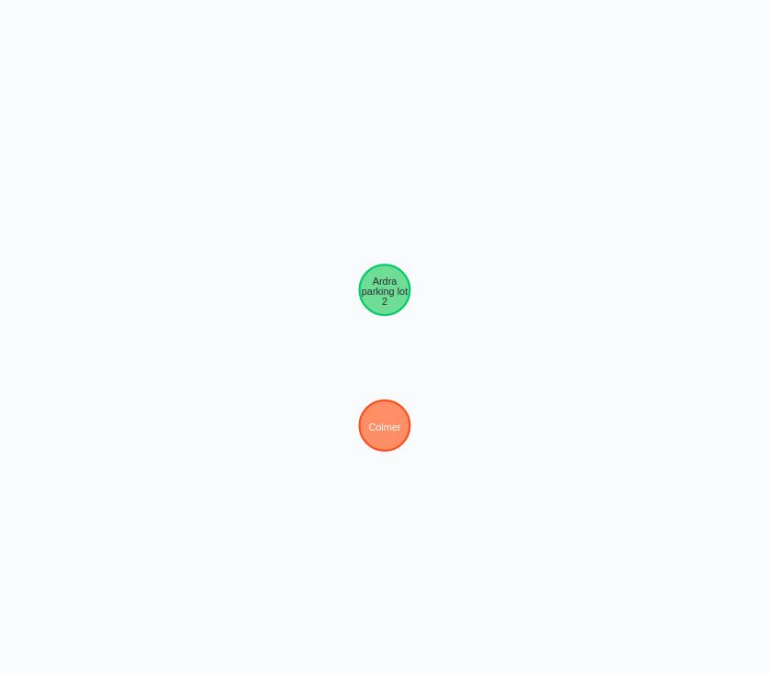

```
MATCH (a:Area)<-[:LIVES_IN]-(residents)
RETURN a, residents
```

BASIC OPERATION:
ADD A USER'S FAVORITE PARKING LOT




```
MATCH (u:User), (pk:ParkingLot)
WHERE u.email = "kcolmer2@sciencedirect.com"
AND pk.short_term_price_per_hour <=20
MERGE (u)-[:FAVORITES]->(pk)
RETURN u, pk
```

BASIC OPERATION:
REMOVE A USER'S FAVORITE PARKING LOT

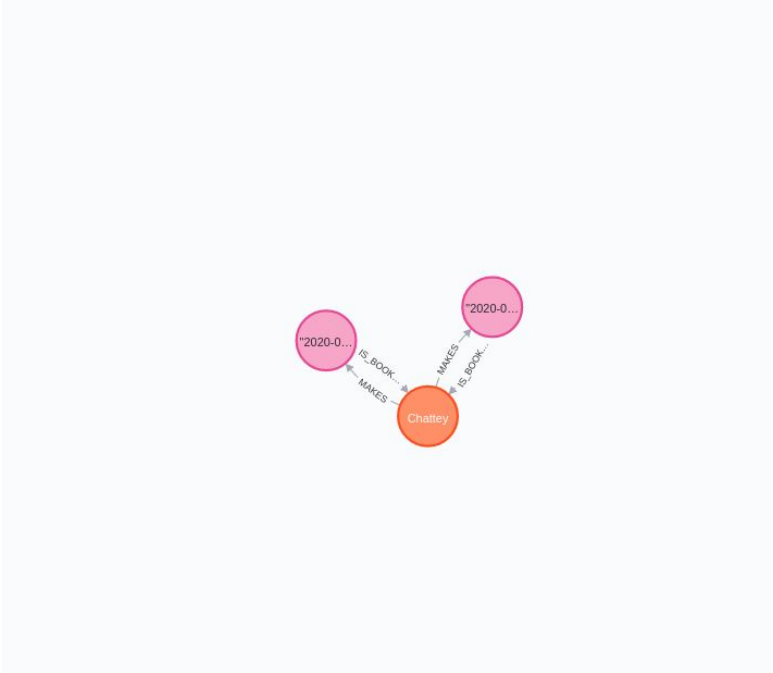



```
MATCH (u:User)-[r:FAVORITES]->(pk:ParkingLot)
WHERE u.email = "kcolmer2@sciencedirect.com"
AND pk.short_term_price_per_hour > 15
DELETE r
RETURN u, pk
```

INTERESTING INQUIRIES:

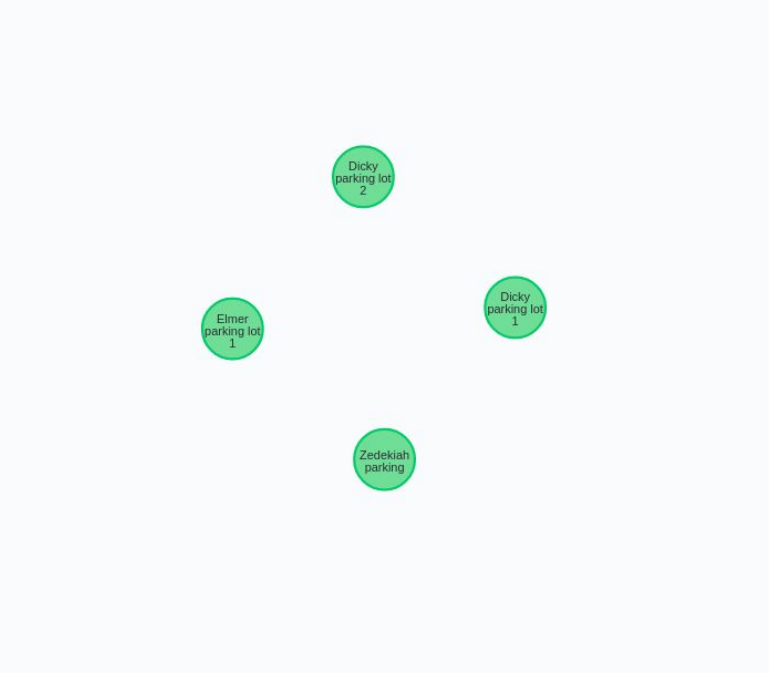
- View a user's booking history
- Top 5 most expensive parking lots
- No. of users per area
- Top 5 best review parking lots
- Top 5 most spent renters
- Top 5 most profitable parking lots
- Top 5 most popular vehicle types
- Provider's total income
- Average short-term parking of all parking lots
- Report of current year profit
- Report of customers of providers
- Report of number of free-cancellation parking lots
- Report of single-booked renters
- Report of number of providers
- Report of number of cancelled parking lot
- Report of average parking time per area
- Recommend parking lots in an area by reviewed score
- Top 5 most profitable areas
- Report of average short-term and long-term price per area
- Recommend areas with long-term parking lot

INTERESTING INQUIRIES:
VIEW A USER'S BOOKING HISTORY




```
MATCH (u:User)-[:MAKES]-(b:Booking)
WHERE u.email = "fchatteyr@dailymail.co.uk"
RETURN u, b
```

INTERESTING INQUIRIES:
TOP 5 MOST EXPENSIVE PARKING LOTS




```
MATCH (n:Booking)-[:USES]-(p:ParkingLot)
RETURN p
ORDER BY n.booking_price DESC
LIMIT 5
```

INTERESTING INQUIRIES:

Number of user per area

"AreaName"	"No_of_Customer"
"Khlong Luang"	4
"Si Yaek Mahanak"	3
"Lat Yao"	3
"Khlong Toei"	2
"Somdet Chao Phraya"	2
"Suan Luang"	2
"Khlong Sam Wa"	1
"Din Daeng"	1
"Dusit"	1
"Phra Khanong"	1
"Chom Phon"	1

```
MATCH (n:User)-[:LIVES_IN]-(a:Area)
RETURN a.name AS AreaName, COUNT(n) AS No_of_Customer
ORDER BY No_of_Customer DESC
```

INTERESTING INQUIRIES:
TOP 5 BEST REVIEWED PARKING LOTS

"Name"	"Address"	"Score"
"Zedekiah parking lot 1"	"92845 Schmedeman Park"	10
"Elmer parking lot 2"	"575 Rigney Court"	9.5
"Kathye parking lot 1"	"6 Holy Cross Street"	9.5
"Zedekiah parking lot 1"	"92845 Schmedeman Park"	9
"Kathye parking lot 1"	"6 Holy Cross Street"	9

```
MATCH (b:Booking)-[:USES]-(p:ParkingLot)
RETURN p.name AS Name,
p.address AS Address,
b.total_score AS Score
ORDER BY b.total_score DESC LIMIT 5
```

INTERESTING INQUIRIES:
TOP 5 MOST SPENT RENTERS

"FirstName"	"LastName"	"TotalSpent"
"Kandace"	"Kleinfeld"	168.5
"Netti"	"Cooch"	125.0
"Dari"	"Tomasian"	104.33333333333333
"Curcio"	"Hatrey"	91.3333333333333
"Zorah"	"Yuryaev"	86.5

```
MATCH (b:Booking)-[:IS_BOOKED_BY]->(u:User)
RETURN u.first_name AS FirstName,u.last_name AS LastName,SUM(b.booking_price) AS
TotalSpent
ORDER BY TotalSpent DESC
LIMIT 5
```

INTERESTING INQUIRIES:
TOP 5 MOST PROFITABLE PARKING LOTS

"ParkingLot_Name"	"profit"
"Zedekiah parking lot 1"	85.475
"Dicky parking lot 1"	81.575
"Zedekiah parking lot 1"	70.175
"Dicky parking lot 2"	66.75
"Elmer parking lot 1"	58.1833333333333

```
MATCH (n:Booking)-[:USES]-(p:ParkingLot)
RETURN p.name AS ParkingLot_Name,
n.booking_price - n.transaction_fee AS profit
ORDER BY profit DESC
LIMIT 5
```

INTERESTING INQUIRIES:
TOP 5 MOST POPULAR VEHICLE TYPES

"Vehicle_type"	"Total"
"Seadan"	6
"SUV"	5
"Hatchback"	3
"Pick-up"	3
"Van"	2

```
MATCH (n:Vehicle)
WITH COLLECT (n.vehicle_type) AS tt
UNWIND tt AS xx
RETURN DISTINCT(xx) AS Vehicle_type, COUNT(xx) AS Total
ORDER BY Total DESC
LIMIT 5
```

INTERESTING INQUIRIES:
PROVIDER'S TOTAL INCOME

"Customere_First_name"	"Last_name"	"Total_income"
"Zedekiah"	"Sor"	232.33333333333331
"Dicky"	"Mallam"	230.83333333333331
"Elmer"	"Moen"	108.3333333333333
"Kathye"	"Partridge"	69.33333333333333
"Kyle"	"Colmer"	62.6666666666667
"Dickie"	"Thyng"	55
"Ardra"	"Fitzsimmons"	43.5
"Hagen"	"Polley"	32.3333333333333
"Neel"	"Lukasen"	20
"Arabela"	"Fort"	15.6666666666667

```
MATCH (u:User)-[r1:OWNS]->(p:ParkingLot)-[r2:IS_BOOKED_IN]->(b:Booking)
WHERE b.status="Completed"
RETURN u.first_name AS Customere_First_name, u.last_name AS Last_name,
SUM(b.booking_price) AS Total_income
ORDER BY Total_income DESC
```

INTERESTING INQUIRIES:
AVERAGE SHORT-TERM PARKING OF ALL PARKING LOTS

"Average_duration_in_minutes"
217.99999999999997

```
MATCH (b:Booking)-[r:USES]->(p:ParkingLot)
WHERE  b.status="Completed" AND p.id=14
AND p.allow_long_term_parking=FALSE
RETURN avg(duration.inSeconds(b.arrived_datetime,
        b.departed_datetime).minutes) AS Average_duration_in_minutes
```

INTERESTING INQUIRIES:
REPORT OF CURRENT YEAR PROFIT

"Current_year_income"
87.97500000000002

```
WHERE b.booking_start_datetime.year = datetime().year
AND b.status="Completed"
RETURN SUM(b.transaction_fee) AS Current_year_income
```

INTERESTING INQUIRIES:
REPORT OF CUSTOMERS OF PROVIDERS

"My_customer_id"	"My_customer_name"	"My_customer_lastname"
19	"Legra"	"Malkin"

```
MATCH(u:User)-[r1:OWNS]->(p:ParkingLot)-[r2:IS_BOOKED_IN]->(b:Booking)-[r3:I
S_BOOKED_BY]->(cus:User)
WHERE u.id=9 AND  b.status="Completed"
RETURN distinct(cus.id) as My_customer_id ,
        cus.first_name as My_customer_name,
        cus.last_name as My_customer_lastname
```

INTERESTING INQUIRIES:
REPORT OF NO. OF FREE CANCELLATION PARKING LOTS

"count(pk)"
11

```
MATCH (pk:ParkingLot)
WHERE  pk.allow_free_cancellation =  true
RETURN   COUNT(pk)
```

INTERESTING INQUIRIES:
REPORT OF SINGLED BOOKED RENTERS

"use_id"	"User_status"	"Booking_status"
12	"Deleted"	"Booked"
17	"Deleted"	"Booked"
20	"Deleted"	"Booked"

```
MATCH (n:User)-[:MAKES]-(p:Booking)
WHERE n.status = "Deleted"
AND p.status = "Booked"
RETURN  n.id AS use_id,
n.status AS User_status,
p.status AS Booking_status
```

INTERESTING INQUIRIES:
REPORT OF NO. OF PROVIDERS

count(pk)
5

```
MATCH (pk:User)
WHERE pk.is_parking_lot_provider= true
RETURN COUNT(pk)
```

INTERESTING INQUIRIES:
REPORT OF NO. OF CANCELLED PARKING LOT

"use_id"	"User_status"	"Booking_status"
12	"Deleted"	"Booked"
17	"Deleted"	"Booked"
20	"Deleted"	"Booked"

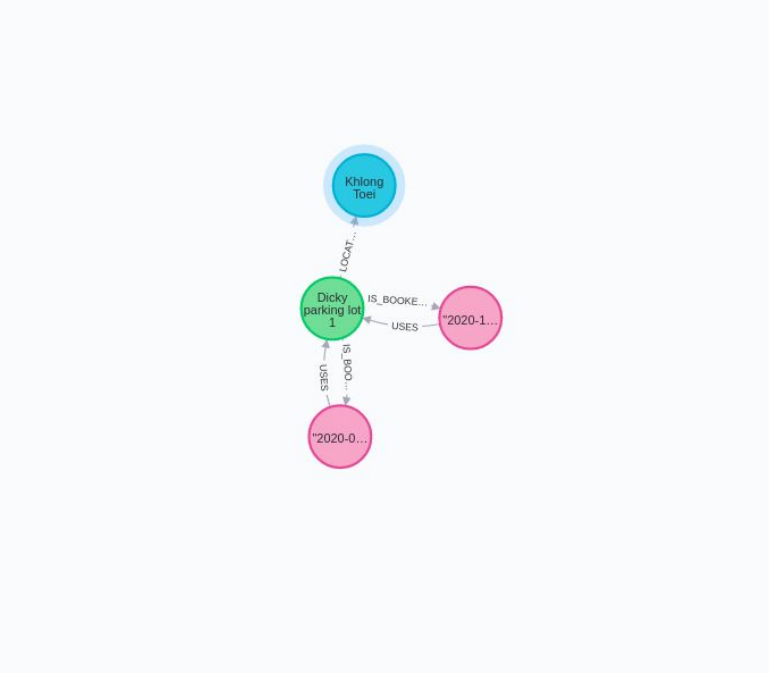
```
MATCH (n:User)-[:MAKES]-(p:Booking)
WHERE n.status = "Deleted" AND p.status = "Booked"
RETURN  n.id AS use_id,
n.status AS User_status,
p.status AS Booking_status
```

INTERESTING INQUIRIES:
REPORT OF AVERAGE PARKING TIME PER AREA

"Name"	"Duration"
"Dusit"	36.0
"Khlong Luang"	12.0
"Din Daeng"	8.0
"Chom Phon"	0.0
"Khlong Toei"	0.0
"Khan Na Yao"	0.0
"Si Yaek Mahanak"	0.0
"Phra Khanong"	0.0

```
MATCH
(a:Area)<-[:LOCATED_IN]-(pk:ParkingLot)-[:IS_BOOKED_IN]->(b:Booking)
RETURN  a.name AS Name,
AVG(duration.inSeconds(date(b.arrived_datetime),
date(b.departed_datetime)).hours) AS Duration
ORDER  BY  Duration DESC
```

INTERESTING INQUIRIES:
RECOMMEND PARKING LOTS IN AN AREA BY REVIEWED SCORE




```
MATCH
(a:Area)<-[:LOCATED_IN]-(pk:ParkingLot)-[:IS_BOOKED_IN]->(b:Booking)
WHERE b.total_score >= 4
AND a.name = "Khlong Toei"
RETURN a, pk, b
```

INTERESTING INQUIRIES:
TOP 5 MOST PROFITABLE AREAS

"Area"	"Profit"
"Khan Na Yao"	257.04166666666663
"Khlong Toei"	137.4083333333333
"Chom Phon"	117.25
"Khlong Luang"	87.025
"Din Daeng"	79.22500000000002

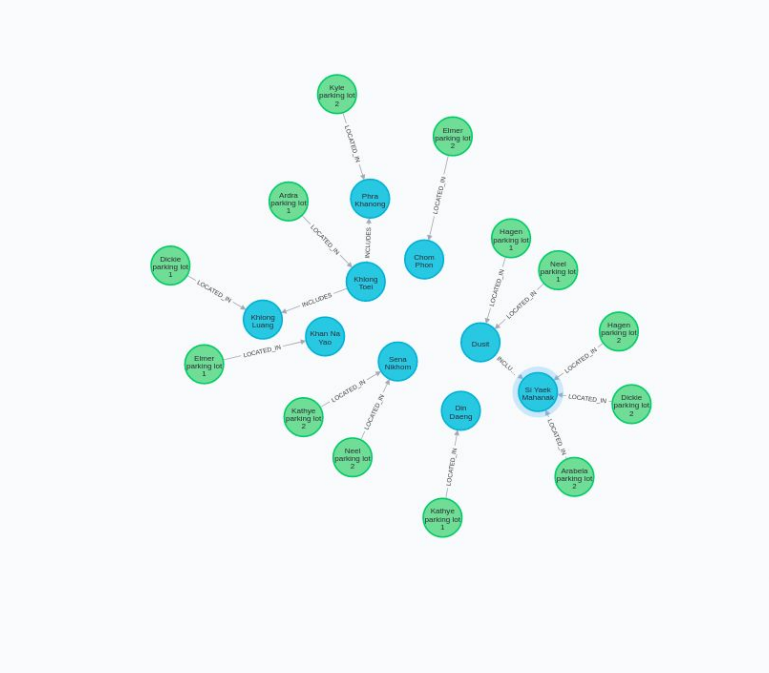
```
MATCH
(a:Area)<-[:LOCATED_IN]-(pk:ParkingLot)-[:IS_BOOKED_IN]->(b:Booking)
RETURN  a.name AS Area,
SUM(b.booking_price) - SUM(b.transaction_fee) AS Profit
ORDER  BY  Profit DESC
LIMIT  5
```

INTERESTING INQUIRIES:
REPORT OF AVERAGE PRICES PER AREA

"Area"	"ShortTerm"	"LongTerm"
"Khlong Luang"	16.5	530.5
"Sena Nikhom"	17.5	697.0
"Dusit"	28.5	565.5
"Si Yaek Mahanak"	30.75	476.0
"Din Daeng"	31.0	785.0
"Khan Na Yao"	31.5	636.5
"Phra Khanong"	37.0	770.0
"Khlong Toei"	40.5	337.5
"Chom Phon"	40.5	440.5
"Khlong Sam Wa"	42.0	333.0

```
MATCH (a:Area)<-[:LOCATED_IN]-(pk:ParkingLot)
RETURN a.name AS Area,
AVG(pk.short_term_price_per_hour) AS ShortTerm,
AVG(pk.long_term_price_per_day) AS LongTerm
ORDER BY ShortTerm
```

INTERESTING INQUIRIES:
RECOMMEND AREAS WITH LONG-TERM PARKING LOTS




```
MATCH (pk:ParkingLot)-[:LOCATED_IN]->(a:Area)
WHERE pk.allow_long_term_parking = true
RETURN pk, a
```

CONCLUSION

CONCLUSION: DOCUMENT DB MODEL

- Embedding can result in data duplication, especially with Vehicles data in Users collection. As unique indexes exist only across collection, to prevent the duplication of data, it needs the client side to check for it.
- As it is not enforced to create schema and rules on collections, there could be an incorrect data, data with incorrect type, or incorrect field name in the document. It is difficult to check these data on all documents in the database. To prevent this, MongoDB provides Schema Validation to create rules and restrictions on collections.
- It is less complex when performs query, compare to Relational DB Model. The essential data is in the same place therefore we don't need to struggle with JOIN to get data on some field of other collection. On the other hand, it's hard to determine which data need to be embedded or reference. One will know until they found the difficulty in query and the expensive reading operation.

CONCLUSION: GRAPH DB MODEL

- Graph DB Model query is fast and can be done, on some complex SQL and MongoDB query, with a single match.
- Graph DB Model visualize how the node related to each other which is easy to understand for everyone.
- A user's favorites entity can be represented as edge in Graph DB Model as it shows the relationship between users and parking lots, while in Relational and Document DB Model, it needs to be a table and a collection respectively.
- It's doesn't enforce to have a rigid schema as same as Document DB Model so it can be problematic with data management and consistency.
- It shouldn't be used to replace a whole Relational DB Model as some data model might needs more consistency and normalization.

Q & A