**IE6700 DATA MANAGEMENT FOR ANALYTICS**

**Project Title: “PARKeasy: Parking Management System”**

**Milestone: Logical (Relational) Model**

**Group 10**

**Student 1:** Neel Kamal

**Student 2:** Rahul Daruka

+18573997983

+16179369851

[lnu.nee@northeastern.edu](mailto:lnu.nee@northeastern.edu)

[daruka.r@northeastern.edu](http://daruka.r@northeastern.edu)

**Percentage of Effort Contributed by Student1:** 50%

**Percentage of Effort Contributed by Student2:** 50%

**Signature of Student 1:**   Neel Kamal

**Signature of Student 2:**  Rahul Daruka

**Submission Date:** 03 December 2023

Question 1 : Find the discount percentage applied to each member's membership.

db.Membership.aggregate([

{

$lookup: {

from: "Discount",

localField: "DiscountID",

foreignField: "DiscountID",

as: "discount\_info"

}

},

{

$match: { "discount\_info.Status": "Active" }

},

{

$project: {

\_id: 0,

MembershipID: 1,

DiscountID: 1,

Status: "$discount\_info.Status",

Percentage: "$discount\_info.Percentage"

}

}

])

Output :

A screen shot of a computer code

Description automatically generated

Question 2 : Identify the admins who have resolved the most number of inspections.

db.Incident.aggregate([  
       { $match: { ResolutionStatus: "Resolved" } },  
       { $lookup: {  
           from: "Inspect",  
           localField: "IncidentID",  
           foreignField: "IncidentID",  
           as: "inspection\_info"  
         }  
       },  
       { $unwind: "$inspection\_info" },  
       { $group: {  
           \_id: "$inspection\_info.AdminID",  
           resolvedCount: { $sum: 1 }  
         }  
       },  
       { $sort: { resolvedCount: -1 } }  
     ])

Output :

A blue background with orange and blue text

Description automatically generated

Question 3 : Retrieve details of members who own a specific type of vehicle 'SUV' and have an active membership.

db.Vehicle.aggregate([

{

$match: { VehicleType: "SUV" }

},

{

$lookup: {

from: "Member",

localField: "MemberID",

foreignField: "MemberID",

as: "member\_info"

}

},

{

$unwind: "$member\_info"

},

{

$lookup: {

from: "Membership",

localField: "member\_info.MembershipID",

foreignField: "MembershipID",

as: "membership\_info"

}

},

{

$match: { "membership\_info.Status": "Active" }

},

{

$project: {

MemberID: "$member\_info.MemberID",

PhoneNo: "$member\_info.PhoneNo",

Email: "$member\_info.Email",

Address: "$member\_info.Address",

Name: "$member\_info.Name",

MembershipID: "$member\_info.MembershipID",

Username: "$member\_info.Username"

}

}

])

Output :

A screen shot of a computer program

Description automatically generated

Question 4 : Retrieve all incidents, including the handling admin's details and the associated member's booking information.

db.Incident.aggregate([

{ $lookup: {

from: "Inspect",

localField: "IncidentID",

foreignField: "IncidentID",

as: "inspect\_info" } }, { $unwind: "$inspect\_info" },

{ $lookup: {

from: "Admin",

localField: "inspect\_info.AdminID",

foreignField: "AdminID",

as: "admin\_info" } },

{ $lookup: {

from: "Booking",

localField: "MemberID",

foreignField: "MemberID",

as: "booking\_info"

}

}

])

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

Question 5 : Summarize the total payments made by members for each type of membership and include their discount details.

db.Member.aggregate([

{ $lookup: {

from: "Payment",

localField: "MemberID",

foreignField: "MemberID",

as: "payment\_info" } }, { $unwind: "$payment\_info" },

{ $group: {

\_id: "$MembershipID",

totalPayments: { $sum: "$payment\_info.Amount" }

} }, { $lookup: {

from: "Membership",

localField: "\_id",

foreignField: "MembershipID",

as: "membership\_info"

}

},

{ $lookup: {

from: "Discount",

localField: "membership\_info.DiscountID",

foreignField: "DiscountID",

as: "discount\_info"

}

}

])

Output :

A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

Question 6 : Compute the average price of parking slots for each parking lot.

db.Includes.aggregate([  
  { $lookup: {  
      from: "ParkingSlot",  
      localField: "SlotID",  
      foreignField: "SlotID",  
      as: "slot\_info"    }  },  { $unwind: "$slot\_info" },  
  { $group: {  
\_id: "$LotID",  
averagePrice: { $avg: "$slot\_info.Price" }  
}  
}  
])

Output :

A screen shot of a computer code

Description automatically generated

Question 7 : Fetch Parking Slots with Specific Status.

db.parkingSlot.find({ Status: "Available" }) // Or "Occupied"

output :

A screen shot of a computer program

Description automatically generated