SQL

-- 4. Find (full) names of students who enroll in at least two different courses. Do not use aggregate and group by.

SELECT DISTINCT CONCAT(s.FirstName, ' ', s.LastName) AS FullName

FROM Students s

JOIN Enrollments e1 ON s.StudentID = e1.StudentID

JOIN Enrollments e2 ON e1.StudentID = e2.StudentID AND e1.CourseID != e2.CourseID;

-- 7. For each student by id, find out his/her total credit hours.

SELECT s.StudentID, CONCAT(s.FirstName, ' ', s.LastName) AS FullName, SUM(c.CreditHours) AS TotalCreditHours

FROM Students s

JOIN Enrollments e ON s.StudentID = e.StudentID

JOIN Courses c ON e.CourseID = c.CourseID

GROUP BY s.StudentID, FullName;

NoSQL

* projection

{

"collection": "orders",

"query": { "status": "shipped" },

"projection": { "userId": 1, "totalAmount": 1, "\_id": 0 }

}

* sort

{

"collection": "orders",

"query": { "status": "pending" },

"sort": { "totalAmount": -1 }

}

* limit

{

"collection": "orders",

"query": {},

"limit": 3,

"skip": 1

}

* aggregation

{

"collection": "orders",

"aggregation": [

{ "$match": { "status": "pending" } },

{ "$group": { "\_id": "$status", "totalSum": { "$sum": "$totalAmount" } } }

]

}

# (1) Find orders that are shipped to Denver and have a status “pending”. Return id of such orders only(excluding \_id too).

{

"collection": "orders",

"aggregation": [

{

"$match": {

"shippingAddress.city": "Denver",

"status": "pending"

}

},

{

"$project": {

"id": "$\_id",

"\_id": 0

}

}

]

}

# (3) Count products in stock for each category. Output category names and stock counts.

{

"collection": "products",

"aggregation": [

{

"$group": {

"\_id": "$category",

"totalstock": { "$sum": "$stock" }

}

},

{

"$lookup": {

"from": "categories",

"localField": "\_id",

"foreignField": "\_id",

"as": "category\_detail"

}

},

{

"$unwind": "$category\_detail"

},

{

"$project": {

"category": "$category\_detail.name",

"totalstock": "$totalstock",

"\_id": 0

}

}

]

}