**Explanation:**

1. **Inserting Data**: The orders and students collections are populated with sample data using insertMany.
2. **Total Price by Pizza Name**: This mapReduce operation calculates the total price for each pizza type (sum of all prices for each pizza name).
3. **Order Count**: It calculates the total number of orders by emitting a count for each order and summing them up.
4. **Students' Ages**: It stores the student's name along with their age and then finds the ages.
5. **Total Age of Students**: It sums the ages of all students.
6. **Graduation Status Count**: It counts the number of students who are graduated and not graduated.
7. **Maths Status Count**: It counts the number of students who have studied Maths and those who haven't.

// Switch to the 'test' database

use test;

// Insert sample data into the 'orders' collection

db.orders.insertMany([

{ \_id: 0, name: 'Pepperoni', size: 'small', price: 19, quantity: 10, date: ISODate("2021-03-13T08:14:30.000Z") },

{ \_id: 1, name: 'Pepperoni', size: 'medium', price: 20, quantity: 20, date: ISODate("2021-03-13T09:13:24.000Z") },

{ \_id: 2, name: 'Pepperoni', size: 'large', price: 21, quantity: 30, date: ISODate("2021-03-17T09:22:12.000Z") },

{ \_id: 3, name: 'Cheese', size: 'small', price: 12, quantity: 15, date: ISODate("2021-03-13T11:21:39.736Z") },

{ \_id: 4, name: 'Cheese', size: 'medium', price: 13, quantity: 50, date: ISODate("2022-01-12T21:23:13.331Z") },

{ \_id: 5, name: 'Cheese', size: 'large', price: 14, quantity: 10, date: ISODate("2022-01-12T05:08:13.000Z") },

{ \_id: 6, name: 'Vegan', size: 'small', price: 17, quantity: 10, date: ISODate("2021-01-13T05:08:13.000Z") },

{ \_id: 7, name: 'Vegan', size: 'medium', price: 18, quantity: 10, date: ISODate("2021-01-13T05:10:13.000Z") }

]);

// Insert sample data into the 'students' collection

db.students.insertMany([

{ name: 'Priya', age: 24, subject: ['History', 'Philosophy'], graduated: true, hasMaths: false },

{ name: 'Sita', age: 25, subject: ['Math', 'Science'], graduated: false, hasMaths: true },

{ name: 'Reema', age: 21, subject: ['Math', 'Science'], graduated: false, hasMaths: true },

{ name: 'Soham', age: 20, subject: ['SC', 'DSA'], graduated: false }

]);

// MapReduce: Calculate the total price for each pizza name

db.orders.mapReduce(

function() {

emit(this.name, this.price); // Emit pizza name as the key and price as the value

},

function(key, values) {

return Array.sum(values); // Sum up all the prices for each pizza

},

{ out: "totalPriceByPizza" }

);

// View the result of total price by pizza name

db.totalPriceByPizza.find();

// MapReduce: Count total orders (number of documents) in the 'orders' collection

db.orders.mapReduce(

function() {

emit("totalOrders", 1); // Emit a single key 'totalOrders' with value 1 for each document

},

function(key, values) {

return Array.sum(values); // Sum up all the counts for each order

},

{ out: "orderCount" }

);

// View the result of total orders count

db.orderCount.find();

// MapReduce: Count number of students by name

db.students.mapReduce(

function() {

emit(this.name, this.age); // Emit the student's name as the key and age as the value

},

function(key, values) {

return values[0]; // Return the age (or use a custom function if needed)

},

{ out: "studentsAge" }

);

// View the result of students' ages

db.studentsAge.find();

// MapReduce: Sum the 'age' for all students

db.students.mapReduce(

function() {

emit("totalAge", this.age); // Emit a key 'totalAge' and the student's age as the value

},

function(key, values) {

return Array.sum(values); // Sum the ages of all students

},

{ out: "totalStudentAge" }

);

// View the total age of students

db.totalStudentAge.find();

// MapReduce: Counting the number of students who graduated and those who did not

db.students.mapReduce(

function() {

if (this.graduated) {

emit("graduated", 1); // Emit 'graduated' with 1 if the student is graduated

} else {

emit("notGraduated", 1); // Emit 'notGraduated' with 1 if the student is not graduated

}

},

function(key, values) {

return Array.sum(values); // Sum the counts for each category

},

{ out: "graduationStatus" }

);

// View the result of graduation status counts

db.graduationStatus.find();

// MapReduce: Counting students who have Maths subject

db.students.mapReduce(

function() {

if (this.hasMaths) {

emit("hasMaths", 1); // Emit 'hasMaths' with 1 if the student has Maths

} else {

emit("noMaths", 1); // Emit 'noMaths' with 1 if the student does not have Maths

}

},

function(key, values) {

return Array.sum(values); // Sum the counts for each category

},

{ out: "mathsStatus" }

);

// View the result of students' maths status count

db.mathsStatus.find();