In database management an aggregate function is a function where the values of multiple rows are grouped together as input on certain criteria to form a single value of more significant meaning.

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\HP> mongosh

Current Mongosh Log ID: 653a0f89437a7b58b26142ee

Connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.0.2

Using MongoDB: 7.0.0

Using Mongosh: 2.0.2

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------

The server generated these startup warnings when booting

2023-10-16T01:34:24.222+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted

------

test> use Shubham

switched to db Shubham

Shubham> db.stud.find({})

[

{

\_id: ObjectId("65394c4179bc104fb33a3d59"),

rno: 16,

name: 'Shubham',

subject: 'DBMS',

marks: 45

},

{

\_id: ObjectId("65394c4179bc104fb33a3d5a"),

rno: 16,

name: 'Shubham',

subject: 'CNS',

marks: 40

},

{

\_id: ObjectId("65394c4179bc104fb33a3d5b"),

rno: 16,

name: 'Shubham',

subject: 'TOC',

marks: 30

},

{

\_id: ObjectId("65394c9c79bc104fb33a3d5c"),

rno: 13,

name: 'Ram',

subject: 'CNS',

marks: 45

},

{

\_id: ObjectId("65394c9c79bc104fb33a3d5d"),

rno: 16,

name: 'Ram',

subject: 'TOC',

marks: 40

},

{

\_id: ObjectId("65394c9c79bc104fb33a3d5e"),

rno: 13,

name: 'Ram',

subject: 'DSA',

marks: 45

},

{

\_id: ObjectId("65394cd979bc104fb33a3d5f"),

rno: 16,

name: 'Pratik',

subject: 'DBMS',

marks: 40

},

{

\_id: ObjectId("65394cd979bc104fb33a3d60"),

rno: 11,

name: 'Pratik',

subject: 'CNS',

marks: 50

},

{

\_id: ObjectId("65394cd979bc104fb33a3d61"),

rno: 11,

name: 'Pratik',

subject: 'TOC',

marks: 20

},

{

\_id: ObjectId("65394d1979bc104fb33a3d62"),

rno: 1,

name: 'Sid',

subject: 'DBMS',

marks: 30

},

{

\_id: ObjectId("65394d1979bc104fb33a3d63"),

rno: 11,

name: 'Sid',

subject: 'CNS',

marks: 30

},

{

\_id: ObjectId("65394d1979bc104fb33a3d64"),

rno: 1,

name: 'Sid',

subject: 'TOC',

marks: 50

}

]

Shubham> show dbs

Shubham 232.00 KiB

admin 40.00 KiB

blogApp 200.00 KiB

config 108.00 KiB

local 72.00 KiB

todo\_DB 72.00 KiB

Shubham> db.stud.find({rno:11}).explain("executionStata")

MongoServerError: Enumeration value 'executionStata' for field 'explain.verbosity' is not a valid value.

Shubham> db.stud.find({rno:11}).explain("executionStats")

{

explainVersion: '2',

queryPlanner: {

namespace: 'Shubham.stud',

indexFilterSet: false,

parsedQuery: { rno: { '$eq': 11 } },

queryHash: 'AD927D6E',

planCacheKey: 'FFD25DD2',

maxIndexedOrSolutionsReached: false,

maxIndexedAndSolutionsReached: false,

maxScansToExplodeReached: false,

winningPlan: {

queryPlan: {

stage: 'COLLSCAN',

planNodeId: 1,

filter: { rno: { '$eq': 11 } },

direction: 'forward'

},

slotBasedPlan: {

slots: '$$RESULT=s5 env: { s3 = 1698304864740 (NOW), s1 = TimeZoneDatabase(EST...Asia/Jayapura) (timeZoneDB), s7 = 11, s2 = Nothing (SEARCH\_META) }',

stages: '[1] filter {traverseF(s4, lambda(l1.0) { ((l1.0 == s7) ?: false) }, false)} \n' +

'[1] scan s5 s6 none none none none lowPriority [s4 = rno] @"be41ffec-f41a-481e-881f-beaf33748b79" true false '

}

},

rejectedPlans: []

},

executionStats: {

executionSuccess: true,

nReturned: 3,

executionTimeMillis: 10,

totalKeysExamined: 0,

totalDocsExamined: 12,

executionStages: {

stage: 'filter',

planNodeId: 1,

nReturned: 3,

executionTimeMillisEstimate: 0,

opens: 1,

closes: 1,

saveState: 0,

restoreState: 0,

isEOF: 1,

numTested: 12,

filter: 'traverseF(s4, lambda(l1.0) { ((l1.0 == s7) ?: false) }, false) ',

inputStage: {

stage: 'scan',

planNodeId: 1,

nReturned: 12,

executionTimeMillisEstimate: 0,

opens: 1,

closes: 1,

saveState: 0,

restoreState: 0,

isEOF: 1,

numReads: 12,

recordSlot: 5,

recordIdSlot: 6,

fields: [ 'rno' ],

outputSlots: [ Long("4") ]

}

}

},

command: { find: 'stud', filter: { rno: 11 }, '$db': 'Shubham' },

serverInfo: {

host: 'LAPTOP-ECO21OAA',

port: 27017,

version: '7.0.0',

gitVersion: '37d84072b5c5b9fd723db5fa133fb202ad2317f1'

},

serverParameters: {

internalQueryFacetBufferSizeBytes: 104857600,

internalQueryFacetMaxOutputDocSizeBytes: 104857600,

internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,

internalDocumentSourceGroupMaxMemoryBytes: 104857600,

internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,

internalQueryProhibitBlockingMergeOnMongoS: 0,

internalQueryMaxAddToSetBytes: 104857600,

internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,

internalQueryFrameworkControl: 'trySbeEngine'

},

ok: 1

}

Shubham> db.stud.find({rno:11})

[

{

\_id: ObjectId("65394cd979bc104fb33a3d60"),

rno: 11,

name: 'Pratik',

subject: 'CNS',

marks: 50

},

{

\_id: ObjectId("65394cd979bc104fb33a3d61"),

rno: 11,

name: 'Pratik',

subject: 'TOC',

marks: 20

},

{

\_id: ObjectId("65394d1979bc104fb33a3d63"),

rno: 11,

name: 'Sid',

subject: 'CNS',

marks: 30

}

]

Shubham> db.stud.createIndex({rno:1})

rno\_1

Shubham> db.stud.find({rno:11}).explain("executionStats")

{

explainVersion: '2',

queryPlanner: {

namespace: 'Shubham.stud',

indexFilterSet: false,

parsedQuery: { rno: { '$eq': 11 } },

queryHash: 'AD927D6E',

planCacheKey: 'C7300B94',

maxIndexedOrSolutionsReached: false,

maxIndexedAndSolutionsReached: false,

maxScansToExplodeReached: false,

winningPlan: {

queryPlan: {

stage: 'FETCH',

planNodeId: 2,

inputStage: {

stage: 'IXSCAN',

planNodeId: 1,

keyPattern: { rno: 1 },

indexName: 'rno\_1',

isMultiKey: false,

multiKeyPaths: { rno: [] },

isUnique: false,

isSparse: false,

isPartial: false,

indexVersion: 2,

direction: 'forward',

indexBounds: { rno: [ '[11, 11]' ] }

}

},

slotBasedPlan: {

slots: '$$RESULT=s11 env: { s5 = KS(2B160104), s2 = Nothing (SEARCH\_META), s10 = {"rno" : 1}, s1 = TimeZoneDatabase(EST...Asia/Jayapura) (timeZoneDB), s6 = KS(2B16FE04), s3 = 1698304981727 (NOW) }',

stages: '[2] nlj inner [] [s4, s7, s8, s9, s10] \n' +

' left \n' +

' [1] cfilter {(exists(s5) && exists(s6))} \n' +

' [1] ixseek s5 s6 s9 s4 s7 s8 [] @"be41ffec-f41a-481e-881f-beaf33748b79" @"rno\_1" true \n' +

' right \n' +

' [2] limit 1 \n' +

' [2] seek s4 s11 s12 s7 s8 s9 s10 [] @"be41ffec-f41a-481e-881f-beaf33748b79" true false \n'

}

},

rejectedPlans: []

},

executionStats: {

executionSuccess: true,

nReturned: 3,

executionTimeMillis: 21,

totalKeysExamined: 3,

totalDocsExamined: 3,

executionStages: {

stage: 'nlj',

planNodeId: 2,

nReturned: 3,

executionTimeMillisEstimate: 20,

opens: 1,

closes: 1,

saveState: 1,

restoreState: 1,

isEOF: 1,

totalDocsExamined: 3,

totalKeysExamined: 3,

collectionScans: 0,

collectionSeeks: 3,

indexScans: 0,

indexSeeks: 1,

indexesUsed: [ 'rno\_1' ],

innerOpens: 3,

innerCloses: 1,

outerProjects: [],

outerCorrelated: [ Long("4"), Long("7"), Long("8"), Long("9"), Long("10") ],

outerStage: {

stage: 'cfilter',

planNodeId: 1,

nReturned: 3,

executionTimeMillisEstimate: 20,

opens: 1,

closes: 1,

saveState: 1,

restoreState: 1,

isEOF: 1,

numTested: 1,

filter: '(exists(s5) && exists(s6)) ',

inputStage: {

stage: 'ixseek',

planNodeId: 1,

nReturned: 3,

executionTimeMillisEstimate: 20,

opens: 1,

closes: 1,

saveState: 1,

restoreState: 1,

isEOF: 1,

indexName: 'rno\_1',

keysExamined: 3,

seeks: 1,

numReads: 4,

indexKeySlot: 9,

recordIdSlot: 4,

snapshotIdSlot: 7,

indexIdentSlot: 8,

outputSlots: [],

indexKeysToInclude: '00000000000000000000000000000000',

seekKeyLow: 's5 ',

seekKeyHigh: 's6 '

}

},

innerStage: {

stage: 'limit',

planNodeId: 2,

nReturned: 3,

executionTimeMillisEstimate: 0,

opens: 3,

closes: 1,

saveState: 1,

restoreState: 1,

isEOF: 1,

limit: 1,

inputStage: {

stage: 'seek',

planNodeId: 2,

nReturned: 3,

executionTimeMillisEstimate: 0,

opens: 3,

closes: 1,

saveState: 1,

restoreState: 1,

isEOF: 0,

numReads: 3,

recordSlot: 11,

recordIdSlot: 12,

seekKeySlot: 4,

snapshotIdSlot: 7,

indexIdentSlot: 8,

indexKeySlot: 9,

indexKeyPatternSlot: 10,

fields: [],

outputSlots: []

}

}

}

},

command: { find: 'stud', filter: { rno: 11 }, '$db': 'Shubham' },

serverInfo: {

host: 'LAPTOP-ECO21OAA',

port: 27017,

version: '7.0.0',

gitVersion: '37d84072b5c5b9fd723db5fa133fb202ad2317f1'

},

serverParameters: {

internalQueryFacetBufferSizeBytes: 104857600,

internalQueryFacetMaxOutputDocSizeBytes: 104857600,

internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,

internalDocumentSourceGroupMaxMemoryBytes: 104857600,

internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,

internalQueryProhibitBlockingMergeOnMongoS: 0,

internalQueryMaxAddToSetBytes: 104857600,

internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,

internalQueryFrameworkControl: 'trySbeEngine'

},

ok: 1

}

Shubham> db.stud.getIndexes()

[

{ v: 2, key: { \_id: 1 }, name: '\_id\_' },

{ v: 2, key: { rno: 1 }, name: 'rno\_1' }

]

Shubham> db.stud.dropIndex('rno\_1')

{ nIndexesWas: 2, ok: 1 }

Shubham> db.stud.getIndexes()

[ { v: 2, key: { \_id: 1 }, name: '\_id\_' } ]

Shubham> db.stud.aggregate({[$group:{\_id:'$name' , count:{$sum:1}}]})

Uncaught:

SyntaxError: Unexpected token, expected "]" (1:26)

> 1 | db.stud.aggregate({[$group:{\_id:'$name' , count:{$sum:1}}]})

| ^

2 |

Shubham> db.stud.aggregate([{$group:{\_id:'$name' , count:{$sum:1}}}])

[

{ \_id: 'Pratik', count: 3 },

{ \_id: 'Shubham', count: 3 },

{ \_id: 'Sid', count: 3 },

{ \_id: 'Ram', count: 3 }

]

Shubham> db.stud.aggregate([{$group:{\_id:'$name' , marksAvg:{$avg:"$marks"}}}])

[

{ \_id: 'Pratik', marksAvg: 36.666666666666664 },

{ \_id: 'Shubham', marksAvg: 38.333333333333336 },

{ \_id: 'Ram', marksAvg: 43.333333333333336 },

{ \_id: 'Sid', marksAvg: 36.666666666666664 }

]

Shubham> db.stud.aggregate([{$group:{\_id:'$name' , marksMax:{$max:"$marks"}}}])

[

{ \_id: 'Pratik', marksMax: 50 },

{ \_id: 'Shubham', marksMax: 45 },

{ \_id: 'Sid', marksMax: 50 },

{ \_id: 'Ram', marksMax: 45 }

]

Shubham> db.stud.aggregate([{$group:{\_id:'$name' , marksMin:{$min:"$marks"}}}])

[

{ \_id: 'Sid', marksMin: 30 },

{ \_id: 'Shubham', marksMin: 30 },

{ \_id: 'Ram', marksMin: 40 },

{ \_id: 'Pratik', marksMin: 20 }

]

Shubham> db.stud.aggregate([ {$match:{marks:{$lte:30}}},{$group:{\_id:'$name' , failInSubject:{$push:"$subject"}}}])

[

{ \_id: 'Pratik', failInSubject: [ 'TOC' ] },

{ \_id: 'Sid', failInSubject: [ 'DBMS', 'CNS' ] },

{ \_id: 'Shubham', failInSubject: [ 'TOC' ] }

]

Shubham> db.stud.aggregate([{$group:{\_id:'$name' , totalMarks:{$sum:"$marks"}}}])

[

{ \_id: 'Pratik', totalMarks: 110 },

{ \_id: 'Ram', totalMarks: 130 },

{ \_id: 'Sid', totalMarks: 110 },

{ \_id: 'Shubham', totalMarks: 115 }

]

Shubham

db.stud.insertMany([{rno:36 , name:'Onkar' , subject:'DBMS' , marks:25},{rno:36 , name:'Onakr' , subject:'CNS' , marks:30},{rno:36 , name:'Onkar' , subject:'TOC' , marks:50}])