Yuwen Sang

A8: Mongo

**Part1: Queries**

**Q2: pid, state, and name of the children for people who have 7 children. Do not include object ids.**

>db.thePeople.aggregate([{$match:{numChildren:{$eq:7}}},{$project:{\_id:0, firstName:0, MI:0, lastName:0, age:0, birth:0, salary:0, numChildren:0, weight:0, height:0}}])

[ { pid: 3,

state: 'TX',

children:

[ 'Victoria M Brown',

'John I Brown',

'Priya J Brown',

'Rahul G Brown',

'Michelle G Brown',

'Linda Y Brown',

'Jennifer B Brown' ] },

{ pid: 513,

state: 'OR',

children:

[ 'Michelle W Jackson',

'David A Jackson',

'Victoria M Jackson',

'Santiago P Jackson',

'Isabella H Jackson',

'Mohammed G Jackson',

'Amy I Jackson' ] },

{ pid: 612,

state: 'SD',

children:

[ 'Joseph Q Wilson',

'Julie A Wilson',

'David X Wilson',

'Paul A Wilson',

'Sofia O Wilson',

'Amy Z Wilson',

'Amy Z Wilson' ] },

{ pid: 745,

state: 'IN',

children:

[ 'Amy T Zhao',

'Victoria E Zhao',

'Jayden E Zhao',

'Kayla V Zhao',

'Madison E Zhao',

'Jayden I Zhao',

'Neha D Zhao' ] },

{ pid: 816,

state: 'MD',

children:

[ 'Demarco Q Le',

'Vicky T Le',

'Isabella G Le',

'Jennifer R Le',

'Sofia Y Le',

'John Y Le',

'Diego H Le' ] },

{ pid: 881,

state: 'NE',

children:

[ 'Madison U Jones',

'Priya F Jones',

'Amy L Jones',

'Kayla T Jones',

'John F Jones',

'Isabella X Jones',

'Paul H Jones' ] },

{ pid: 993,

state: 'OR',

children:

[ 'Ashley C Wu',

'Demarco R Wu',

'Hasan C Wu',

'David N Wu',

'Ava U Wu',

'Jeffrey V Wu',

'Neha H Wu' ] },

{ pid: 1010,

state: 'MS',

children:

[ 'Bob X Park',

'Mohammad I Park',

'Amy F Park',

'Ashley N Park',

'Santiago Z Park',

'Julie D Park',

'Amy O Park' ] },

{ pid: 1240,

state: 'CT',

children:

[ 'Jayden V Wong',

'Demarco U Wong',

'Julie R Wong',

'Julie F Wong',

'Hannah K Wong',

'Jeffrey Z Wong',

'Julie O Wong' ] },

{ pid: 1321,

state: 'KS',

children:

[ 'Hasan V Anderson',

'Madison Z Anderson',

'Kayla M Anderson',

'Jayden D Anderson',

'John D Anderson',

'Julia J Anderson',

'John N Anderson' ] },

{ pid: 1343,

state: 'KY',

children:

[ 'Jennifer C Tanaka',

'Amy R Tanaka',

'Hannah U Tanaka',

'Isabella I Tanaka',

'Michelle Z Tanaka',

'Mohammed D Tanaka',

'Mohammad P Tanaka' ] },

{ pid: 1537,

state: 'KS',

children:

[ 'John Z Sato',

'Madison F Sato',

'Jeffrey T Sato',

'Demarco P Sato',

'Santiago I Sato',

'Victoria A Sato',

'Jennifer K Sato' ] },

{ pid: 1791,

state: 'VT',

children:

[ 'Julie K Williams',

'David Z Williams',

'Vivek G Williams',

'Madison B Williams',

'Rahul N Williams',

'Alejandro A Williams',

'Alejandro R Williams' ] },

{ pid: 1879,

state: 'WV',

children:

[ 'Noah Y Jones',

'Paul K Jones',

'Noah B Jones',

'Michelle B Jones',

'Sofia Y Jones',

'Diego V Jones',

'Victoria R Jones' ] },

{ pid: 1994,

state: 'AL',

children:

[ 'Priya D Lee',

'Mary D Lee',

'Kayla G Lee',

'Isabella O Lee',

'Priya H Lee',

'Ashley H Lee',

'Jayden Z Lee' ] } ]

**Q4: Complete info of people who live in CA and have 5 or 6 children**

>db.thePeople.aggregate([{$match:{$or:[{numChildren:{$eq:5}},{numChildren:{$eq:6}}]}},{$match:{state:{$eq:"CA"}}}])

[ { \_id: ObjectId("604fe1a25701ce73669814c4"),

pid: 169,

firstName: 'Amit',

MI: 'X',

lastName: 'Lee',

state: 'CA',

age: 117,

birth: 1902,

salary: 61614,

numChildren: 6,

children:

[ 'Sarah G Lee',

'Madison D Lee',

'Victoria O Lee',

'Isabella C Lee',

'William O Lee',

'Ava Q Lee' ],

weight: 114,

height: 180 },

{ \_id: ObjectId("604fe1a35701ce7366981701"),

pid: 742,

firstName: 'Demarco',

MI: 'Y',

lastName: 'Jackson',

state: 'CA',

age: 84,

birth: 1935,

salary: 67539,

numChildren: 5,

children:

[ 'Bob F Jackson',

'Daniella I Jackson',

'Peter U Jackson',

'Julie F Jackson',

'Neha J Jackson' ],

weight: 134,

height: 191 },

{ \_id: ObjectId("604fe1a35701ce73669817dc"),

pid: 961,

firstName: 'Fatimah',

MI: 'X',

lastName: 'Baker',

state: 'CA',

age: 84,

birth: 1935,

salary: 118150,

numChildren: 5,

children:

[ 'Victoria V Baker',

'Mary B Baker',

'Bob F Baker',

'Kayla L Baker',

'Amit R Baker' ],

weight: 62,

height: 197 },

{ \_id: ObjectId("604fe1a45701ce736698199b"),

pid: 1408,

firstName: 'Mary',

MI: 'X',

lastName: 'Brown',

state: 'CA',

age: 30,

birth: 1989,

salary: 43826,

numChildren: 5,

children:

[ 'Hannah Y Brown',

'Hasan Q Brown',

'Peter D Brown',

'Jayla D Brown',

'William X Brown' ],

weight: 82,

height: 189 } ]

**Q5: List the pid and children names for all people who have a child whose name contains 'Bob A':**

>db.thePeople.aggregate([{$match:{children:{$regex: "Bob A"}}},{$project:{\_id:0, firstName:0, MI:0, lastName:0, state:0, age:0, birth:0, salary:0, numChildren:0, weight:0, height:0}}])

[ { pid: 692,

children:

[ 'Bob A Chen',

'Linda P Chen',

'Madison Z Chen',

'Linda P Chen',

'Jennifer D Chen' ] },

{ pid: 991,

children:

[ 'Mohammad N Chan',

'Bob A Chan',

'Madison T Chan',

'Victoria N Chan' ] },

{ pid: 1612,

children:

[ 'Bob A Takahashi',

'Neha H Takahashi',

'Jennifer L Takahashi',

'Hannah C Takahashi' ] } ]

**Q9: Aggregation: average/min/max salary for midwest state, where I am assuming there are 12 midwest states:**

>db.thePeople.aggregate([{$match:{$or:[{state:{$eq:"ND"}},{state:{$eq:"SD"}},{state:{$eq:"NE"}},{state:{$eq:"KS"}},{state:{$eq:"MN"}},{state:{$eq:"IA"}},{state:{$eq:"MS"}},{state:{$eq:"WI"}},{state:{$eq:"IL"}},{state:{$eq:"IN"}},{state:{$eq:"MI"}},{state:{$eq:"OH"}}]}}, {$group:{\_id:"$state",avgSalary:{$avg:"$salary"},minSalary:{$min:"$salary"},maxSalary:{$max:"$salary"},"numInGroup":{$sum:1}}},{$sort:{"\_id":1}}])

[ { \_id: 'IA',

avgSalary: 83429.84615384616,

minSalary: 32606,

maxSalary: 128314,

numInGroup: 39 },

{ \_id: 'IL',

avgSalary: 74737.85294117648,

minSalary: 31481,

maxSalary: 114905,

numInGroup: 34 },

{ \_id: 'IN',

avgSalary: 85336.21212121213,

minSalary: 36705,

maxSalary: 129445,

numInGroup: 33 },

{ \_id: 'KS',

avgSalary: 76605.38888888889,

minSalary: 37380,

maxSalary: 125949,

numInGroup: 36 },

{ \_id: 'MI',

avgSalary: 78596.93023255814,

minSalary: 30837,

maxSalary: 128545,

numInGroup: 43 },

{ \_id: 'MN',

avgSalary: 79549.45714285714,

minSalary: 33945,

maxSalary: 126530,

numInGroup: 35 },

{ \_id: 'MS',

avgSalary: 82858.68,

minSalary: 30591,

maxSalary: 124771,

numInGroup: 25 },

{ \_id: 'ND',

avgSalary: 84828.57142857143,

minSalary: 31346,

maxSalary: 126747,

numInGroup: 42 },

{ \_id: 'NE',

avgSalary: 76561.78723404255,

minSalary: 31571,

maxSalary: 125839,

numInGroup: 47 },

{ \_id: 'OH',

avgSalary: 81283.39130434782,

minSalary: 32641,

maxSalary: 129161,

numInGroup: 46 },

{ \_id: 'SD',

avgSalary: 81643.41666666667,

minSalary: 35200,

maxSalary: 128021,

numInGroup: 36 },

{ \_id: 'WI',

avgSalary: 79170.7435897436,

minSalary: 30949,

maxSalary: 126922,

numInGroup: 39 } ]

**Q10: Aggregation: avgerage salary in states where the average salary within that state is >= 82,000 and how many people in the grouping for each state:**

>db.thePeople.aggregate([{$group:{\_id:"$state",avgSalary:{$avg:"$salary"},numInGroup:{$sum:1}}},{$match:{avgSalary:{$gte:82000}}},{$sort:{"\_id":1}}])

[ { \_id: 'CO', avgSalary: 82695.52380952382, numInGroup: 42 },

{ \_id: 'IA', avgSalary: 83429.84615384616, numInGroup: 39 },

{ \_id: 'IN', avgSalary: 85336.21212121213, numInGroup: 33 },

{ \_id: 'MD', avgSalary: 87204.64406779662, numInGroup: 59 },

{ \_id: 'MS', avgSalary: 82858.68, numInGroup: 25 },

{ \_id: 'MT', avgSalary: 83048.78048780488, numInGroup: 41 },

{ \_id: 'NC', avgSalary: 83518.30555555556, numInGroup: 36 },

{ \_id: 'ND', avgSalary: 84828.57142857143, numInGroup: 42 },

{ \_id: 'NH', avgSalary: 83915.575, numInGroup: 40 },

{ \_id: 'NM', avgSalary: 83036.27777777778, numInGroup: 36 },

{ \_id: 'NY', avgSalary: 82193.19565217392, numInGroup: 46 },

{ \_id: 'OK', avgSalary: 85440.54838709677, numInGroup: 31 },

{ \_id: 'PA', avgSalary: 85585.52173913043, numInGroup: 46 },

{ \_id: 'SC', avgSalary: 85474.84615384616, numInGroup: 39 },

{ \_id: 'VT', avgSalary: 89343.45714285714, numInGroup: 35 },

{ \_id: 'WV', avgSalary: 87893.84210526316, numInGroup: 38 } ]

**Q11: Aggregation: average/min/max salary for midwest state whose average salary > 82000:**

>db.thePeople.aggregate([{$match:{$or:[{state:{$eq:"ND"}},{state:{$eq:"SD"}},{state:{$eq:"NE"}},{state:{$eq:"KS"}},{state:{$eq:"MN"}},{state:{$eq:"IA"}},{state:{$eq:"MS"}},{state:{$eq:"WI"}},{state:{$eq:"IL"}},{state:{$eq:"IN"}},{state:{$eq:"MI"}},{state:{$eq:"OH"}}]}}, {$group:{\_id:"$state",avgSalary:{$avg:"$salary"},minSalary:{$min:"$salary"},maxSalary:{$max:"$salary"},numInGroup:{$sum:1}}}, {$match:{avgSalary:{$gt:82000}}},{$sort:{"\_id":1}}])

[ { \_id: 'IA',

avgSalary: 83429.84615384616,

minSalary: 32606,

maxSalary: 128314,

numInGroup: 39 },

{ \_id: 'IN',

avgSalary: 85336.21212121213,

minSalary: 36705,

maxSalary: 129445,

numInGroup: 33 },

{ \_id: 'MS',

avgSalary: 82858.68,

minSalary: 30591,

maxSalary: 124771,

numInGroup: 25 },

{ \_id: 'ND',

avgSalary: 84828.57142857143,

minSalary: 31346,

maxSalary: 126747,

numInGroup: 42 } ]

**Part2: Updates/Deletes**

1. Change 1 Tuple:

Based on the result of the query 2, change the people whose pid = 3, its state to “CA”.

Before Update, the people whose id = 3, was from “TX” (see part1 query 2).

Then, after input the command in MongoDB:

*>db.thePeople.updateOne({pid:3},{$set:{state:"CA"}})*

Run Q2 command again, we get, for pid= 3:

[ { pid: 3,

state: 'CA',

children:

[ 'Victoria M Brown',

'John I Brown',

'Priya J Brown',

'Rahul G Brown',

'Michelle G Brown',

'Linda Y Brown',

'Jennifer B Brown' ] },

1. Change Multiple Tuples

Let’s change the people living in CA their first name to “ABC”.

>db.thePeople.updateMany({state:{$eq:*"CA"*}},{$set:{firstName: *"ABC"*}})

After the update, we get the new result of Q4 will be:

[ { \_id: ObjectId("604fe1a25701ce73669814c4"),

pid: 169,

firstName: 'ABC',

MI: 'X',

lastName: 'Lee',

state: 'CA',

age: 117,

birth: 1902,

salary: 61614,

numChildren: 6,

children:

[ 'Sarah G Lee',

'Madison D Lee',

'Victoria O Lee',

'Isabella C Lee',

'William O Lee',

'Ava Q Lee' ],

weight: 114,

height: 180 },

{ \_id: ObjectId("604fe1a35701ce7366981701"),

pid: 742,

firstName: 'ABC',

MI: 'Y',

lastName: 'Jackson',

state: 'CA',

age: 84,

birth: 1935,

salary: 67539,

numChildren: 5,

children:

[ 'Bob F Jackson',

'Daniella I Jackson',

'Peter U Jackson',

'Julie F Jackson',

'Neha J Jackson' ],

weight: 134,

height: 191 },

{ \_id: ObjectId("604fe1a35701ce73669817dc"),

pid: 961,

firstName: 'ABC',

MI: 'X',

lastName: 'Baker',

state: 'CA',

age: 84,

birth: 1935,

salary: 118150,

numChildren: 5,

children:

[ 'Victoria V Baker',

'Mary B Baker',

'Bob F Baker',

'Kayla L Baker',

'Amit R Baker' ],

weight: 62,

height: 197 },

{ \_id: ObjectId("604fe1a45701ce736698199b"),

pid: 1408,

firstName: 'ABC',

MI: 'X',

lastName: 'Brown',

state: 'CA',

age: 30,

birth: 1989,

salary: 43826,

numChildren: 5,

children:

[ 'Hannah Y Brown',

'Hasan Q Brown',

'Peter D Brown',

'Jayla D Brown',

'William X Brown' ],

weight: 82,

height: 189 } ]

1. Delete multiple documents

For query 2, we can see there are 2 people living in KS and having 7 children. Let’s delete them:

> db.thePeople.deleteMany({$and:[{ numChildren:{$eq:7}},{state:{$eq:"KS"}}]})

{ acknowledged: true, deletedCount: 2 }

Then run Q2 again:

[ { pid: 3,

state: 'CA',

children:

[ 'Victoria M Brown',

'John I Brown',

'Priya J Brown',

'Rahul G Brown',

'Michelle G Brown',

'Linda Y Brown',

'Jennifer B Brown' ] },

{ pid: 513,

state: 'OR',

children:

[ 'Michelle W Jackson',

'David A Jackson',

'Victoria M Jackson',

'Santiago P Jackson',

'Isabella H Jackson',

'Mohammed G Jackson',

'Amy I Jackson' ] },

{ pid: 612,

state: 'SD',

children:

[ 'Joseph Q Wilson',

'Julie A Wilson',

'David X Wilson',

'Paul A Wilson',

'Sofia O Wilson',

'Amy Z Wilson',

'Amy Z Wilson' ] },

{ pid: 745,

state: 'IN',

children:

[ 'Amy T Zhao',

'Victoria E Zhao',

'Jayden E Zhao',

'Kayla V Zhao',

'Madison E Zhao',

'Jayden I Zhao',

'Neha D Zhao' ] },

{ pid: 816,

state: 'MD',

children:

[ 'Demarco Q Le',

'Vicky T Le',

'Isabella G Le',

'Jennifer R Le',

'Sofia Y Le',

'John Y Le',

'Diego H Le' ] },

{ pid: 881,

state: 'NE',

children:

[ 'Madison U Jones',

'Priya F Jones',

'Amy L Jones',

'Kayla T Jones',

'John F Jones',

'Isabella X Jones',

'Paul H Jones' ] },

{ pid: 993,

state: 'OR',

children:

[ 'Ashley C Wu',

'Demarco R Wu',

'Hasan C Wu',

'David N Wu',

'Ava U Wu',

'Jeffrey V Wu',

'Neha H Wu' ] },

{ pid: 1010,

state: 'MS',

children:

[ 'Bob X Park',

'Mohammad I Park',

'Amy F Park',

'Ashley N Park',

'Santiago Z Park',

'Julie D Park',

'Amy O Park' ] },

{ pid: 1240,

state: 'CT',

children:

[ 'Jayden V Wong',

'Demarco U Wong',

'Julie R Wong',

'Julie F Wong',

'Hannah K Wong',

'Jeffrey Z Wong',

'Julie O Wong' ] },

{ pid: 1343,

state: 'KY',

children:

[ 'Jennifer C Tanaka',

'Amy R Tanaka',

'Hannah U Tanaka',

'Isabella I Tanaka',

'Michelle Z Tanaka',

'Mohammed D Tanaka',

'Mohammad P Tanaka' ] },

{ pid: 1791,

state: 'VT',

children:

[ 'Julie K Williams',

'David Z Williams',

'Vivek G Williams',

'Madison B Williams',

'Rahul N Williams',

'Alejandro A Williams',

'Alejandro R Williams' ] },

{ pid: 1879,

state: 'WV',

children:

[ 'Noah Y Jones',

'Paul K Jones',

'Noah B Jones',

'Michelle B Jones',

'Sofia Y Jones',

'Diego V Jones',

'Victoria R Jones' ] },

{ pid: 1994,

state: 'AL',

children:

[ 'Priya D Lee',

'Mary D Lee',

'Kayla G Lee',

'Isabella O Lee',

'Priya H Lee',

'Ashley H Lee',

'Jayden Z Lee' ] } ]

**Part 3: Indexing**

Now let’s expand the database into 200,000, and find the people that has more than 1 children.

Then, before using index, the query is:

*>db.thePeople.find({numChildren:{$gt:1}}).explain("executionStats")*

The result is:

{ queryPlanner:

{ plannerVersion: 1,

namespace: 'db\_people.thePeople',

indexFilterSet: false,

parsedQuery: { numChildren: { '$gt': 1 } },

winningPlan:

{ stage: 'COLLSCAN',

filter: { numChildren: { '$gt': 1 } },

direction: 'forward' },

rejectedPlans: [] },

executionStats:

{ executionSuccess: true,

nReturned: 173181,

executionTimeMillis: 348,

totalKeysExamined: 0,

totalDocsExamined: 200000,

executionStages:

{ stage: 'COLLSCAN',

filter: { numChildren: { '$gt': 1 } },

nReturned: 173181,

executionTimeMillisEstimate: 26,

works: 200002,

advanced: 173181,

needTime: 26820,

needYield: 0,

saveState: 200,

restoreState: 200,

isEOF: 1,

direction: 'forward',

docsExamined: 200000 } },

serverInfo:

{ host: 'LAPTOP-ANQ85J7N',

port: 27017,

version: '4.4.4',

gitVersion: '8db30a63db1a9d84bdcad0c83369623f708e0397' },

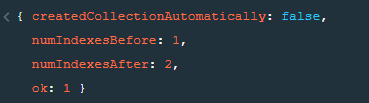
ok: 1 }

,which spend 348 milsec.

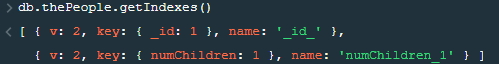
Now create an index:

>db.thePeople.ensureIndex({numChildren:1})

We get return:



Test the index that we just created:



Run the query again:

*>db.thePeople.find({numChildren:{$gt:1}}).explain("executionStats")*

We get the result:

{ queryPlanner:

{ plannerVersion: 1,

namespace: 'db\_people.thePeople',

indexFilterSet: false,

parsedQuery: { numChildren: { '$gt': 1 } },

winningPlan:

{ stage: 'FETCH',

inputStage:

{ stage: 'IXSCAN',

keyPattern: { numChildren: 1 },

indexName: 'numChildren\_1',

isMultiKey: false,

multiKeyPaths: { numChildren: [] },

isUnique: false,

isSparse: false,

isPartial: false,

indexVersion: 2,

direction: 'forward',

indexBounds: { numChildren: [ '(1, inf.0]' ] } } },

rejectedPlans: [] },

executionStats:

{ executionSuccess: true,

nReturned: 173181,

executionTimeMillis: 27,

totalKeysExamined: 173181,

totalDocsExamined: 173181,

executionStages:

{ stage: 'FETCH',

nReturned: 173181,

executionTimeMillisEstimate: 7,

works: 173182,

advanced: 173181,

needTime: 0,

needYield: 0,

saveState: 173,

restoreState: 173,

isEOF: 1,

docsExamined: 173181,

alreadyHasObj: 0,

inputStage:

{ stage: 'IXSCAN',

nReturned: 173181,

executionTimeMillisEstimate: 3,

works: 173182,

advanced: 173181,

needTime: 0,

needYield: 0,

saveState: 173,

restoreState: 173,

isEOF: 1,

keyPattern: { numChildren: 1 },

indexName: 'numChildren\_1',

isMultiKey: false,

multiKeyPaths: { numChildren: [] },

isUnique: false,

isSparse: false,

isPartial: false,

indexVersion: 2,

direction: 'forward',

indexBounds: { numChildren: [ '(1, inf.0]' ] },

keysExamined: 173181,

seeks: 1,

dupsTested: 0,

dupsDropped: 0 } } },

serverInfo:

{ host: 'LAPTOP-ANQ85J7N',

port: 27017,

version: '4.4.4',

gitVersion: '8db30a63db1a9d84bdcad0c83369623f708e0397' },

ok: 1 }

, which using indexing, the *execution time* reduced from 348 to 27*, execution time millis* estimate from 26 to 7 milsec.