**Describir el funcionamiento de cada ejemplo - AgregateAssignment**

**Se crea la colección db.universities y se insertan los datos de la misma.**

|  |
| --- |
| db.universities.insert([  {  country : 'Spain',  city : 'Salamanca',  name : 'USAL',  location : {  type : 'Point',  coordinates : [ -5.6722512,17, 40.9607792 ]  },  students : [  { year : 2014, number : 24774 },  { year : 2015, number : 23166 },  { year : 2016, number : 21913 },  { year : 2017, number : 21715 }  ]  },  {  country : 'Spain',  city : 'Salamanca',  name : 'UPSA',  location : {  type : 'Point',  coordinates : [ -5.6691191,17, 40.9631732 ]  },  students : [  { year : 2014, number : 4788 },  { year : 2015, number : 4821 },  { year : 2016, number : 6550 },  { year : 2017, number : 6125 }  ]  }  ]); |

**Se crea la colección db.courses y se insertan los datos de la misma.**

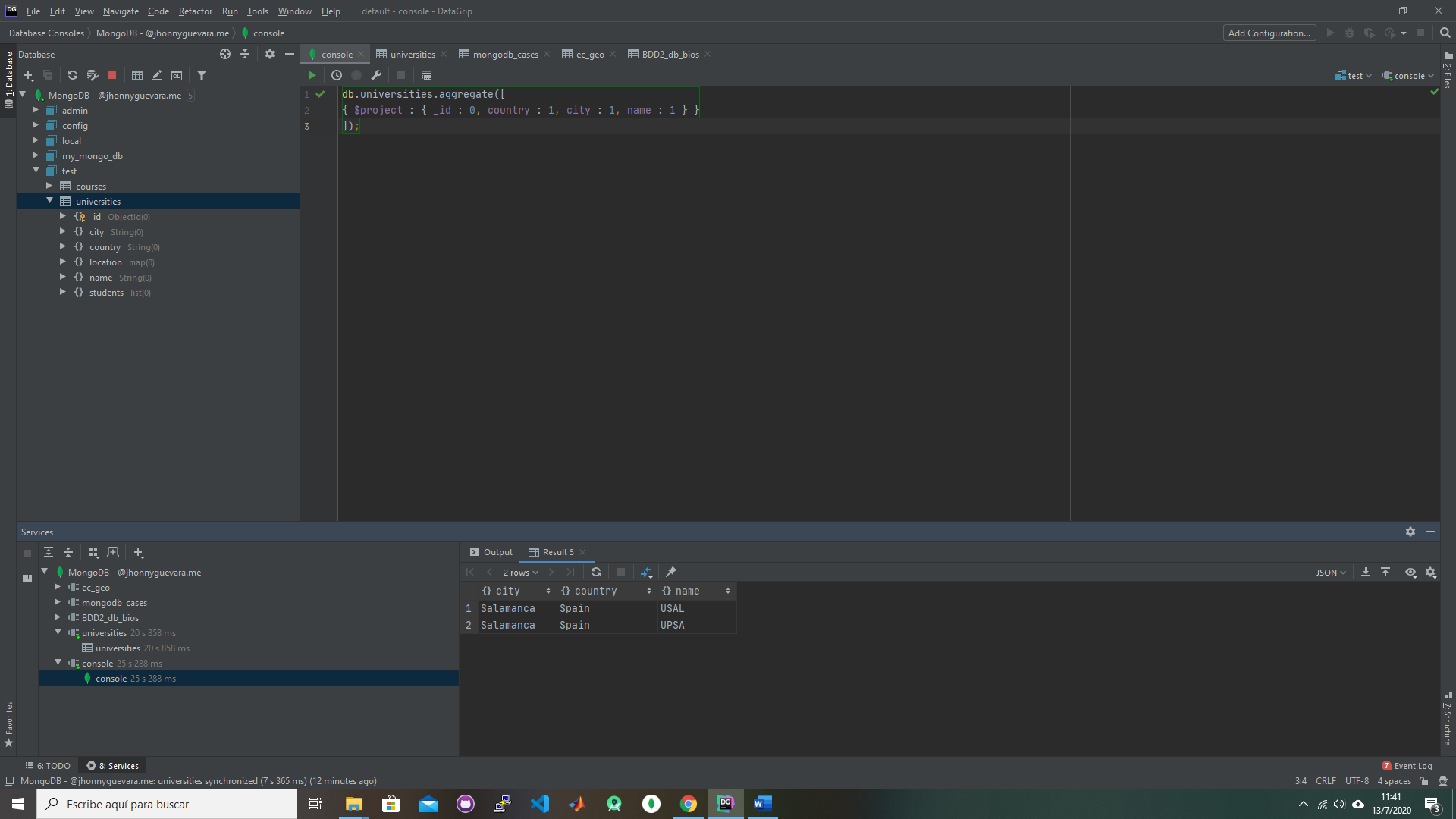
|  |
| --- |
| db.courses.insert([ { university : 'USAL', name : 'Computer Science', level : 'Excellent' }, { university : 'USAL', name : 'Electronics', level : 'Intermediate' }, { university : 'USAL', name : 'Communication', level : 'Excellent' } ]); |

**Especifica los documentos donde Spain es el valor del campo country y Salamanca el valor del campo city.**

|  |
| --- |
| db.universities.aggregate([ { $match : { country : 'Spain', city : 'Salamanca' } } ]); |

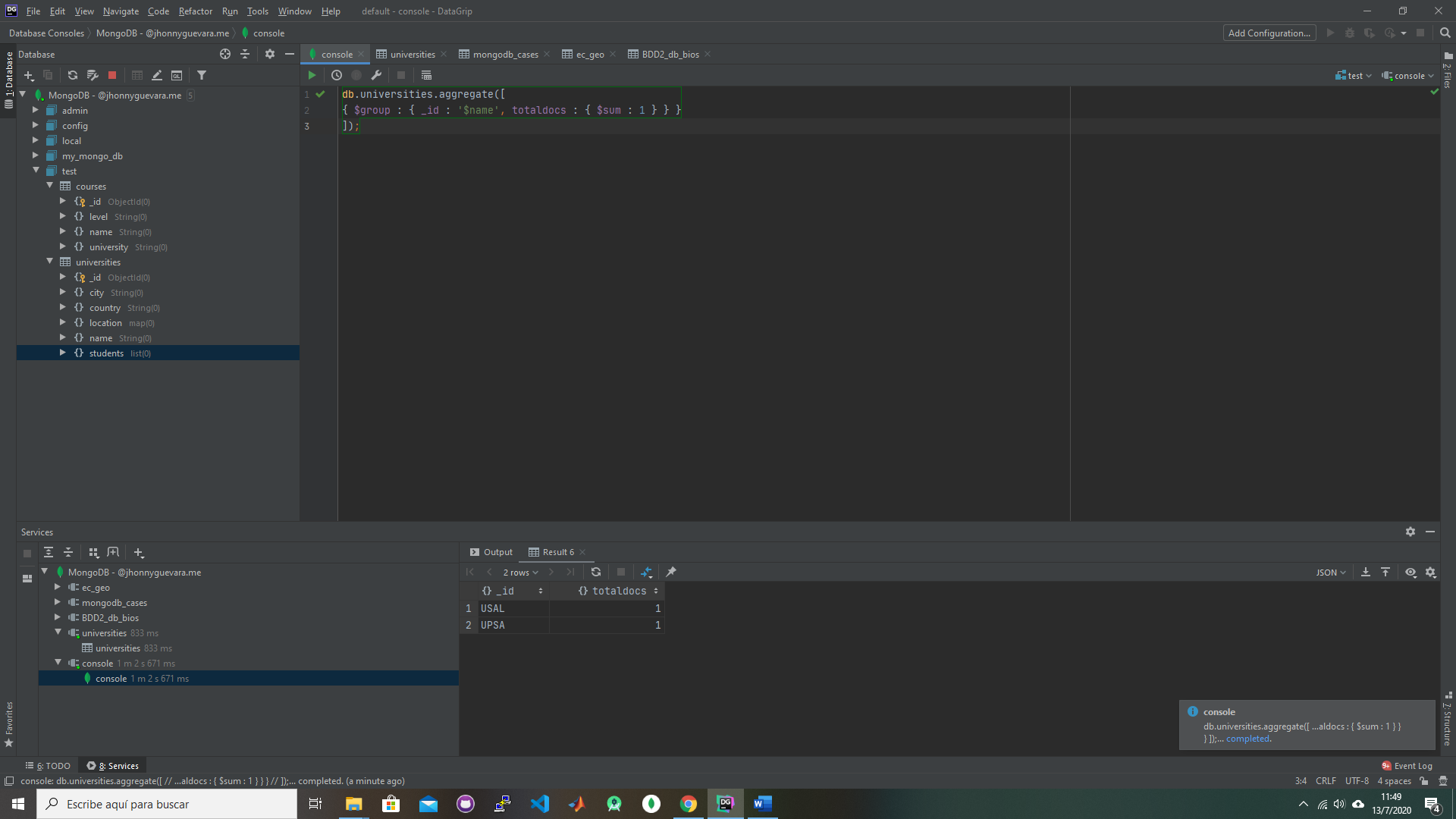
**Nos muestra los datos que necesitamos y especifiquemos en la base.**

|  |
| --- |
| db.universities.aggregate([ { $project : { \_id : 0, country : 1, city : 1, name : 1 } } ]); |



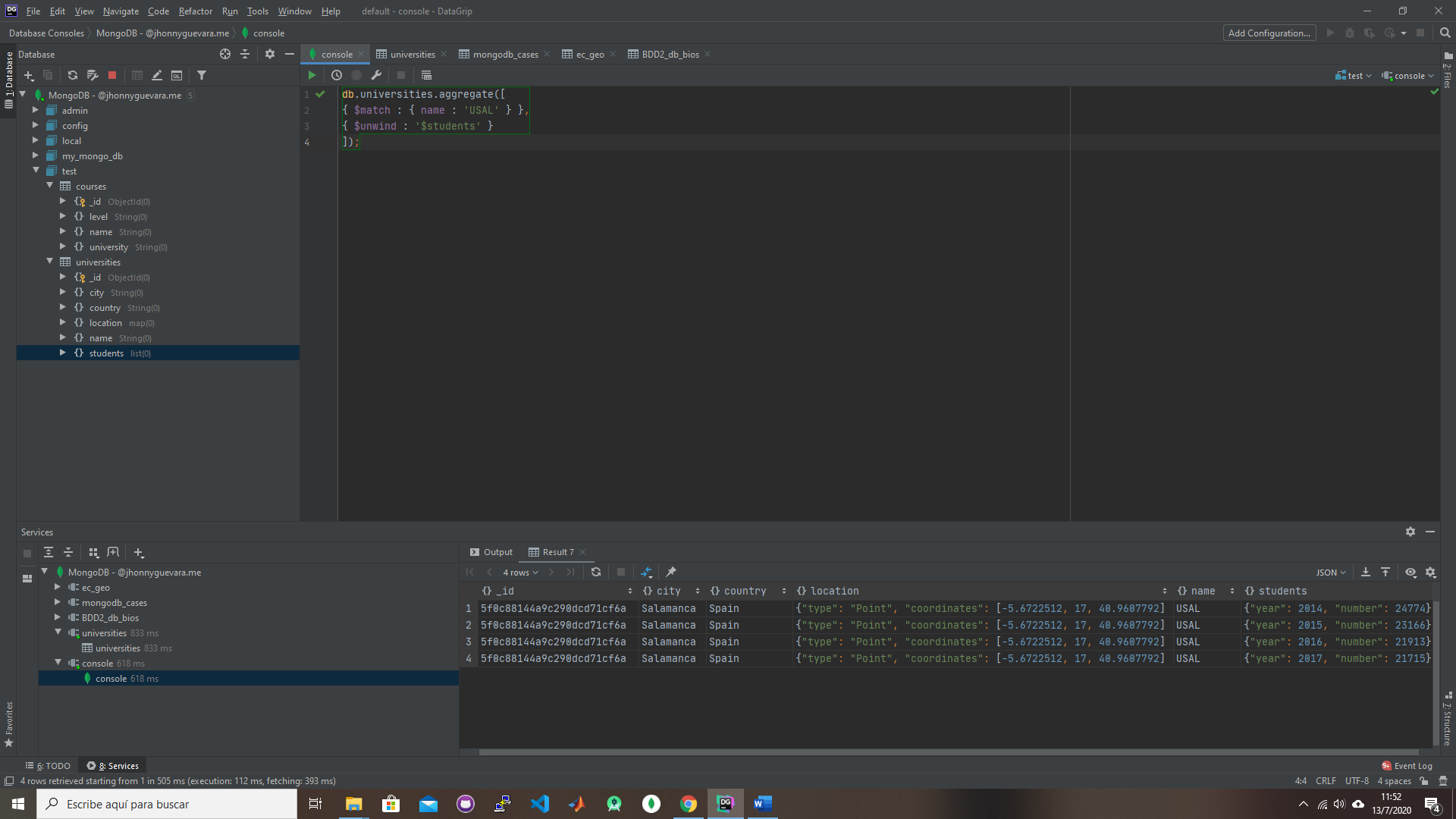
**Nos muestra la cantidad de documentos por universidad en nuestra colección.**

|  |
| --- |
| db.universities.aggregate([ { $group : { \_id : '$name', totaldocs : { $sum : 1 } } } ]); |



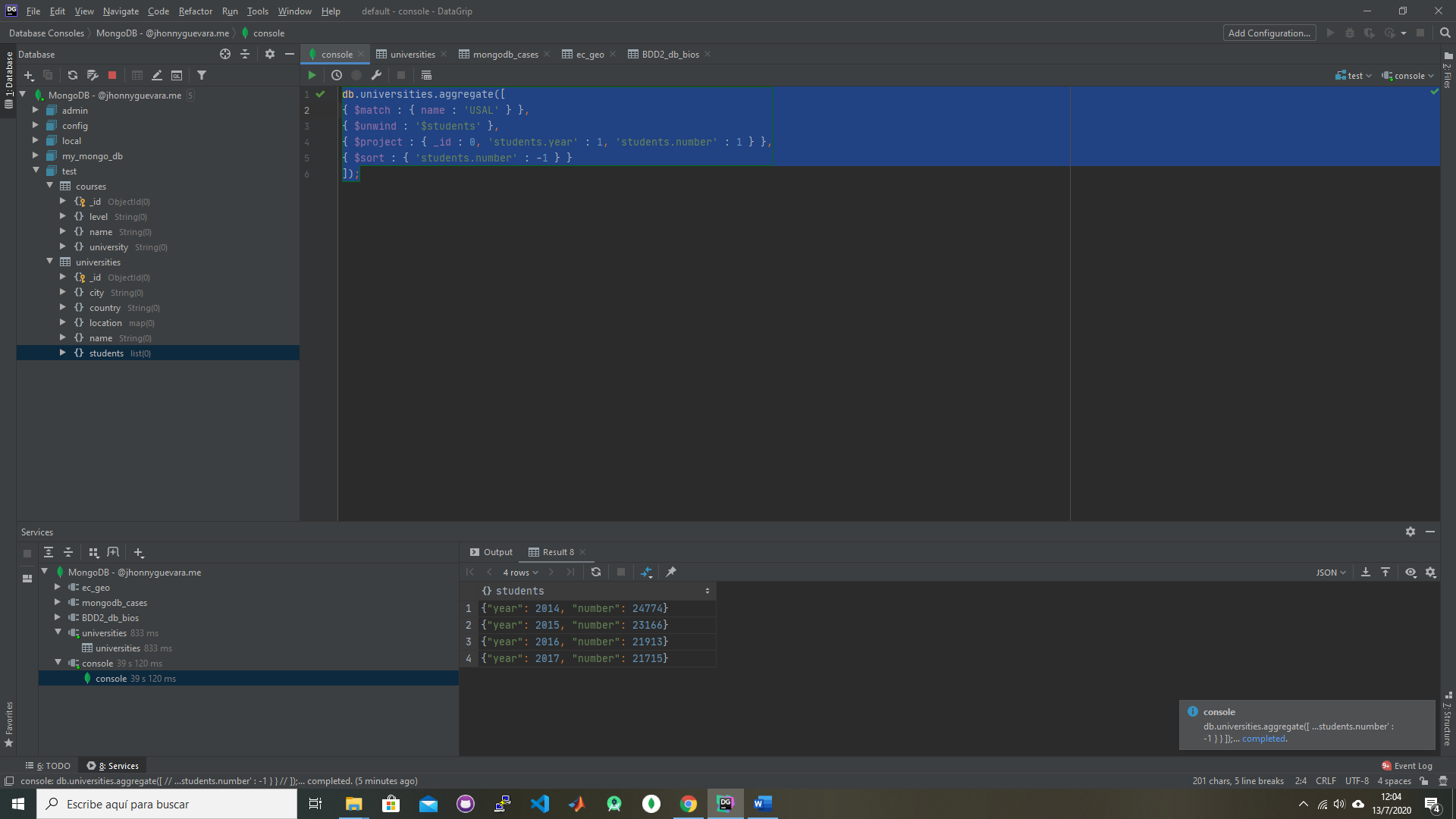
**Se aplica el método $unwind(), sobre la matriz del students, y verificamos que obtenga un documento por cada elemento de la matriz.**

|  |
| --- |
| db.universities.aggregate([ { $match : { name : 'USAL' } }, { $unwind : '$students' } ]); |



**Se aplica el método $sort(), para clasificar el resultado del método $unwind por el número de estudiantes y año en orden descendente.**

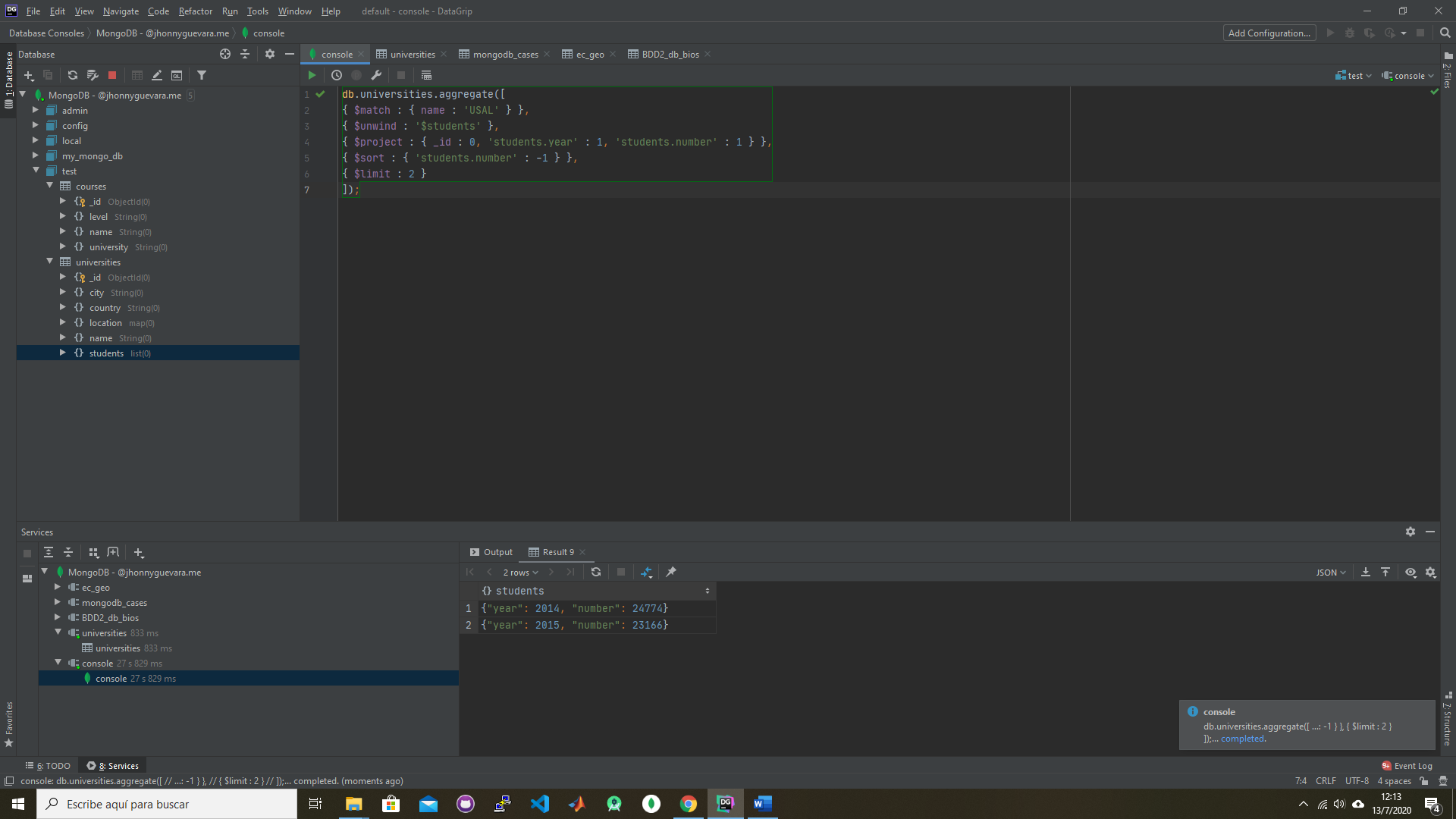
|  |
| --- |
| db.universities.aggregate([ { $match : { name : 'USAL' } }, { $unwind : '$students' }, { $project : { \_id : 0, 'students.year' : 1, 'students.number' : 1 } }, { $sort : { 'students.number' : -1 } } ]); |



**Se aplica el método $sort(), para clasificar el resultado del método $unwind por el número de estudiantes y año en orden descendente.**

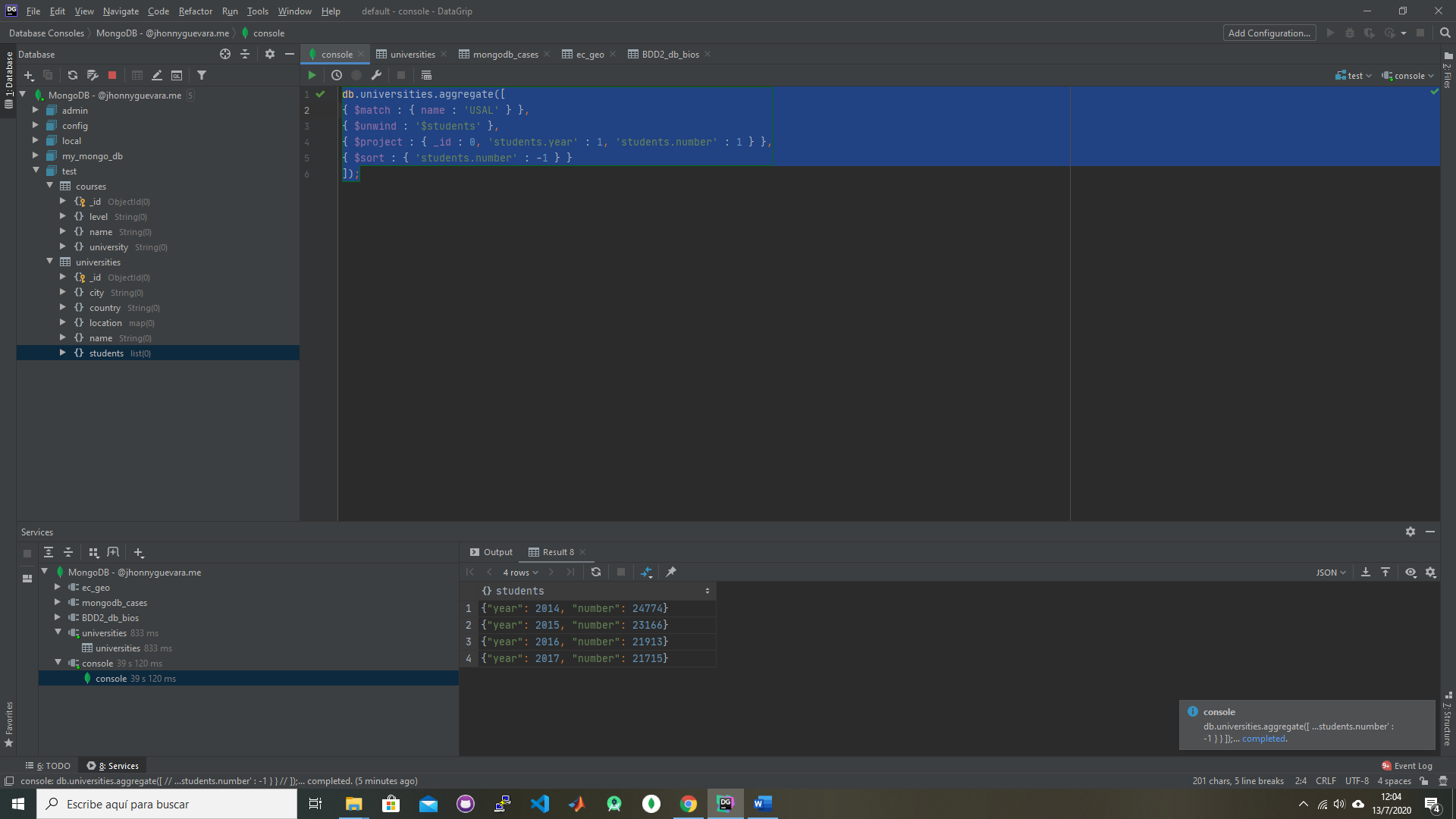
**Pero ahora** $limit para limitar el número de documentos ordenados a mostrar.

|  |
| --- |
| db.universities.aggregate([ { $match : { name : 'USAL' } }, { $unwind : '$students' }, { $project : { \_id : 0, 'students.year' : 1, 'students.number' : 1 } }, { $sort : { 'students.number' : -1 } }, { $limit : 2 } ]); |



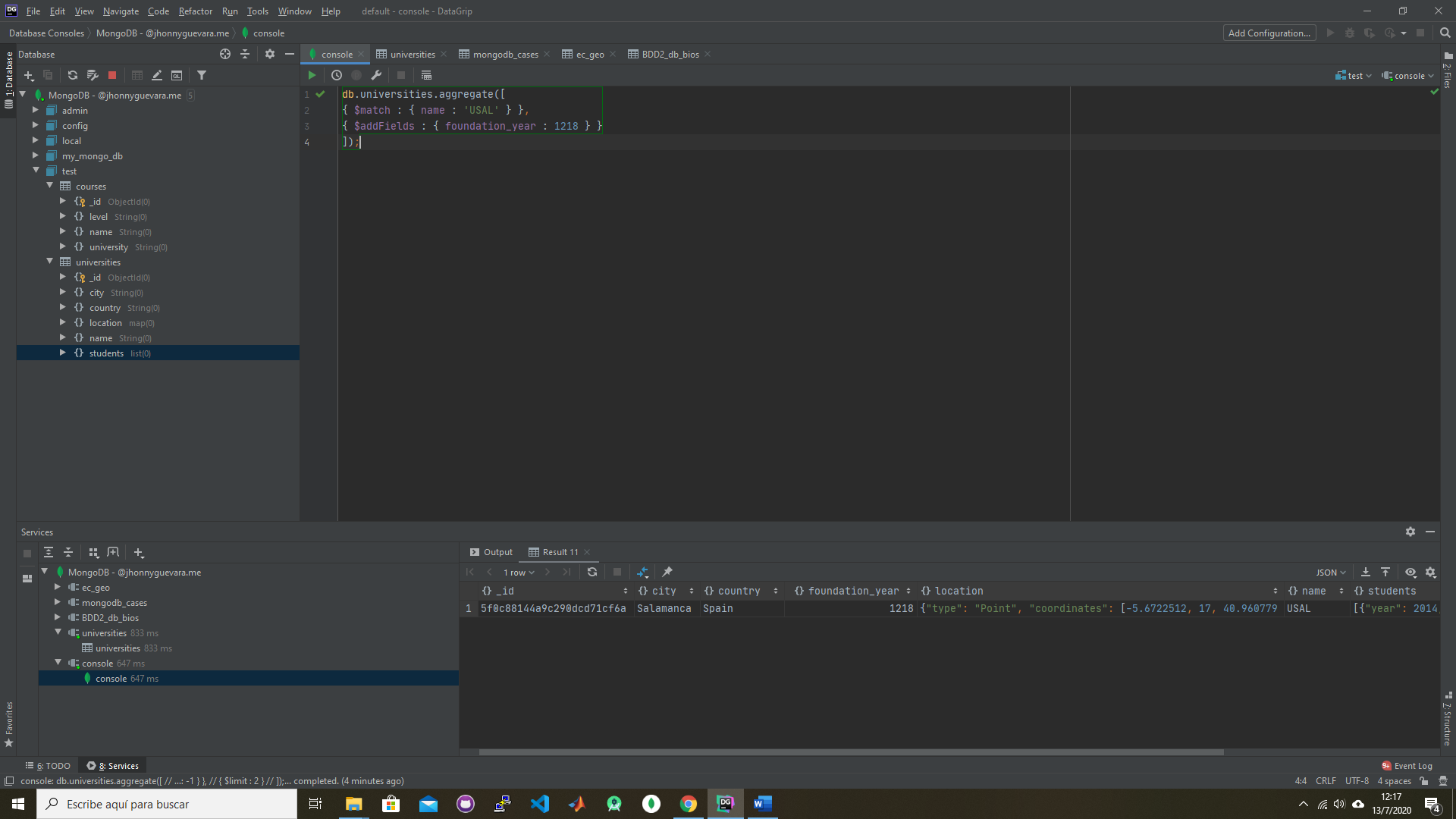
**Se aplica el método $sort(), para clasificar el resultado del método $unwind por el número de estudiantes y año en orden descendente.**

|  |
| --- |
| db.universities.aggregate([ { $match : { name : 'USAL' } }, { $unwind : '$students' }, { $project : { \_id : 0, 'students.year' : 1, 'students.number' : 1 } }, { $sort : { 'students.number' : -1 } } ]); |



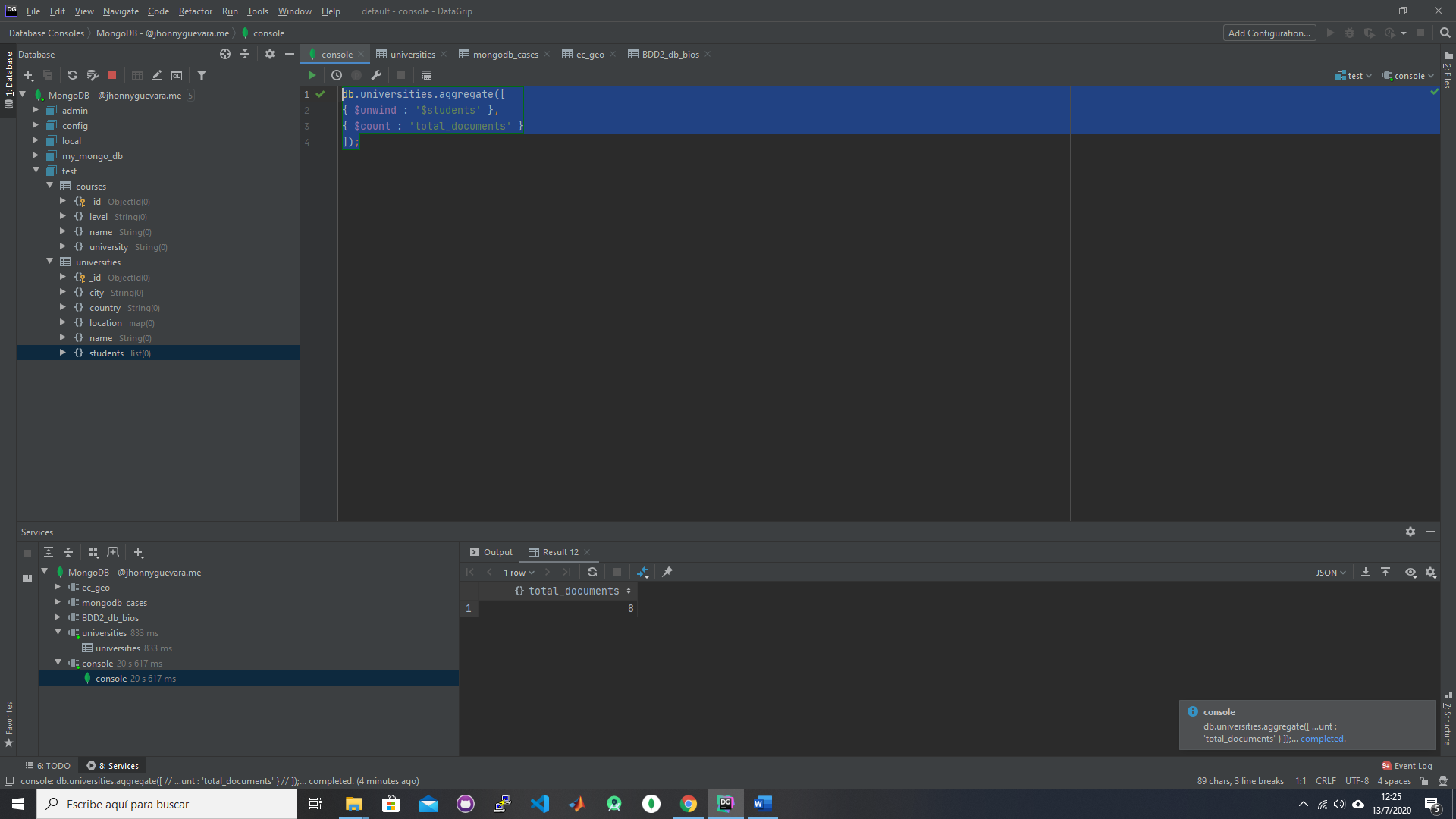
**Si necesita hacer algunos y agregar un nuevo campo utilizamos el método $addFields. Por ejemplo ahí se ve como se agrego el año de la fundación.**

|  |
| --- |
| db.universities.aggregate([ { $match : { name : 'USAL' } }, { $addFields : { foundation\_year : 1218 } } ]); |



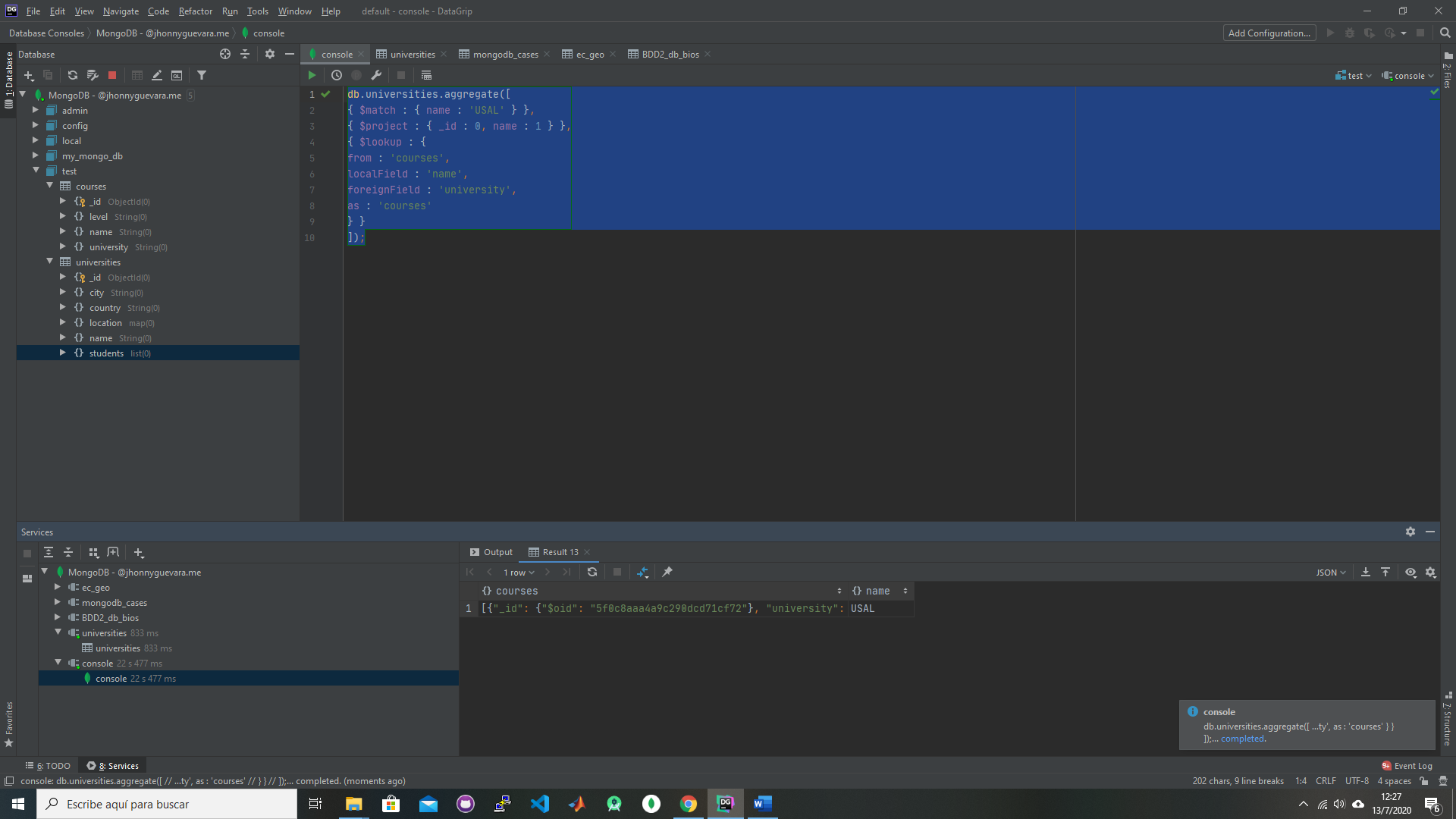
**El método $count() muestra la cantidad de documentos en la pipeline.**

|  |
| --- |
| db.universities.aggregate([ { $unwind : '$students' }, { $count : 'total\_documents' } ]); |



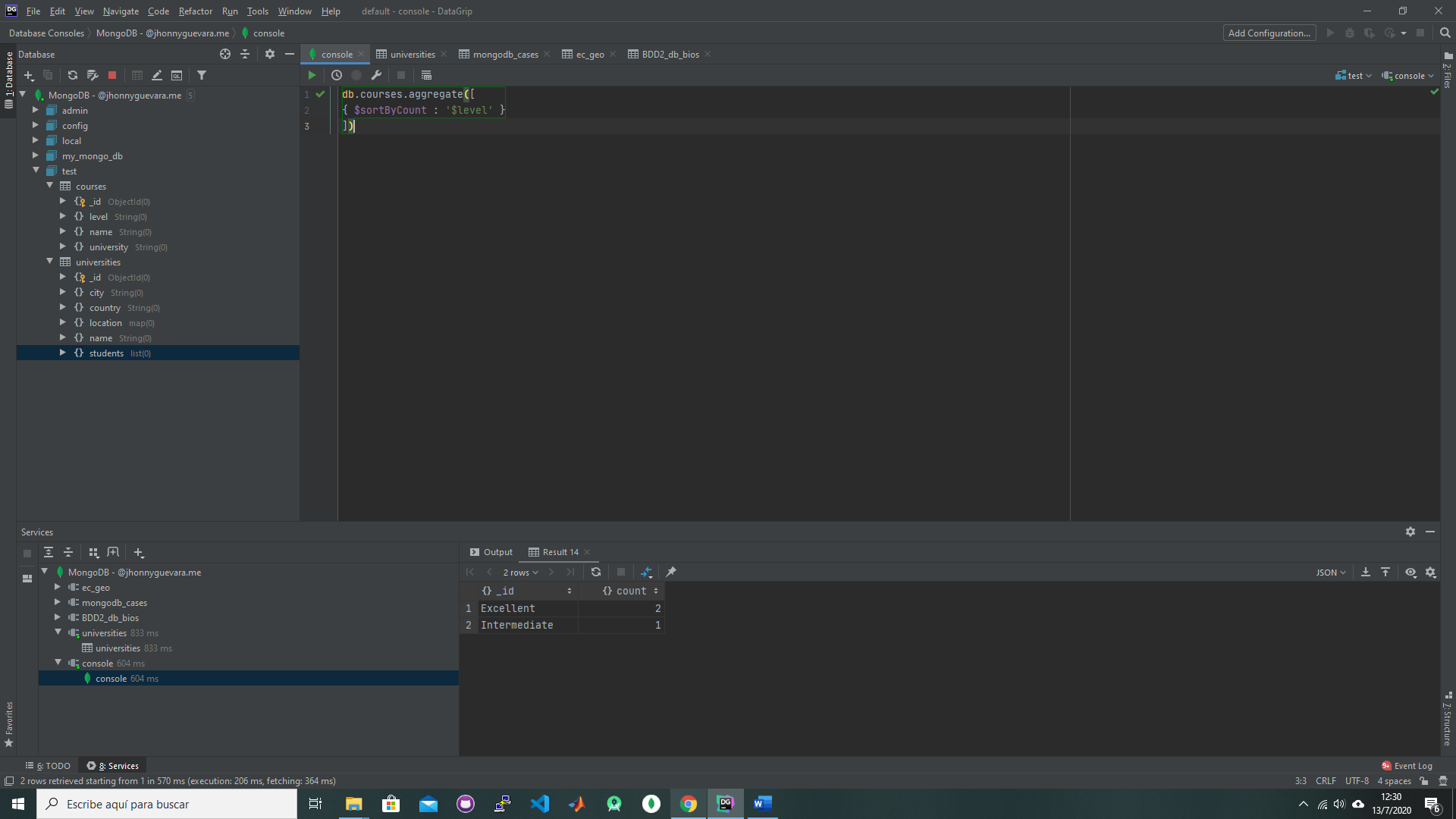
**Usando $lookup(), podemos consultar y combina campos de dos colecciones.**

|  |
| --- |
| db.universities.aggregate([ { $match : { name : 'USAL' } }, { $project : { \_id : 0, name : 1 } }, { $lookup : { from : 'courses', localField : 'name', foreignField : 'university', as : 'courses' } } ]); |



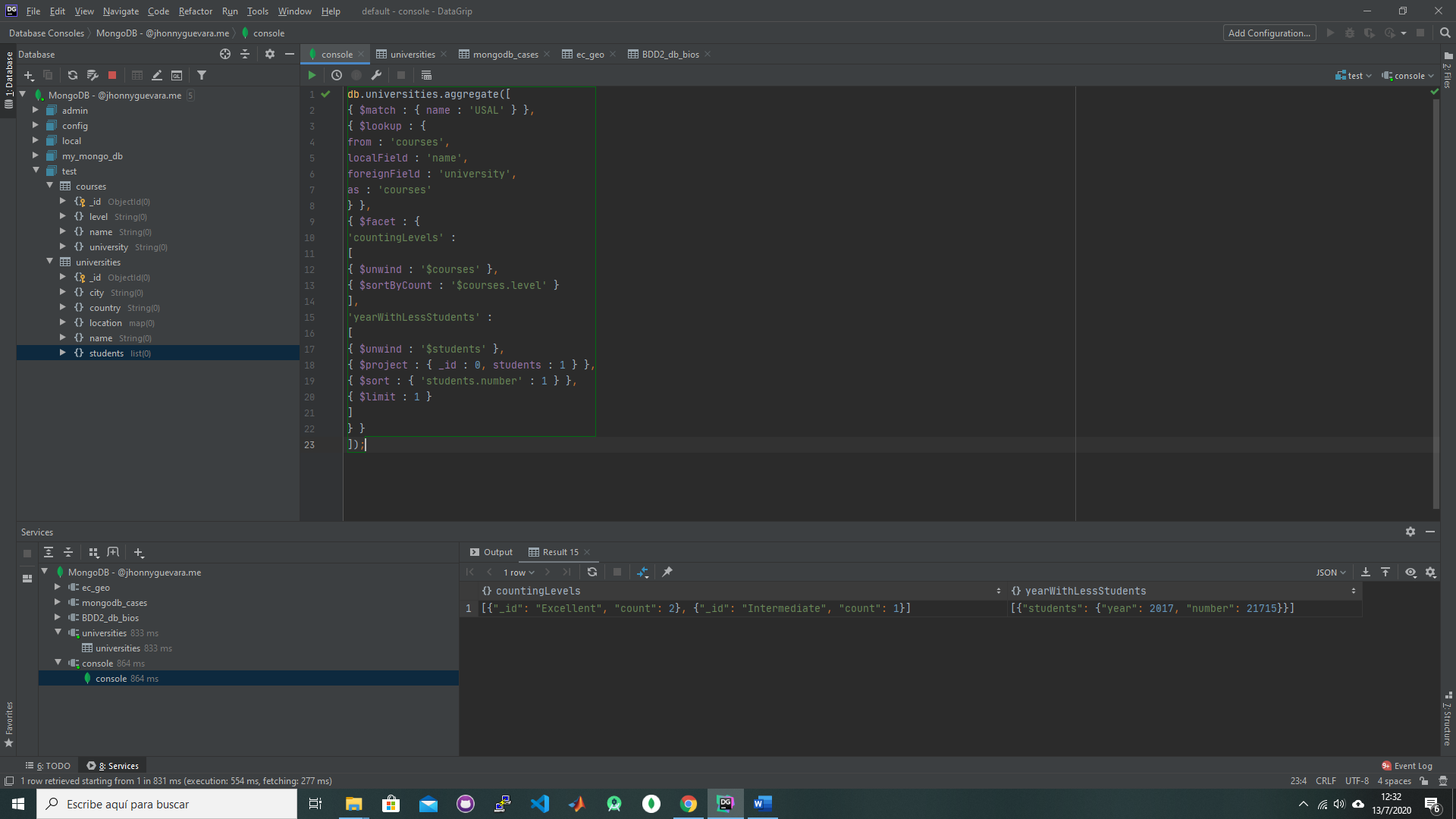
**Este método nos permite crear un atajo para agrupar, contar y luego ordenar en orden descendente el número de valores diferentes en un campo.**

|  |
| --- |
| db.courses.aggregate([ { $sortByCount : '$level' } ]) |



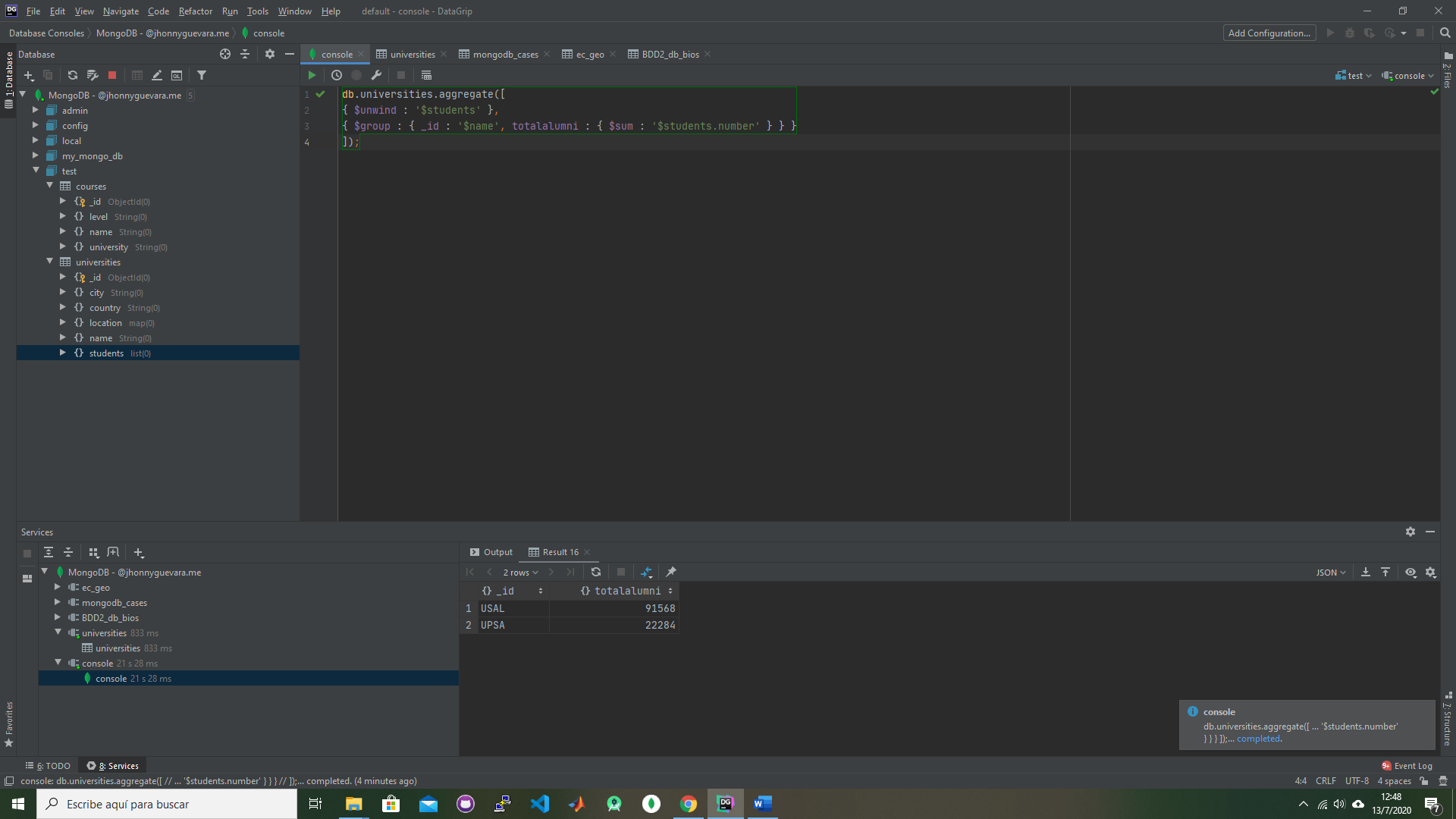
**Se han creado dos informes de nuestra base de datos de cursos universitarios. Contando niveles y año con menos estudiantes.**

|  |
| --- |
| db.universities.aggregate([ { $match : { name : 'USAL' } }, { $lookup : { from : 'courses', localField : 'name', foreignField : 'university', as : 'courses' } }, { $facet : { 'countingLevels' : [ { $unwind : '$courses' }, { $sortByCount : '$courses.level' } ], 'yearWithLessStudents' : [ { $unwind : '$students' }, { $project : { \_id : 0, students : 1 } }, { $sort : { 'students.number' : 1 } }, { $limit : 1 } ] } } ]); |



**Obtenemos el número total de estudiantes que alguna vez pertenecieron a cada una de las universidades**

|  |
| --- |
| db.universities.aggregate([ { $unwind : '$students' }, { $group : { \_id : '$name', totalalumni : { $sum : '$students.number' } } } ]); |



**El**

**Obtenemos el número total de estudiantes que alguna vez pertenecieron a cada una de las universidades y ordena la salida por totalalumni campo en orden descendente**

|  |
| --- |
| db.universities.aggregate([ { $unwind : '$students' }, { $group : { \_id : '$name', totalalumni : { $sum : '$students.number' } } }, { $sort : { totalalumni : -1 } } ]); |

