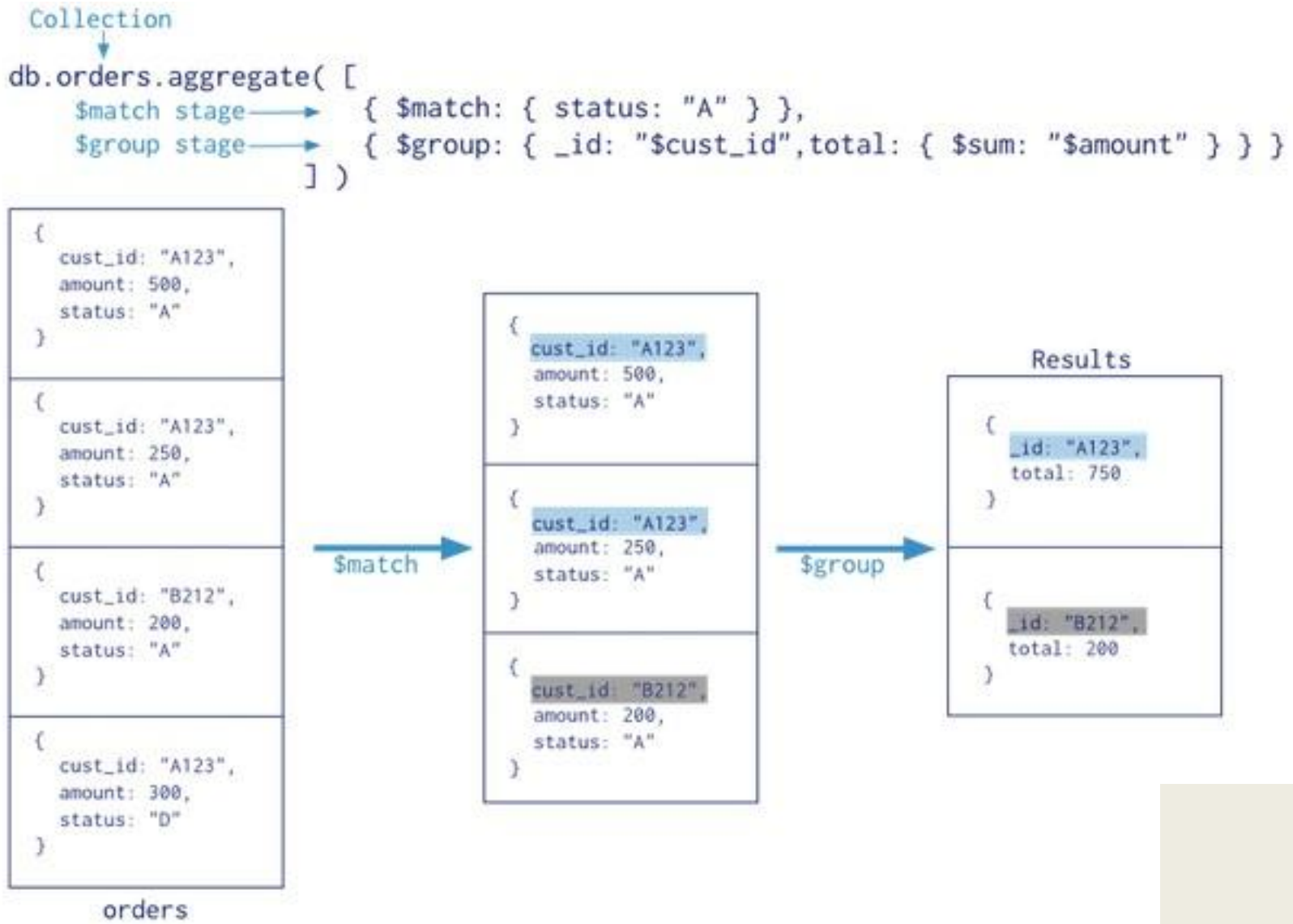
MongoDB – Aggregation Framework - MapReduce

Aggregation Framework

Sintaxis

```
db.collection_name.aggregat
e  ([
    {$pipeline_operator: “expression”},
    {$pipeline_operator:
    {document_expression}},
     …
])
```

Aggregation Framework



Operadores de Agregación

SQL	MongoDB Aggregation Operators
WHERE	$match
GROUP BY	$group
HAVING	$match
SELECT	$project
ORDER BY	$sort
LIMIT	$limit
SUM()	$sum
COUNT()	$sum
Join	$lookup para colecciones distintas ó para arrays de una misma colección $unwind

Aggregation Framework

$match

En esta operación se pueden usar algunos de los operadores ya vistos con el find:

```
$eq, $gt, $gte, $lt, $lte, $ne, $in, $nin, $or, $and, $not,
$exists, $type, $regex, $text, $all, $elemMatch, $size,
$geoWithin.
```

```
$text, de ser usado, el $match debe ir al principio del
aggregate.
```

No puede usarse el **$near**, para esto hay otra operación de la pipeline llamada **$geoNear**.

Aggregation Framework

$project

Se realiza una proyección de los documentos. Pueden seleccionarse atributos como en el segundo documento del find, asi como también renombrarlos y realizar operaciones:

Renombrar un atributo:

Se deberá poner $ antes del atributo y todo entre comillas.

```
{apellido:"$cliente.apellido"}
```

Aggregation Framework

$project

Operaciones matemáticas:

```
{$add:[<expr1>, …, <exprN>]}

{$multiply:[<expr1>, …, <exprN>]}

{$substract:[<expr1>, <expr2>]}

{$divide:[<expr1>, <expr2>]}
```

Ej:

```
total:{$substract:[{$add:[“$salario”, “$bono”]},
2000]}
```

Aggregation Framework

$project

Operaciones con strings:

```
{$concat:[<expr1>, …, <exprN>]}

{$toLower:<expr>}

{$toUpper:<expr>}

{$substr:[<expr>, <offset>,
<len>]}
```

Aggregation Framework

$project

Operaciones con fechas:

```
$year, $month, $week, $dayOfMonth, $dayOfWeek, $dayOfYear,
$hour,
$minute
```

Ej:

```
{año:{$substract:[{$year: new Date()},{$year:
"$fechaInicial"}]}}
```

Aggregation Framework

$group

Los grupos se arman especificando los atributos en el _id. Los operadores disponibles son los siguientes:

```
$sum, $avg, $max, $min, $first, $last, $push,
$addToSet
```

Aggregation Framework

$lookup (3.2)

Realiza un **left outer join** entre colecciones de una misma base de datos. La **colección from no** puede estar **distribuida**.

```
{ $lookup: {
    from: "<colección a unir>",
    localField: "<atributo de los documentos de entrada>",
    foreignField: "<atributo de los documentos de la colección
    from>",
    as: "<atributo en los documentos salientes (Array)>"
} }
```

Aggregation Framework

$lookup (3.2)

Los documentos de salida serán como los documentos de entrada, agregandole el atributo *as*, que será un array con todos los documentos de la colección *from* que tengan el mismo *foreignField* que el *localField* de los documentos de entrada.

```
{ $lookup: {
    from: "<colección a unir>",
    localField: "<atributo de los documentos de entrada>",
    foreignField: "<atributo de los documentos de la colección from>",
    as: "<atributo en los documentos salientes (Array)>"
} }
```

Aggregation Framework

$unwind

Se le especifica un atributo de los documentos de entrada que contenga un array. Se creará un documento de salida por cada elemento del array, donde se mantendrán iguales todos los atributos, salvo el campo del array que tendrá sólo un elemento del array.

entrada *salida*

```
{ atr1:
  "A",
  atr2: 5,
  item: [
    {num: 1, cant:
    100},
    {num: 2, cant: 50},
}   {num: 3, cant: 70}
  ]
```


```
{atr1: "A", atr2: 5, item: {num: 1, cant: 100}
}
{atr1: "A", atr2: 5, item: {num: 2, cant: 50} }
{atr1: "A", atr2: 5, item: {num: 3, cant: 70} }
```

Algunos Ejemplos

BD RELACIONAL

```
CREATE TABLE Ordenes
{ nroOrden INT PRIMARY
   KEY,  fechaEmision    DATE,
   fechaVencimiento DATE,
   condPago    VARCHAR2(60),
   cliente  VARCHAR2(60),
   region  VARCHAR2(60),
   totalOrden INTEGER
  );

CREATE TABLE
Ordenes_items  nroOrden
INT,
nroItem   SMALLINT,
producto VARCHAR(60),
cantidad
SMALLINT
}
```

BD DOCUMENTAL

Colección Ordenes

```
{
        "_id" : ObjectId("536183a934600053a7b6bc67"),
        "nroOrden" : 1466,
        "fechaEmision" : ISODate("2014-04-29T00:00:00Z"),
        "fechaVencimiento" : ISODate("2014-05-30T00:00:00Z"),
        "condPago"                    :
        "CONTADO",    "cliente"   :
        "GALINDO  S.A.",   "cuit"  :
        3040488484,
        "region" : "CABA",
        "totalOrden" : 1500,
        "item" : [ {" : [ "producto" : "mesa 2 x 1 m", "cantidad" : 1
                },
                {      "producto" : "sillas Z322", "cantidad" : 4 }
}                     ]
```

Algunos Ejemplos – Aggregate Framework

Sentencias SQL	MongoDB Aggregation Framework

SELECT COUNT(*) AS cantidad
FROM ordenes
	db.ordenes.aggregate([
	{ $group:
	{
	_id: null,
	cantidad: { $sum: 1 }
	}
	}
	])

Algunos Ejemplos – Aggregate Framework

Sentencias SQL	MongoDB Aggregation Framework

SELECT SUM(totalOrden) AS total
FROM ordenes
	db.ordenes.aggregate([
	{
	$group: {
	_id: null,
	total: { $sum: "$totalOrden" }
	}
	}
	])

Algunos Ejemplos – Aggregate Framework

Sentencias SQL	MongoDB Aggregation Framework

SELECT condPago,
SUM(totalOrden) AS total
FROM ordenes
GROUP BY condPago
	db.ordenes.aggregate([
	{
	$group: {
	_id: "$condPago",
	total: { $sum: "$totalOrden" }
	}
	}
	])

Algunos Ejemplos – Aggregate Framework

Sentencias SQL	MongoDB Aggregation Framework

SELECT condPago,
SUM(totalOrden) AS total
FROM ordenes
GROUP BY
condPago ORDER
BY total
	db.ordenes.aggregate([
	{
	$group: {
	_id: "$condPago",
	total: { $sum: "$totalOrden" }
	}
	},
	{ $sort: { total: 1 } }
	])

Algunos Ejemplos – Aggregate Framework

Sentencias SQL	MongoDB Aggregation Framework

SELECT cliente,
condPago,
SUM(totalOrden) AS total
FROM ordenes
GROUP BY
cliente,
condPago
	db.ordenes.aggregate([
	{
	$group: {
	_id: {
	cliente: "$cliente",
	condPago: "$condPago"
	},
	total: { $sum: "$totalOrden" }
	}
	}])

Algunos Ejemplos – Aggregate Framework

Sentencias SQL	MongoDB Aggregation Framework
SELECT cliente, SUM(totalOrden) as total FROM ordenes WHERE condPago = 'CONTADO' GROUP BY cliente	db.ordenes.aggregate([{ $match: { condPago: 'CONTADO' } }, { $group: {_id: "$cliente", total: { $sum: "$totalOrden" } } }])

Algunos Ejemplos – Aggregate Framework

Sentencias SQL	MongoDB Aggregation Framework

SELECT cliente,
count(*)
FROM ordenes
GROUP BY cliente
HAVING count(*) > 3
	db.ordenes.aggregate([
	{
	$group: {
	_id: "$cliente",
	count: { $sum: 1 }}
	},
	{ $match: { count: { $gt: 3 } } }
	])

Algunos Ejemplos – Aggregate Framework

Sentencias SQL	MongoDB Aggregation Framework

SELECT cliente,
fechaEmision,
SUM(totalOrden) AS total
FROM ordenes
GROUP BY cliente,
fechaEmision
HAVING SUM(totalOrden)
> 10000
	db.ordenes.aggregate([
	{
	$group: { _id:
	{ cust_id: "$cliente",
	ord_date: "$fechaEmision"
	},
	total: { $sum: "$totalOrden" }
	}
	},
	{ $match: { total: { $gt: 10000 } } }
	])

Algunos Ejemplos

Sentencias SQL	MongoDB Aggregation Framework

SELECT cliente,
SUM(totalOrden) as total
FROM ordenes
WHERE condPago = 'CONTADO'
GROUP BY cliente
HAVING SUM(totalOrden)
> 50000
	db.ordenes.aggregate([
	{ $match: { condPago: 'CONTADO' } },
	{
	$group: {
	_id: "$cliente",
	total: { $sum: "$totalOrden" }
	}
	},
	{ $match: { total: { $gt: 50000 } } }
	])

Algunos Ejemplos – Aggregate Framework

Sentencias SQL	MongoDB Aggregation Framework

SELECT COUNT(*)
FROM (SELECT cliente,
fechaEmision
FROM ordenes
GROUP BY cliente,
fechaEmision
) as
TablaDerivada
	db.ordenes.aggregate([
	{
	$group: {
	_id: {
	cust_id: "$cliente",
	ord_date:
	"$fechaEmision"
	}
	}
	},
	{
	$group: {
	_id: null,
	count: { $sum: 1 }
	}
	}
	])

Algunos Ejemplos – Aggregate Framework

Sentencias SQL	MongoDB Aggregation Framework

SELECT *
FROM ordenes
LEFT JOIN clientes
ON
ordenes.cliente.cuit=clientes.cuit
	db.ordenes.aggregate([
	{
	$lookup: { from: "clientes",
	localField: "cuit",
	foreignField: "cuit",
	as: "datosCliente" }
	}
	]).pretty()

$lookup: esta presente a partir de la version 3.2 y realiza un LEFT JOIN de las colecciones

MapReduce

MapReduce es un modelo de programación y su implementación asociada para procesar y generar grandes sets de datos.

Modelo de programación

- **Tiene abstracciones para expresar cálculos simples**

Librería

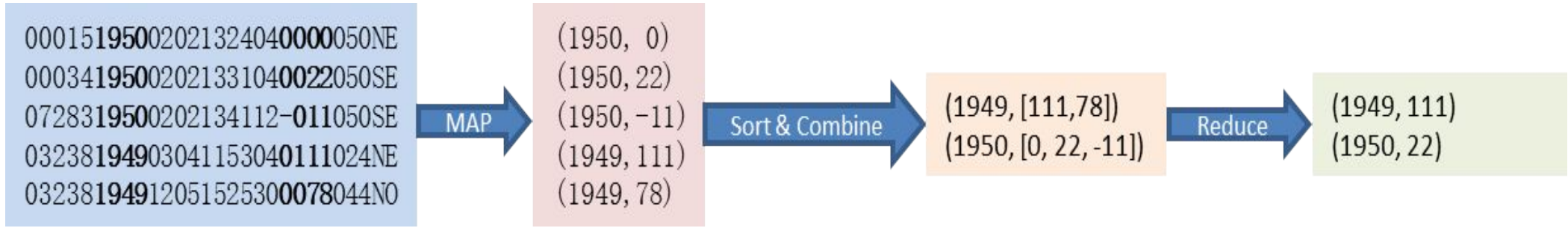
- **Se ocupa del trabajo pesado: Paralelismo, Fault Tolerance, Data Distribution y Load Balancing**

MapReduce

- ***MapReduce es en realidad un patrón basado en una forma de Scatter- Gather que fue luego popularizado por Google a través de su Google's MapReduce Framework.***

- ***Actualmente, la implementación más usada es parte del proyecto Hadoop, aunque varias bases de datos NoSQL (MongoDB) lo implementan internamente.***

MapReduce Ejemplo



Función MAP (Entrada Texto linea,

Retornar Entero año,

Retornar Entero temperatura)

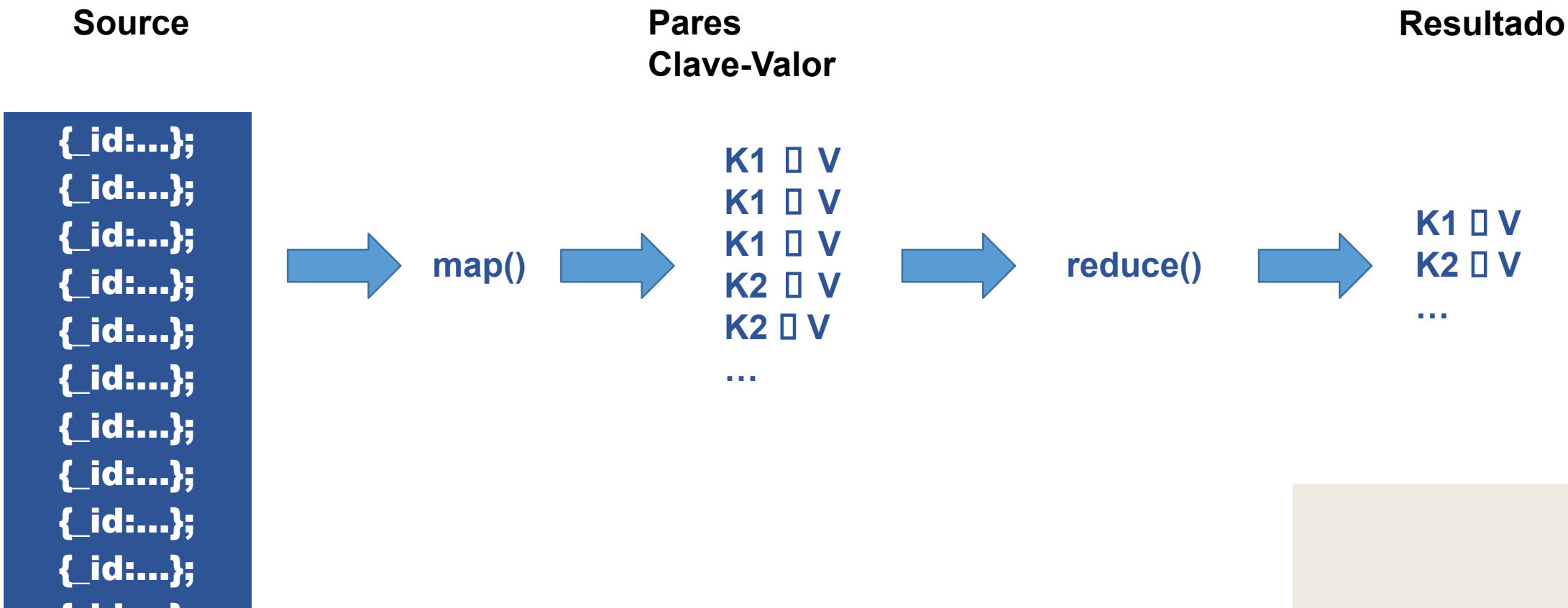
Función REDUCE (Entrada Texto Clave,

Entrada Lista Valor,

Retornar Entero
Año,
Retornar Entero temperaturaMax)

MapReduce

Para realizar operaciones de MapReduce, MongoDB provee el comando **mapReduce**.



MapReduce

Sintaxis

```
db.runCommand(
    {       mapReduce:
        <collection>,   map:
        <function>,
        reduce:
        <function>,   out:
        <output>,   sort:
        <document>,
        limit: <number>,
        finalize:
        <function>,   scope:
        <document>,   jsMode:
        <boolean>,   verbose:
        <boolean>
    }
)
```

```
db.coll_name.mapReduce (
                    map_function,
                    reduce_function
                    ,
                    { out: <collection>,
                      sort: <document>,
                      limit: <number>,
                      finalize:
                      <function>,   scope:
                      <document>,   jsMode:
                      <boolean>,   verbose:
)                     <boolean>
                    }
```

Ejemplo de MapReduce en MongoDB

MAP

```
map = function(){
  this.item.forEach
  (
       function(item){
          emit(item.producto,
          1);
       }
  );
```

```
reduce = function( key,        }
       values){  var total = 0
       for (var i=0; i<values.length; i++
              )  total += values[i];
       return total;
}
```

REDUCE

Ejecución de mapReduce

```
db.ordenes.mapReduce( map ,reduce ,{out:{inline:1}}
```

Ejemplo de MapReduce en MongoDB

Ejecución de mapReduce

```
db.ordenes.mapReduce( map ,reduce ,{out:{inline:1}}
)
```

```
{
    "results" :
        [
        {
            "_id" : "mesa 1 x
            1 m",  "value" : {
                "count" : 1
            }
        },
        {
            "_id" : "mesa 2 x
            1 m",  "value" : {
                "count" : 11
            }
        },
    ..............
    ..............  {
                "_id" : "sillas
                Z324",  "value"
                : {
                    "count" : 3
            }    }
        ]
        ;"timeMillis" :
        218,
        "counts" : {
            "input" :
            22,
            "emit" :
            44,
            "reduce" :
    }       7,
            "output" :
```

Ejemplo de MapReduce en MongoDB

Ejecución de mapReduce – Validación con Aggregation Framework

```
db.ordenes.mapReduce( map ,reduce ,{out:{inline:1}} )
```

```
"timeMillis" : 218,
"counts" : {
        "input" : 22,
        "emit" : 44,
        "reduce" : 7,
        "output" : 8
},
"ok" : 1,
}
```

Input: db.ordenes.count()

22

output:

```
db.ordenes.aggregate(
[  {$project : { item: 1 , _id :0 }},
   { $unwind: "$item" },
   { $group: { _id: "$item.producto"} },
]) { $group: { _id: null, count : { $sum: 1 } }
   },
```

{ "_id" : null, "count" : 8

emit:

```
db.ordenes.aggregate( [
{$project : { item: 1 , _id :0 }},
{ $unwind: "$item" },
{ $group: { _id: null, count: { $sum: 1 } }
}
 ])
```

{ "_id" : null, "count" : 44 }