Métricas

En este apartado veremos cómo obtener información sobre el rendimiento de las consultas que realizamos sobre MongoDB.

Para obtener información sobre cómo se ha realizado una consulta podemos usar el método `explain()`.

```
db.alum.explain().find({nombre: 'Juan'})
```

El método `explain()`

El método `explain()` nos permite obtener información sobre cómo se ha realizado una consulta en forma de un documento JSON. Este método se puede aplicar a cualquier *consulta* (`explain` es un método de `cursor`) que se realice sobre una colección.

`explain` admite un único argumento opcional que consiste en una cadena con la que podremos indicarle lo detallado que queremos que sea el documento que nos devolverá. Este argumento puede tomar tres valores:

- `"queryPlanner"`: es el valor por defecto y hará que `explain` nos devuelva información sobre el resultado del **optimizador de consultas** que ha ejecutando mongo.
- `"executionStats"`: esta opción hará que se ejecute el optimizador de la consulta, se elija un plan y se lleve la consulta a término. Devolverá las estadísticas sobre la ejecución de dicho plan.
- `"allPlansExecution"`: en este caso se hace lo mismo que en el caso anterior pero además se devuelve información sobre los otros planes candidatos que fueron rechazados por el optimizador de consultas.

Ejemplos de `explain`

Hay dos formas de invocar `explain`

```
db.alumnos.explain().find({nombre: 'Juan'})
```

que permitirá encadenar modificadores de la consulta y

```
db.alumnos.find({nombre: 'Juan'}).explain()
```

que devolverá un cursor con el documento que contiene la información sobre la consulta.

A veces nos interesará única y exclusivamente la información sobre el rendimiento de la consulta.

```
db.alumnos.explain('executionStats').find({nombre: 'Juan'})
```

Un ejemplo del documento resultado es el siguiente:

```
{
  explainVersion: '2',
  queryPlanner: {
```

```
    namespace: 'test.chorradas',
    indexFilterSet: false,
    parsedQuery: {},
    queryHash: 'E475932B',
    planCacheKey: '31A214E9',
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    winningPlan: {
      queryPlan: {
        stage: 'COLLSCAN',
        planNodeId: 1,
        filter: {},
        direction: 'forward'
      },
      slotBasedPlan: {
        slots: '$$RESULT=s4 env: { s1 =
TimeZoneDatabase(Asia/Kamchatka...America/Recife) (timeZoneDB), s3 =
1704658291453 (NOW), s2 = Nothing (SEARCH_META) }',
        stages: '[1] scan s4 s5 none none none none lowPriority [] @"f8c29114-
6987-400a-bc92-82f7bb00a913" true false '
      }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 11,
    executionTimeMillis: 3,
    totalKeysExamined: 0,
    totalDocsExamined: 11,
    executionStages: {
      stage: 'scan',
      planNodeId: 1,
      nReturned: 11,
      executionTimeMillisEstimate: 0,
      opens: 1,
      closes: 1,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      numReads: 11,
      recordSlot: 4,
      recordIdSlot: 5,
      fields: [],
      outputSlots: []
    }
  },
  command: { find: 'chorradas', filter: {}, '$db': 'test' },
  serverInfo: {
    host: 'mongodbserver',
    port: 27017,
    version: '7.0.4',
    gitVersion: '38f3e37057a43d2e9f41a39142681a76062d582e'
  },
  serverParameters: {
```

```
    internalQueryFacetBufferSizeBytes: 104857600,
    internalQueryFacetMaxOutputDocSizeBytes: 104857600,
    internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
    internalDocumentSourceGroupMaxMemoryBytes: 104857600,
    internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
    internalQueryProhibitBlockingMergeOnMongoS: 0,
    internalQueryMaxAddToSetBytes: 104857600,
    internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,
    internalQueryFrameworkControl: 'trySbeEngine'
  },
  ok: 1
}
```