Universidade do Porto

Faculdade de Engenharia

FEUP

NoSQL Assignment

Version 3 – Cultural facilities

Tecnologias de Bases de Dados
4º ano do Mestrado Integrado em Engenharia Informática e Computação

Elementos do grupo F:

Catarina Ramos - up201406219 - up201406219@fe.up.pt
Inês Gomes - up201405778 - up201405778@fe.up.pt
Mário Fernandes - up201201705 - up201201705@fe.up.pt

5 de Junho de 2018

Questão 1

"Design a Mongo document model for the Cultural Facilities example, explaining the decisions made. The information available under this model should be equivalent to the information in the relational database."

Em geral, em MongoDB, documentos embebidos oferecem melhor performance para operações de leitura bem como a possibilidade de retornar os dados numa só operação. Também é possível fazer updates de dados relacionados numa só operação de escrita atómica.

Com esta definição em mente, decidimos criar uma só *collection* que abrange todas as classes do modelo original, embebidas em subdocumentos. Esta prática, tem a desvantagem de originar muita duplicação. No entanto, visto que não existem *joins* em MongoDB, tem a vantagem de evitar a criação de duas ou mais queries para fazer uma interrogação à base de dados. Esta *collection* terá a estrutura apresentada no seguinte exemplo:


```
{
   "_id": 1215,
   "designation": "Sousel",
   "district": {
     "cod": 12,
     "designation": "Portalegre",
     "region": {
       "cod": 4,
       "designation": "Alentejo",
       "nut1": "Continente"
     }
   },
   "region": {
     "cod": 4,
     "designation": "Alentejo",
     "nut1": "Continente"
   },
   "facilities": [
     {
       "id": 968,
       "name": "PRAÇA DE TOIROS PEDRO LOUCEIRO-SOUSEL",
       "capacity": 1000,
       "roomtype": "Praça de touros",
       "address": "SOUSEL",
       "activities": [
         "tauromaquia"
       ]
     },
```

```
    {
      "id": 969,
      "name": "AUDITÓRIO DA BIBLIOTECA MUNICIPAL DE SOUSEL",
      "capacity": 110,
      "roomtype": "Auditório",
      "address": "SOUSEL",
      "activities": [
        "cinema"
      ]
    }
  ]
}
```

Cada objeto é um município, ou seja, existem 308 documentos com os sub-documentos *district*, *region* e facility embebidos. Cada município, distrito e região são caracterizados por um código e uma designação. Na região ainda acresce o nut1. Como já foi mencionado na enunciado, o distrito pode ou não ter uma região associada (depende se todos os seus municípios pertencem à mesma região). Cada município terá um conjunto de *facilities* (locais para a realização de espetáculos) que correspondem a esse município. Cada facility é caracterizada por um conjunto de atividades para a qual essa facility pode ser usada bem como o seu tipo (roomtype) o seu id, nome, capacidade e endereço.

Este modelo tem duplicação nos distritos, regiões, atividades e roomtype. No caso dos municípios, distritos e regiões (assim como as relações entre eles) sabemos que são fixos ao longo do tempo, ou seja, nunca vão crescer na BD (o seu número não aumenta). Também sabemos que a duplicação é só de duas a cinco *strings* para cada município. As *facilities* são documentos que aumentarão na BD mas não com grande frequência. No entanto, acaba por ser um conjunto de objetos com um tamanho considerável quando pensamos em todas as salas de espetáculo do país. No entanto, como cada município tem um conjunto de *facilities*, não existe duplicação das mesmas, o que se torna vantajoso considerando o tamanho considerável das *facilities.* Por cada *facility* existem atividades e *roomtypes* que apresentam duplicação. No entanto, achamos que seria mais adequado fazer duplicação do *roomtype* por ser apenas uma *string* e não ser uma enumeração, ou seja, não existem *roomtypes* fixos. No caso das atividades, deixamos de ter a tabelas de *Uses* e *Activities* passando para um array das atividades associadas a cada *facility*. Embora cada facility tenha um novo array de tamanho entre 1 e 6 (número máximo de atividades), preenchidos com strings repetidas (as 6 atividades) esta duplicação é vantajosa para a leitura das facilities e não apresenta um grande aumento do tamanho da BD ou overhead das queries.

A grande vantagem do MongoDB em relação a uma base de dados relacional, é a sua flexibilidade. No entanto, neste caso, queremos garantir que, pelo menos, os dados que estão definidos no esquema relacional estejam presentes e com os tipos certos. Para tal, definimos o *schema* a seguir apresentado:

```
db.createCollection("facility", {
   validator: {
       "_id": { $type: "double", $exists: true, $gt: 0 },
       "designation": { $type: "string", $exists: true },
       "district.cod": { $type: "double", $exists: true },
       "district.designation": { $type: "string", $exists: true},
       "district.region.cod": { $type: "double" },
       "district.region.designation": { $type: "string" },
       "district.region.nut1" : { $in: [ "Continente", "Açores", "Madeira" ] },
       "region.cod": { $type: "double", $exists: true},
       "region.designation": { $type: "string", $exists: true},
       "region.nut1" : { $in: [ "Continente", "Açores", "Madeira" ],  $exists:
true},
       "facilities.id" : { $type: "double", $exists: true, $gt: 0 },
       "facilities.name" : { $type: "string", $exists: true, $gt: 0 },
       "facilities.capacity" : { $type: "double", $exists: true, $gt: 0 },
       "facilities.roomtype" : { $type: "string", $exists: true, $gt: 0 },
       "facilities.address" : { $type: "string", $exists: true, $gt: 0 },
       "facilities.activities": [ { $in: [ "cinema", "circo", "dança", "música",
"tauromaquia", "teatro" ]} ],
   },
   validationAction: "warn"
})
```

Questão 2

"Migrate the data from the Oracle user GTD8 into the NoSQL database. One possibility is to write a PL/SQL package able to extract the data using an appropriate SQL query and to produce the appropriate Mongo method calls to populate the NoSQL database designed in 1). Other methods may be used. Document the procedure in the report, anyway."

Para esta questão o grupo decidiu usar PL/SQL para exportar os dados da base de dados GTD8. Começamos por criar um xsl com o formato desejado para cada coleção, e fazemos a sua visualização em web. A seguir copiamos os dados para um ficheiro .json e executamos o comando *mongoimport* na linha de comandos para importar esse json para o mongo.

1. Criação dos procedures

```
CREATE OR REPLACE PACKAGE export_facilities AS
PROCEDURE export_database;
END export_facilities;

CREATE OR REPLACE PACKAGE BODY export_facilities
AS
       PROCEDURE export_database
       IS
       v_sqlselect  VARCHAR2(4000);
       v_queryctx   DBMS_XMLQuery.ctxType;
       v_clob_par   CLOB;
       xsl_transformXMLTYPE;
       re XMLTYPE;
       xsl_result XMLTYPE;
       v_offset number default 1;
       v_chunk_size number := 10000;
       BEGIN
       v_sqlselect :='
              select mu.COD, mu.designation, CURSOR(
                     select d.cod, d.designation, CURSOR(
                            select r2.cod, r2.designation, r2.nut1
                            from GTD8.regions r2
                            where r2.cod = d.region) as REGION
                     from GTD8.districts d where d.cod = mu.district) AS
DISTRICT, CURSOR(
                            select r.cod, r.designation, r.nut1
                            from GTD8.regions r
                            where r.cod = mu.region) AS REGION, CURSOR(
                                   select f.id, f.name, f.capacity, CURSOR(
                                          select rt.description
                                          from GTD8.roomtypes rt
                                          where f.roomtype = rt.roomtype) AS
```

```
ROOMTYPE, f.address, CURSOR(
                                                select a.activity
                                                from GTD8.uses u JOIN
GTD8.activities a ON(u.ref = a.ref )
                                                where f.id = u.id ) AS ACTIVITIES
                                    from GTD8.facilities f  where mu.cod =
f.municipality) AS FACILITIES
               from GTD8.municipalities mu';

       v_queryctx := DBMS_XMLQuery.newContext(v_sqlselect);
       DBMS_XMLQuery.setEncodingTag(v_queryctx, 'ISO-8859-1');
       v_clob_par := DBMS_XMLQuery.getXML(v_queryctx);
       DBMS_XMLQuery.closeContext(v_queryctx);

       xsl_transform := XMLTYPE.CREATEXML('
              <?xml version="1.0" encoding="utf-8"?>
              <xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
              <xsl:template match="/ROWSET">[
              <xsl:for-each select="ROW">
              {
             "_id": <xsl:value-of select="COD" />,
             "designation": "<xsl:value-of select="DESIGNATION" />",
             "district":{
             "cod": <xsl:value-of select="DISTRICT/DISTRICT_ROW/COD" />,
             "designation": "<xsl:value-of
select="DISTRICT/DISTRICT_ROW/DESIGNATION" />"
            <xsl:if test="DISTRICT/DISTRICT_ROW/REGION/REGION_ROW/COD" >,
             "region":{
                    "cod": <xsl:value-of
select="DISTRICT/DISTRICT_ROW/REGION/REGION_ROW/COD" />,
                    "designation": "<xsl:value-of
select="DISTRICT/DISTRICT_ROW/REGION/REGION_ROW/DESIGNATION" />",
                    "nut1": "<xsl:value-of
select="DISTRICT/DISTRICT_ROW/REGION/REGION_ROW/NUT1" />"
             }
             </xsl:if>
             },
             "region":{
             "cod": <xsl:value-of select="REGION/REGION_ROW/COD" />,
             "designation": "<xsl:value-of
select="REGION/REGION_ROW/DESIGNATION" />",
             "nut1": "<xsl:value-of select="REGION/REGION_ROW/NUT1" />"
             },
             "facilities": [
             <xsl:for-each select="FACILITIES/FACILITIES_ROW">
             {
             "id": <xsl:value-of select="ID" />,
             "name": "<xsl:value-of select="NAME" />",
             "capacity": <xsl:value-of select="CAPACITY" />,
```

```
            "roomtype": "<xsl:value-of
select="ROOMTYPE/ROOMTYPE_ROW/DESCRIPTION" />",
            "address": "<xsl:value-of select="ADDRESS" />"
            "activities": [
                  <xsl:for-each select="ACTIVITIES/ACTIVITIES_ROW">
                  "<xsl:value-of select="ACTIVITY" />"
                  <xsl:if test="./following-sibling::ACTIVITIES_ROW">,
</xsl:if>
                  </xsl:for-each>
            ]
            }
            <xsl:if test="./following-sibling::FACILITIES_ROW">,</xsl:if>
            </xsl:for-each>
            ]
            }
            <xsl:if test="./following-sibling::ROW">,</xsl:if>
             </xsl:for-each>
             ]
             </xsl:template>
             </xsl:stylesheet>');

       re := XMLTYPE.createXML(v_clob_par);
       xsl_result := re.transform(xsl => xsl_transform);
       v_clob_par := xsl_result.getClobVal();

      loop
             exit when v_offset > dbms_lob.getlength(v_clob_par);
             htp.prn(dbms_lob.substr(v_clob_par, v_chunk_size, v_offset));
             v_offset := v_offset +  v_chunk_size;
       end loop;
    END export_database;
END export_facilities;
```

2. Criação dos ficheiros .json

Após a execução dos procedures, acedendo a cada um dos resultados (http://oraalu.fe.up.pt/pls/export_facilities.export_database) é possível extrair o json resultante para o ficheiro t*bda_db.json*.

3. Import

Por fim, para colocar os ficheiros json no mongo executamos o comando seguinte para a base de dados e utilizador escolhidos.

mongoimport --host vdbase.inesctec.pt:27017 --db <dbname> --username <username> --password <password> --collection facility --file tbda_db.json --jsonArray --authenticationDatabase admin

Questão 3

Prepare Mongo queries for the following questions:

a) Which are the facilities where the room type description contains ‘touros’ and have ‘teatro’ as one of their activities? Show the id, name, description and activity.

i) Query

```
db.facility.aggregate([
{$unwind: "$facilities"},
{$match: {
  "facilities.roomtype": {$regex: "touros"},
  "facilities.activities": "teatro"}
},
{$group : {"_id":null, facs: {$addToSet: "$facilities"}}},
{$project: {
   "_id":0,
   "facs.id":1,
   "facs.name":1,
   "facs.roomtype":1,
   "facs.activities":1}},
{$unwind:"$facs"}
])
```

ii) Resultados

```
/* 1 */
{
       "facs" : {
       "id" : 916,
       "name" : "COLISEU JOSÉ RONDÃO DE ALMEIDA-EX PRAÇA DE TOIROS",
       "roomtype" : "Praça de touros multiusos",
       "activities" : [
              "dança",
              "música",
              "tauromaquia",
              "teatro"
       ]
       }
}

/* 2 */
{
       "facs" : {
```

```
        "id" : 957,
        "name" : "COLISEU DE REDONDO - EX PRAÇA DE TOIROS",
        "roomtype" : "Praça de touros multiusos",
        "activities" : [
                "dança",
                "música",
                "tauromaquia",
                "teatro"
        ]
        }
}

/* 3 */
{
        "facs" : {
        "id" : 940,
        "name" : "ARENA DE ÉVORA - EX PRAÇA DE TOIROS",
        "roomtype" : "Praça de touros multiusos",
        "activities" : [
                "dança",
                "música",
                "tauromaquia",
                "teatro"
        ]
        }
}
```


iii) Tempo

Time (ms)	18

b) How many facilities with ‘touros’ in the room type description are there in each region?

i) Query

```
db.facility.aggregate([
{$unwind: "$facilities"},
{$match: {"facilities.roomtype": {$regex: "touros"}}},
{$group: {"_id": "$region","facilities" : {$addToSet: "$facilities"}}},
{$project: {"_id":0,"region": "$_id", "number_of_facilities" : {$size:
"$facilities"}}}
])
```

ii) Resultados


```
/* 1 */
{
       "region" : {
       "cod" : 3,
       "designation" : "Lisboa",
       "nut1" : "Continente"
       },
       "number_of_facilities" : 6
}

/* 2 */
{
       "region" : {
       "cod" : 4,
       "designation" : "Alentejo",
       "nut1" : "Continente"
       },
       "number_of_facilities" : 43
}

/* 3 */
{
       "region" : {
       "cod" : 1,
       "designation" : "Norte",
       "nut1" : "Continente"
       },
       "number_of_facilities" : 3
}

/* 4 */
{
       "region" : {
       "cod" : 2,
       "designation" : "Centro",
       "nut1" : "Continente"
       },
       "number_of_facilities" : 11
}

/* 5 */
{
       "region" : {
       "cod" : 5,
       "designation" : "Algarve",
       "nut1" : "Continente"
       },
       "number_of_facilities" : 1
```

```
}
```

iii) Tempos

Time (ms)	20

c) How many municipalities do not have any facility with an activity of ‘cinema’?

i) Query

```
db.facility.count({'facilities.activities': {$nin: ["cinema"]}})
```

ii) Resultados

```
100
```

iii) Tempos

Time (ms)	15

d) Which is the municipality with more facilities engaged in each of the six kinds of activities? Show the activity, the municipality name and the corresponding number of facilities.

i) Query

```
db.facility.aggregate([
{$unwind : "$facilities"},
{$unwind : "$facilities.activities"},
{$project : {"_id":1, "designation":1,activity : "$facilities.activities"}},
{$group : {"_id" : {"municipality" : "$designation","activity":"$activity"},
count : {$sum : 1}}},
{$group : {"_id":"$_id.activity", max : {$max :
{"n_facilities":"$count","municipality" : "$_id.municipality"}}}},
{$project : {"_id":0,"activity":"$_id", "max_n_facilities":"$max.n_facilities",
"municipality":"$max.municipality"}}
])
```

ii) Resultados


```
/* 1 */
{
	"activity" : "circo",
	"max_n_facilities" : 2.0,
	"municipality" : "Lisboa"
}

/* 2 */
{
	"activity" : "dança",
	"max_n_facilities" : 47.0,
	"municipality" : "Lisboa"
}

/* 3 */
{
	"activity" : "música",
	"max_n_facilities" : 77.0,
	"municipality" : "Lisboa"
}

/* 4 */
{
	"activity" : "cinema",
	"max_n_facilities" : 96.0,
	"municipality" : "Lisboa"
}

/* 5 */
{
	"activity" : "tauromaquia",
	"max_n_facilities" : 4.0,
	"municipality" : "Moura"
}

/* 6 */
{
	"activity" : "teatro",
	"max_n_facilities" : 66.0,
	"municipality" : "Lisboa"
}
```


iii) Tempos

Time (ms)	20

e) Which are the codes and designations of the districts with facilities in all the municipalities?

i) Query

```
function districts_with_total_facilities(){

//Districts with municipalities without facilities
var missing = db.facility.aggregate([
	{$match : {facilities : []}},
	{$group : {_id:"$district.cod"}},
	{$group : {_id:null, "districts": {$addToSet : "$_id"}}}
])._batch[0].districts;

//Districts with facilities on all municipalities
var complete = db.facility.aggregate([
	{$group : {_id:"$district"}},
	{$match : {"_id.cod" : {$nin : missing}}},
	{$project : {"_id":0, "cod":
"$_id.cod","designation":"$_id.designation"}}
])._batch;

	for(var i = 0; i < complete.length; i++){
		print(complete[i]);
	}
};

districts_with_total_facilities();
```

ii) Resultados

```
/* 1 */
{
	"cod" : 7,
	"designation" : "Évora"
}

/* 2 */
{
	"cod" : 11,
	"designation" : "Lisboa"
}

/* 3 */
{
```

```
        "cod" : 12,
        "designation" : "Portalegre"
}

/* 4 */
{
        "cod" : 15,
        "designation" : "Setúbal"
}
```

iii) Tempos

Time (ms)	31

f) Ask the database a query you think is interesting: Quais os municípios que tenham todas as actividades.

i) Query

```
function municipality_with_all_activities(){

//Max number of activities
var max_activities = db.facility.aggregate([
        {$unwind : "$facilities"},
        {$unwind : "$facilities.activities"},
        {$group : {"_id" : "$facilities.activities"}},
        {$count:"number_activities"}
 ])._batch[0].number_activities;

//Number of activities per municipality
var tmp = db.facility.aggregate([
        {$unwind : "$facilities"},
        {$unwind : "$facilities.activities"},
        {$project : {"_id": "$designation" ,
"activity":"$facilities.activities"}},
        {$group : {"_id" : "$_id","activities" : {$addToSet : "$activity"}}},
        {$project : {"_id":0,"designation":"$_id","n_activities" : {$size :
"$activities"}}},
        {$match : {"n_activities":max_activities}}
 ])._batch;

        for(var i = 0; i < tmp.length; i++){
                print(tmp[i]);
        };
};
municipality_with_all_activities();
```

ii) Resultados

```
/* 1 */
{
       "designation" : "Lisboa",
       "n_activities" : 6
}
```

iii) Tempos

Time (ms)	37

Questão 4

Compare the Mongo and the Oracle implementations from the viewpoints of data size, processing time, and query easiness.

Para podermos proceder a uma análise do tempo de processamento e facilidade na pesquisa, tanto na implementação em Mongo como em Oracle, as queries da alínea 3 foram refeitas e executadas em sql. As queries e os tempos de processamento podem ser visualizados abaixo. As respostas são as mesmas das alíneas correspondentes na questão 3. Os tempos apresentados, quer em SQL quer em MongoDB, correspondem à média de 5 medições de execução das queries.

As conclusões formuladas apresentam-se após as seguintes queries.

Query a.

```
SELECT id, name, description, activity
FROM GTD8.FACILITIES f
JOIN GTD8.ROOMTYPES r ON f.ROOMTYPE = r.ROOMTYPE
JOIN (
    SELECT u.id as facility, activity
    FROM GTD8.USES u
    JOIN GTD8.ACTIVITIES a ON u.ref = a.ref
    WHERE activity = 'teatro'
)
ON facility = f.id
WHERE r.DESCRIPTION like '%touros%';
```

Time (ms)	27

Query b.

```
SELECT r.COD, r.DESIGNATION, count(*)
FROM GTD8.FACILITIES f
JOIN GTD8.ROOMTYPES rt ON f.ROOMTYPE = rt.ROOMTYPE
JOIN GTD8.MUNICIPALITIES m ON m.COD = f.MUNICIPALITY
JOIN GTD8.REGIONS r ON m.REGION = r.COD
WHERE rt.DESCRIPTION like '%touros%'
group by r.COD, r.DESIGNATION
order by r.COD;
```

Time (ms)	22

Query c.

```
SELECT count(DISTINCT m.cod)
FROM GTD8.MUNICIPALITIES m
WHERE m.COD NOT IN (
    SELECT distinct(m.COD) as municipality
    FROM GTD8.MUNICIPALITIES m
    JOIN GTD8.FACILITIES f ON m.cod = f.MUNICIPALITY
    JOIN GTD8.USES u ON u.ID = f.ID
    JOIN GTD8.ACTIVITIES a ON u.REF = a.REF
    WHERE activity = 'cinema'
);
```

Time (ms)	30

Query d.

```
SELECT top.activity, tmp.municipality, top.JOINED_FACILITIES
FROM
(
    SELECT MAX(joined_facilities) as joined_facilities, activity
    FROM(
        SELECT f.municipality, count(f.id) as JOINED_FACILITIES, a.ACTIVITY
        FROM GTD8.USES u
        JOIN GTD8.ACTIVITIES a ON u.REF = a.REF
        JOIN GTD8.FACILITIES f ON f.ID = u.ID
        GROUP BY f.MUNICIPALITY, a.ACTIVITY
    )
    GROUP BY activity
) top
JOIN
(
    SELECT f.municipality, count(f.id) as JOINED_FACILITIES, a.ACTIVITY
    FROM GTD8.USES u
    JOIN GTD8.ACTIVITIES a ON u.REF = a.REF
    JOIN GTD8.FACILITIES f ON f.ID = u.ID
    GROUP BY f.MUNICIPALITY, a.ACTIVITY
) tmp
ON tmp.activity = top.activity AND tmp.joined_facilities =
top.joined_facilities;
```

Time (ms)	41

Query e.

```
SELECT COD, DESIGNATION
FROM GTD8.DISTRICTS d
WHERE d.COD NOT IN
(
    -- districts where there are municipalities
    -- with 0 facilities
    SELECT m.DISTRICT
    FROM GTD8.MUNICIPALITIES m
    JOIN (
        -- facilities count per municipality
        SELECT m.COD, count(f.ID) as n_facilities
        FROM GTD8.FACILITIES f
        RIGHT JOIN GTD8.MUNICIPALITIES m ON f.MUNICIPALITY = m.cod
        GROUP BY m.COD
    ) tmp
    ON m.COD = tmp.COD
    WHERE tmp.n_facilities = 0
    GROUP BY m.district
);
```

Time (ms)	35

Query f.

```
SELECT *
FROM
(
SELECT f.MUNICIPALITY, count(distinct activity) as count_activity
FROM GTD8.USES u
JOIN GTD8.ACTIVITIES a ON u.ref = a.ref
JOIN GTD8.FACILITIES f ON u.ID = f.ID
GROUP BY f.MUNICIPALITY
)
WHERE count_activity = (
    SELECT count(distinct a.activity)
    FROM GTD8.ACTIVITIES a
);
```

Time (ms)	25

A tabela seguinte resume os tempos medidos para a mesma questão na base de dados relacional (GTD8) e na base de dados MongoDB.

Alínea	Oracle	Mongo DB
a)	27	18
b)	22	20
c)	30	15
d)	41	20
e)	35	31
f)	25	37

Esta tabela mostra que os tempos de execução para em MongoDB são, regra geral, menores do que os dados para o modelo de dados relacional. Estes tempos podem ser explicados pelas características do MongoDB que permitem facilidade de leitura e atomicidade de operações. No entanto, estes tempos não podem ser generalizados dado que os nossos dados são demasiado pequenos para que as flutuações dos tempos se tornem menos significativas e para ver diferenças mais reais.

Em termos de facilidade de queries, podemos concluir que, para quem já está habituado a fazer queries em em SQL, será essa a metodologia mais simples. No entanto a longo prazo, aprofundado os elementos que o MongoDB oferece, teríamos a vantagem de estar a trabalhar diretamente sobre JSON que é mais legível e direto. Por fim ainda temos a vantagem de conseguir criar um modelo de dados em muito menos tempo do que o normal para criar um modelo de dados relacional.

O tamanho da informação a guardar em Mongodb depende de como o modelo é construído. Neste caso, temos a duplicação de informação na coleção ‘municipality’ que, para além da informação relativa a este, também guarda a informação sobre o ‘district’ e a ‘region’. Desta forma, a informação dos municípios, distritos e regiões ocupa mais espaço na implementação em mongo. Por outro lado, no modelo atual, foram efectuadas escolhas de modelação que permitiram reduzir, pelo menos, o número de coleções para guardar a informação. Por exemplo, na colecção ‘facilities’ a informação ‘roomtypes’ já corresponde diretamente à designação do tipo de salas do edifício e não a uma referência para uma entrada numa tabela com a informação sobre as designações de cada tipo de quarto. A representação das ‘activities’ é representada por um array com o nome de cada actividade relativamente a uma ‘facility’. Apesar de haver informação repetida, o número de coleções diminuí e não é necessário guardar a informação das tabelas ‘use’ e ‘activity’.

MongoDB é uma base de dados programada para trabalhar com um grande volume de dados, ou com schemas que precisam de ser facilmente adaptáveis, coisas que não são

verificáveis neste trabalho, pelo que não é suficiente para conseguirmos usufruir das capacidades do MongoDB em plenitude.