



DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA



Departamento de Computación,
Facultad de Ciencias Exactas y Naturales,
Universidad de Buenos Aires

Trabajo Práctico 2

Bases de Datos

Primer Cuatrimestre de 2016

Apellido y Nombre	LU	E-mail
Federico Hosen	825/12	federico.hosen@gmail.com
Martin 'Martó' Caravario	470/12	martin.caravario@gmail.com
Guido Rajngewerc	379/12	guido.raj@gmail.com
Christian Russo	679/10	christian.russo8@gmail.com

2.2. Colecciones

A continuacion mostramos las colecciones que utilizamos para resolver las consultas propuestas en el enunciado

2.2.1. Publicaciones

```
1 {
2   "Id": 65,
3   "TipoPublicacion": "Servicio"
4 }
```

2.2.2. Compra

```
1 {
2   "idCompra": 656897,
3   "idUsuarioComprador": 65468,
4   "Fecha": "2-06-2014",
5   "Cantidad": 2,
6   "Publicacion": {
7     "idPublicacion": 2365,
8     "idUsuario": 123,
9     "Precio": 3655,
10    "PorcentajeVenta": 20 //entre 0 y 100
11  },
12   "CalificacionDelVendedor": 8,
13   "CalificacionDelComprador": 6
14 }
```

2.2.3. Factura

```
1 {
2   "idFactura": 687468,
3   "IdUsuario": 654,
4   "Fecha": "16-9-2015",
5   "TotalAPagar": 5624,
6   "estoyPagando": [
7     {
8       "idPublicacion": 54,
9       "TipoSuscripcion": "Rubi"
10    },
11    {
12      "idPublicacion": 54,
13      "TipoSuscripcion": "Libre"
14    }
15  ]
16 }
```

2.3. Migracion datos SQL a Colecciones

Para migrar los datos de SQL (MySQLWorkwench) al formato de JSON que se necesita en MongoDB lo que hicimos fue exportar los datos directo desde MySQLWorkwench, pero estos se exportan en JSON plano, es decir no quedaba formateado de la forma que nosotros queriamos. Para solucionar este problema realizamos unos algoritmos en python para formatearlos a mano. Estos algoritmos reciben como input el JSON plano y devuelven un JSON formateado para MongoDB

2.3.1. Parser para el JSON de la compra

```
import json
from pprint import pprint

with open(raw_input()) as data_file:
    data = json.load(data_file)
for a in data:
    print "{"
    print '"idCompra": ' , a['idCompra'], ","
    print '"idUsuario": ' , a['idUsuario'],","
    print '"fecha": ' , ''',a['fecha'],'','',","
    print '"cantidad": ' , a['cantidad'],","
    print '"Publicacion": {'
    print '"idPublicacion": ' , a['idPublicacion'],","
    print '"idUsuario": ' , a['idUsuario'],","
    print '"Precio": ' , a['precio'],","
    print '"PorcentajeVenta": ' , a['porcentajeVenta']
    print "}" ,","
    print '"nombre": ' , ''', a['nombre'], ''
    print '}'
```

2.3.2. Parser para el JSON de la Facutra

```
import json
from pprint import pprint

with open('factura_sql.json') as data_file:
    data = json.load(data_file)

c = {}
for a in data:
    c.setdefault(a["idFactura"], []).append(a)

for a in c:
    print '{'
    print '"idFactura": ', c[a][0]['idFactura'], ","
    print '"IdUsuario": ', c[a][0]['idUsuario'], ","
    print '"Fecha": ', ' ', c[a][0]['fecha'], ' ', "
    total = 0
    for x in c[a]:
        total = total + c[a][c[a].index(x)]['totalAPagar']
    print '"TotalAPagar": ', total, ","
    print '"estoyPagando": ['
    i = 1
    for x in c[a]:
        print "{"
        print '"idPublicacion": ', c[a][c[a].index(x)]['idPublicacion'], ","
        print '"TipoSuscripcion": ', ' ', c[a][c[a].index(x)]['nombre'], ' '
        if i == len(c[a]):
            print "}"
        else:
            print "},"
        i = i+1
    print ']'
    print '}'
```

Nota: para la coleccion de publicaciones no tuvimos que hacer ningun algoritmo.

2.4. Consultas SQL

A continuacion listamos las consultas realizadas para conseguir los datos necesarios para nuestras colecciones:

2.4.1. SQL de Publicaciones

```
select p.idPublicacion, t.Nombre
from publicaciones p
inner join TipoDePublicacion t
on t.idTipoPublicacion = p.idTipoPublicacion
```

2.4.2. SQL de Compras

```
select c.idCompra, c.idUsuario, c.fecha, c.cantidad, c.idPublicacion, p.precio, tp.costo, tp.porcentaje
from compra c
join publicacion p
on c.idPublicacion = p.idPublicacion
join tipo_de_publicacion tp
on p.idTipoDePublicacion = tp.idTipoDePublicacion
```

2.4.3. SQL de Facturas

```
select f.idFactura, p.idUsuario, f.fecha, f.totalAPagar, p.idPublicacion, tp.nombre
from factura f
inner join corresponde c
on c.idFactura = f.idFactura
inner join publicacion p
on c.idPublicacion = p.idPublicacion
inner join tipo_de_publicacion tp
on tp.idTipoDePublicacion = p.idTipoDePublicacion
```

2.5. Implementacion MongoDB

Para implementar la base de datos de documentos en mongoDB utilizamos los JSON de las colecciones descriptos anteriormente y los importamos de la siguiente forma:

```
mongoimport --db tp2 --collection compras --drop --file ./compra_mongo.json
mongoimport --db tp2 --collection facturas --drop --file ./factura_mongo.json
mongoimport --db tp2 --collection publicacion --drop --file ./publicacion_mongo.json
```

3. Parte 2 - Map Reduce

3.1. Ejercicio 1

El importe total de ventas por usuario.

3.2. Ejercicio 2

La reputación histórica de cada usuario según la calificación.

3.3. Ejercicio 3

Las operaciones con comisión más alta.

3.4. Ejercicio 4

El monto total facturado por año.

3.5. Ejercicio 5

El monto total facturado por año de las operaciones pertenecientes a usuarios con suscripciones Rubi Oriente.

3.6. Ejercicio 6

El total de publicaciones por tipo de publicación (productos, servicios o mixtas).

4. Parte 3 - Sharding**5. Conclusiones**