Bases de Datos 2022
Recuperatorio II: NoSQL

Sergio Canchi, Cristian Cardellino,
Ramiro Demasi, Juan Cabral

Contexto

La base de datos **restaurantdb** contiene una colección llamada **restaurants** con información de restaurantes de la ciudad de New York.

Para cargar la base de datos **restaurantdb** ejecutar los siguientes comandos:

```
$ tar xzvf retaurantdb.tar.gz
$ mongoimport –host <host> --db restaurant --collection restaurants --drop
--file restaurantdb/restaurants.json
```

Consignas

Las consignas 1 a 3 se deben resolver utilizando las operaciones de consulta y crud básicas de MongoDB (i.e. no se puede utilizar el pipeline de agregación).

1. Listar el nombre (**`name`**) y barrio (**`borough`**) de todos los restaurantes de cocina (**`cuisine`**) tipo "Italian" y que entre sus notas (**`grades`**) tengan al menos una entrada con nota (**`grade`**) "A" y puntaje (**`score`**) mayor o igual a 10. La lista final sólo deberá mostrar 1 entrada por restaurante y deberá estar ordenada de manera alfabética por el barrio primero y el nombre después. Hint: Revisar operadores **`$regex`** y **`$elemMatch`**.
2. Actualizar las panaderías (**`cuisine ~ Bakery`**) y las cafeterías (**`cuisine ~ Coffee`**) agregando un nuevo campo **discounts** que sea un objeto con dos campos: **`day`** y **`amount`**. Si el local se ubica en Manhattan, el día será "**`Monday`**" y el descuento será "**`%10`**". En caso contrario el día será "**`Tuesday`**" y el descuento será "**`5%`**". Hint: Revisar el operador **`$cond`**.
3. Contar la cantidad de restaurantes cuyo **`address.zipcode`** se encuentre entre 10000 y 11000. Tener en cuenta que el valor original es un string y deberá ser convertido. También tener en cuenta que hay casos erróneos que no pueden ser convertidos a número, en cuyo caso el valor será reemplazado por **`0`**. Hint: Revisar el operador **`$convert`**.

Las consignas 4 y 5 se pueden resolver utilizando pipeline de agregación.

4. Por cada tipo de cocina (`cuisine`), contar la cantidad de notas distintas recibidas (`grades.grade`) en el segundo semestre de 2013. Ordenar por tipo de cocina y nota.
5. Data la siguiente tabla de conversión de notas (`grades.grade`):

A	5
B	4
C	3
D	2
*	1

   Donde "*" sería el resto de los casos posibles. Transformar las notas de los restaurantes de acuerdo a la tabla. Luego, calcular la nota promedio, máxima y mínima por tipo de cocina (`cuisine`). El resultado final deberá mostrar la cocina, la nota promedio, la nota máxima y la nota mínima, ordenadas de manera descendente por la nota promedio. Hint: Revisar el operador `$switch`.

Las consignas 6 y 7 son de modelado de datos.

6. Especificar reglas de validación para la colección restaurant utilizando JSON Schema. Tener en cuenta los campos: address (con sus campos anidados), borough, cuisine, grades (con sus campos anidados), name, restaurant_id, y discount (con sus campos anidados). Inferir tipos y otras restricciones que considere adecuadas (incluyendo campos requeridos). Agregar una regla de validación para que el zipcode, aún siendo un string, verifique que el rango esté dentro de lo permitido para New York City (i.e. 10001-11697). Finalmente dejar 2 casos de falla ante el esquema de validación y 1 caso de éxito. Hint: Deberán hacer conversión con $convert en el caso de la regla de validación. Los casos no deben ser triviales (i.e. sólo casos de falla por un error de tipos).
7. Se desean agregar "client reviews", dados por los clientes de los restaurantes. Los reviews cuentan de un título de menos de 50 caracteres, un puntaje entero entre 0 y 5, una reseña de máximo 250 caracteres (que es opcional) y una fecha y un cliente que lo realizó (con información de nombre y correo electrónico del cliente). Cada review está asociado a un restaurante y un mismo restaurante puede tener varios reviews. Asimismo, un cliente puede hacer reviews de varios restaurantes distintos. Teniendo en cuenta esto, decida la mejor manera de agregar esta información a la base de datos (y justifique su decisión en un comentario), genere un esquema de validación para dicha información y agregue algunos documentos de ejemplo.

Puntos a tener en cuenta

- Resolver las consultas sin vistas.
- Mostrar únicamente los campos pedidos en la consigna.
- Se piden que los campos que se devuelven sean valores escalares a menos que se pida los valores de los campos devueltos podrán ser documentos anidados, arreglos de escalares o arreglos de documentos.
- Buscar hacer la consulta de la forma más sencilla y eficiente posible.
- Se evaluará el correcto formato de las soluciones:
    - El código entregado debe ser legible.
    - Utilizar indentación de espacios de manera uniforme.

Entrega

- Se entregará un archivo comprimido `**soluciones.zip**` (con `**soluciones.js**` adentro, por favor no poner otro nombre) con las soluciones de los 7 ejercicios. Separar las soluciones mediante comentarios de Javascript (/* */ o //).
- La entrega se hará mediante el Aula Virtual en el correspondiente apartado.
    - Tendrán hasta las 18:30 para que se considere una entrega completa. La recomendación es empezar a subir el archivo a las 18 para evitar cualquier eventualidad.
    - Si se entrega después de esa hora, el límite serán las 19:00 y se descontará 1 punto por entrega tardía.
    - Después de las 19:00 se cerrará la entrega y el parcial se considerará desaprobado.