Bases de Datos 2022
Parcial II: NoSQL

Sergio Canchi, Cristian Cardellino,
Ramiro Demasi, Juan Cabral

Contexto

La base de datos **analytics** contiene tres colecciones con información de servicios financieros.

Colección	**Descripción**
accounts	Contiene información de las cuentas de los clientes.
customers	Contiene información de los clientes.
transactions	Contiene información de las transacciones de los clientes.

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

$ tar xzvf analytics.tar.gz
$ mongorestore --host <host> --drop --gzip --db analytics analytics/

Consignas

1. Buscar los clientes que no tengan el campo *active* y que o bien posean más de 4 cuentas o bien nacieron entre Abril de 1995 y Marzo de 1997 inclusives. Listar el nombre, email, fecha de nacimiento y cantidad de cuentas. Limitar el resultado a los 50 primeros clientes de acuerdo al orden alfabético.
2. Actualizar las cuentas que tengan un límite entre 8000 y 9000 inclusives, agregando un nuevo campo "class" con el valor "A" si la cuenta tiene hasta dos productos y con el valor "B" si tiene 3 o más productos.
3. Buscar las transacciones donde la cantidad de transacciones sea mayor a 94. Listar id de transacción, id de la cuenta, y solo aquellas transacciones que tengan el código de transacción igual a "buy" y con "total" mayor a 500000. Mostrar el resultado ordenados por el id de la cuenta en orden decreciente.
   HINTS: (i) El operador $filter puede ser de utilidad. (ii) Notar que el valor del campo total está en string y requiere conversión.
4. Crear la vista "transactionCountByCode" que lista el id de transacción, id de la cuenta, cantidad de transacciones, cantidad de transacciones de compra (transacciones con transaction_code igual a buy) y cantidad de transacciones de venta (transacciones con transaction_code igual a sell). Listar el resultado ordenados por cantidad de transacciones (orden decreciente).

5. Calcular la suma total, suma total de ventas y suma total de compras de las transacciones realizadas por año y mes. Mostrar el resultado en orden cronológico. No se debe mostrar resultados anidados en el resultado.
   HINT: El operador $cond o $switch puede ser de utilidad.
6. Especificar reglas de validación en la colección *transactions* (a) usando JSON Schema a los campos: *account_id*, *transaction_count*, *bucket_start_date*, *bucket_end_date* y *transactions* (y todos sus campos anidados). Inferir los tipos y otras restricciones que considere adecuados para especificar las reglas a partir de los documentos de la colección. (b) Luego añadir una regla de validación tal que *bucket_start_date* debe ser menor o igual a *bucket_end_date*. (c) Testear la regla de validación generando dos casos de falla en la regla de validación y dos casos donde cumple la regla de validación. Aclarar en la entrega cuales son los casos que fallan y cuales cumplen la regla de validación. Los casos no deben ser triviales.
7. Listar el *username* del cliente, cuentas y sus transacciones más recientes de cada cuenta asociada. Para ejemplificar se muestra un fragmento del resultado:

```
[
 {
  username: 'abrown',
  accounts: [ 146756, 120270 ],
  most_recent_transactions: [
   {
    account_id: 146756,
    date: ISODate("2015-02-19T00:00:00.000Z"),
    amount: 2945,
    transaction_code: 'buy',
    symbol: 'ebay',
    price: '23.852746211522603658750085742212831974029541015625',
    total: '70246.33759293406777501900251'
   },
   {
    account_id: 120270,
    date: ISODate("2007-01-12T00:00:00.000Z"),
    amount: 6026,
    transaction_code: 'buy',
    symbol: 'ebay',
    price: '12.617388927046476965188048779964447021484375',
    total: '76032.3856743820701922231819 5'
   }
  ]
 },
…
```

   Donde se puede ver que el cliente 'abrown' tiene dos cuentas asociadas y se muestra la transacción más reciente (de acuerdo al campo date) de cada cuenta del cliente. Se puede asumir que el campo *transactions* está ordenado.
   Listar el resultado en orden alfabético.
   HINT: El operador $map puede ser de utilidad.
   Extra: Este ejercicio suma hasta 1 punto, pero no resta.

Puntos a tener en cuenta

- Algunas consultas pueden resolverse con find y aggregate. En estos casos se puede resolver eligiendo alguno de estos métodos.
- Resolver las consultas sin vistas salvo que se lo pida explícitamente.
- 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.js.gz**` o `**soluciones.zip**` (con `**soluciones.js**` adentro) 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.