Selección multicolección

Tiempo estimado: 15min

Hasta el momento, hemos usado consultas simples, recordemos, aquellas que pueden procesar documentos procedentes de una única colección. Ahora, vamos a presentar las consultas multicolección, aquellas cuyos resultados combinan documentos procedentes de varias colecciones.

La lección comienza introduciendo los conceptos de consulta multicolección y de reunión. A continuación, se presenta cómo realizar las operaciones de reunión: producto cartesiano y reunión interna. Después, se describe las subconsultas. Y finalmente, se presenta el comando LET.

Al finalizar la lección, el estudiante sabrá:

- Cómo realizar consultas multicolección mediante AQL.
- Qué es una subconsulta.
- Cuál es la diferencia entre subconsulta correlacionada y subconsulta no correlacionada.
- Qué es una reunión.

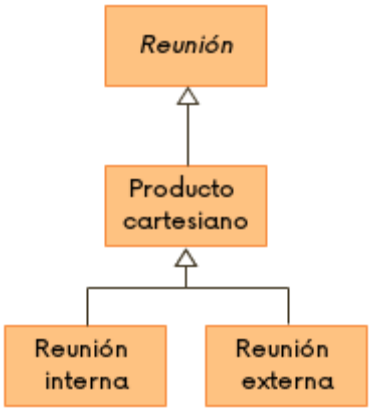
Introducción

Una consulta multicolección (*multicollection query*) es aquella que extrae documentos de varias colecciones. Generalmente, las consultas multicolección se realizan mediante reuniones y subconsultas.

En SQL, se utiliza una única cláusula FROM para realizar la combinación mediante reuniones (*joins*), aquellas que combinan filas de varias tablas en una única tabla intermedia, para así poder ser procesada por el resto de operaciones de la consulta SELECT. En AQL, se usa varios comandos FOR, uno para cada colección, y se reúnen mediante comandos FILTER. Veamos un ejemplo para ir abriendo boca:

```
FOR b IN bands
FOR a IN albums
FILTER b._key == a.band
RETURN {band: b.name, album: a.title}
```

Recordemos los tipos de reunión que se pueden realizar en SQL, para analizar así lo que se puede hacer también con AQL: producto cartesiano, reunión interna y reunión externa.



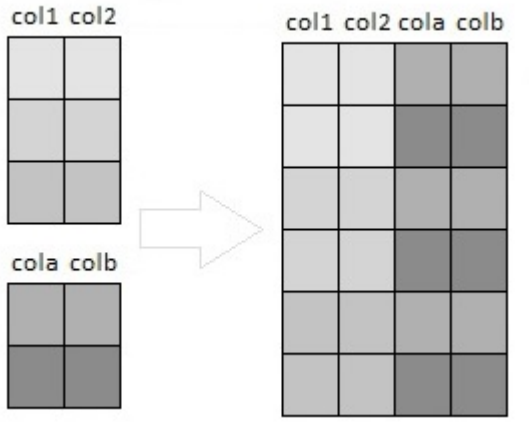
El producto cartesiano (*cartesian product*) o reunión cruzada (*cross join*) recibe dos tablas como operandos y devuelve otra como resultado que contiene todas las posibles combinaciones de pares de filas de ambas tablas. Una reunión interna (*inner join*) es aquella que realiza un producto cartesiano y, a continuación, una operación de filtro que restringe la tabla resultado a aquellas que cumplen una determinada condición. Finalmente, una reunión externa (*outer join*) es aquella que, tras realizar una reunión interna, garantiza que las filas de una o de las dos tablas, según el tipo de reunión, formarán siempre parte del resultado final, aunque no hayan encontrado pareja en la otra tabla. Todas estas filas

que no se han podido emparejar, son añadidas por la operación.

Vamos a ver cada reunión de manera independiente y cómo implementarlas con AQL.

Productos cartesianos

El producto cartesiano (*cartesian product*) o reunión cruzada (*cross join*) recibe dos colecciones como operandos y devuelve otra que tiene como contenido todas las posibles combinaciones de documentos de ambas colecciones.



En SQL, el producto cartesiano se puede especificar de dos maneras distintas, mediante el operador coma o el operador CROSS JOIN:

```
Tabla, Tabla
Tabla CROSS JOIN Tabla
```

En AQL, no hay más que añadir un comando FOR para cada colección, siendo el comando RETURN el encargado de determinar el esquema final del resultado. Veamos un ejemplo ilustrativo:

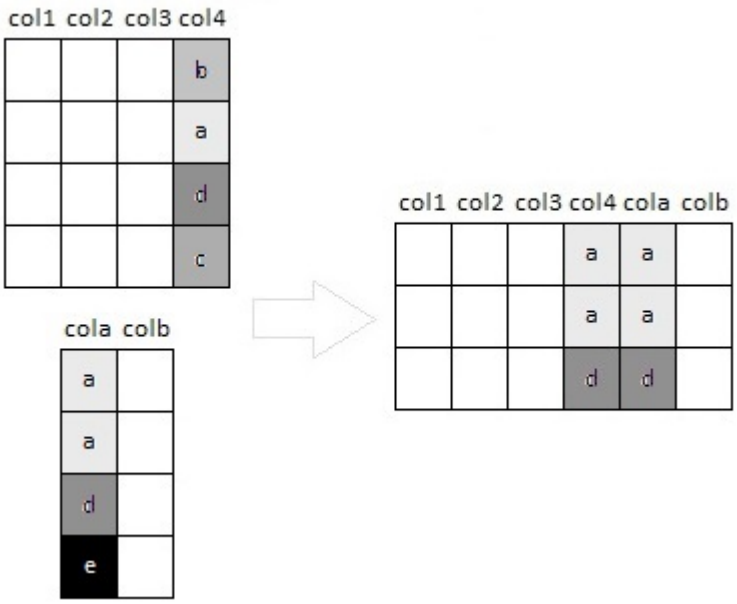
```
127.0.0.1:8529@prueba> db._query("FOR b IN bands RETURN b._key").toArray()
[
  "phoenix",
  "doves",
  "athlete"
]
127.0.0.1:8529@prueba> db._query("FOR a IN albums RETURN {band: a.band, title:
a.title}").toArray()
[
  {
    "band" : "doves",
    "title" : "Lost Souls"
  },
  {
    "band" : "athlete",
    "title" : "Black Swan"
  },
  {
    "band" : "athlete",
    "title" : "Beyond the Neighbourhood"
  },
  {
    "band" : "doves",
    "title" : "The Last Broadcast"
  }
]
127.0.0.1:8529@prueba> db._query("FOR b IN bands FOR a IN albums SORT b.name RETURN
{band: b.name, title: a.title}").toArray()
[
  {
    "band" : "Athlete",
    "title" : "Lost Souls"
  },
```



```
  {
    "band" : "Athlete",
    "title" : "Black Swan"
  },
  {
    "band" : "Athlete",
    "title" : "Beyond the Neighbourhood"
  },
  {
    "band" : "Athlete",
    "title" : "The Last Broadcast"
  },
  {
    "band" : "Doves",
    "title" : "Lost Souls"
  },
  {
    "band" : "Doves",
    "title" : "Black Swan"
  },
  {
    "band" : "Doves",
    "title" : "Beyond the Neighbourhood"
  },
  {
    "band" : "Doves",
    "title" : "The Last Broadcast"
  },
  {
    "band" : "Phoenix",
    "title" : "Lost Souls"
  },
  {
    "band" : "Phoenix",
    "title" : "Black Swan"
  },
  {
    "band" : "Phoenix",
    "title" : "Beyond the Neighbourhood"
  },
  {
    "band" : "Phoenix",
    "title" : "The Last Broadcast"
  }
]
127.0.0.1:8529@prueba>
```

Reuniones internas

Una reunión interna (*inner join*) es aquella que realiza un producto cartesiano y, a continuación, una operación de filtro que restringe el resultado a aquellas combinaciones que cumplen una determinada condición. En otras palabras, es el producto cartesiano de dos o más colecciones con algunas combinaciones eliminadas, aquellas que no cumplen la condición de filtro. Es el tipo de reunión más común cuando se trabaja con consultas multicolección y se utiliza, principalmente, para combinar colecciones con relaciones padre-hijo.



En SQL, las reuniones internas se realizan mediante el operador INNER JOIN:

```
Tabla [INNER] JOIN Tabla ON CondiciónDeReunión
```

En AQL, basta con añadir el comando FILTER mediante el cual restringir las combinaciones. He aquí un ejemplo ilustrativo:

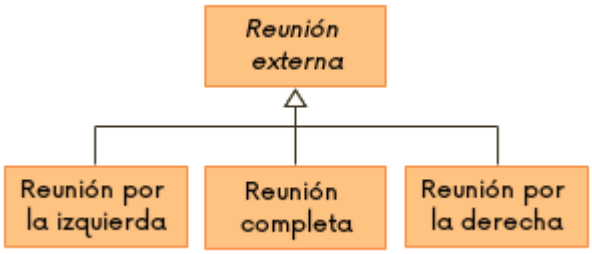
```
127.0.0.1:8529@prueba> db._query("FOR b IN bands FOR a IN albums FILTER b._key ==
a.band SORT b.name RETURN {band: b.name, title: a.title}").toArray()
[
  {
    "band" : "Athlete",
    "title" : "Black Swan"
  },
  {
    "band" : "Athlete",
    "title" : "Beyond the Neighbourhood"
  },
  {
    "band" : "Doves",
    "title" : "Lost Souls"
  },
  {
    "band" : "Doves",
    "title" : "The Last Broadcast"
  }
]
127.0.0.1:8529@prueba>
```

El predicado que sirve como filtro se conoce formalmente como condición de reunión (*join condition*). Esta condición es la que debe cumplir toda combinación para, digamos, llegar al comando RETURN y así poder formar parte del resultado final de la consulta.

Reuniones externas

Una reunión externa (*outer join*) es aquella que, tras realizar una reunión interna, garantiza que los documentos de una o de las dos colecciones, según el tipo de reunión, formarán siempre parte del resultado final, aunque no hayan encontrado pareja en la otra colección. Todos esos documentos que no se han podido emparejar, son añadidos por la propia operación de reunión. A la colección que contendrá todos sus documentos en el resultado, se la conoce formalmente como colección preservada (*preserved collection*).

Se distingue entre reuniones externas por la derecha, por la izquierda y completa.



La reunión externa por la izquierda (*left outer join*) preserva la colección que se indica como operando izquierdo. La reunión externa por la derecha (*right outer join*), la colección indicada como operando derecho. Y la reunión externa completa (*full outer join*), ambas.

En SQL, este tipo de reuniones se puede realizar fácilmente mediante OUTER JOINs. En cambio, en AQL, se puede hacer, pero no es tan fácil como el producto cartesiano y la reunión interna. Y suelen ser muy costosas en cuanto a recursos invertidos para llevarlas a cabo.

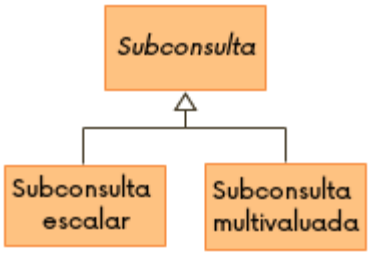
Subconsultas

Una subconsulta (*subquery*) es aquella que se encuentra dentro de otra, por ejemplo, en el predicado del comando FILTER o en un comando LET. A la consulta en la que se encuentra declarada la subconsulta, se la conoce formalmente como consulta externa (*outer query*); y a los documentos de la consulta externa, se los conoce como documentos externos (*outer documents*).

Las subconsultas se pueden usar en comandos LET o delimitarse mediante paréntesis. Para ir abriendo boca, veamos un ejemplo:

```
FOR b IN bands
FILTER b._key IN (
  FOR a IN albums
  RETURN a.band
)
RETURN b
```

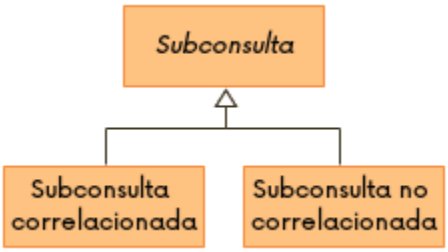
Atendiendo al número de filas que devuelve una subconsulta, se puede clasificar en escalar y multivaluada.



Una subconsulta escalar (*scalar subquery*) es aquella que devuelve un único documento con un único campo. En cambio, una subconsulta multivaluada (*multivalued subquery*) es aquella que devuelve varios valores, sea en forma de varios documentos y/o campos.

Correlatividad

La correlatividad (*correlativity*) hace referencia a la relación entre la subconsulta y la consulta externa en la que se define. Atendiendo a esta relación de subordinación, se distingue dos tipos de subconsultas, las correlacionadas y las no correlacionadas.



Una subconsulta no correlacionada (*non-correlated subquery*) es aquella cuya ejecución es independiente del contenido de la consulta externa en la que se define. Mientras que una subconsulta correlacionada (*correlated subquery*) es aquella cuya ejecución depende de la consulta externa.

Subconsultas no correlacionadas

Una subconsulta no correlacionada (*non-correlated subquery*), también conocida como subconsulta independiente (*independent subquery*) o subconsulta autocontenida (*self-contained subquery*), es aquella cuya ejecución es independiente del contenido de la consulta externa. A diferencia de las subconsultas correlacionadas o dependientes, no hace ninguna referencia al documento actual externo. Así pues, devuelve siempre el mismo resultado para todos los documentos externos y, por lo tanto, se ejecuta una única vez para toda la consulta externa.

Veamos un ejemplo. A continuación, se muestra cómo obtener las bandas que tienen al menos un álbum registrado en la colección albums:

```
FOR b IN bands
FILTER b._key IN (
  FOR a IN albums
  RETURN DISTINCT a.band
)
RETURN b
```

Observe que la subconsulta es una consulta completamente independiente. No usa el documento externo en curso. La subconsulta devuelve siempre el mismo resultado, la lista de bandas que tienen registrado un álbum en la colección albums. Por esta razón, como *no* depende en absoluto de la consulta externa, el motor de bases de datos puede ejecutarla una única vez, ya sea al comienzo de la ejecución de la consulta o justo antes de su utilización con el primer documento externo en procesamiento.

Subconsultas correlacionadas

Una subconsulta correlacionada (*correlated subquery*), también conocida como subconsulta dependiente (*dependent subquery*) o subconsulta subordinada (*subordinate subquery*), es aquella cuya ejecución depende de la consulta externa. Referencia al documento actual de la consulta externa, lo que significa que el resultado de la subconsulta puede ser distinto atendiendo al documento externo y, por lo tanto, debe ejecutarse para cada uno de ellos.

He aquí un sencillo ejemplo de subconsulta correlacionada cuyo resultado es proyectado en la consulta externa:

```
FOR b IN bands
FILTER length(
  FOR a IN albums
  FILTER b._key == a.band
  RETURN a
) > 0
RETURN b
```

Para cada banda, la consulta determina si tiene algún álbum registrado en la colección albums. Observe que en la subconsulta, su comando FILTER hace referencia al documento actual externo mediante la variable b. En nuestro caso, la colección bands tiene sólo tres documentos, por lo que la subconsulta se ejecutará tres veces.

Comando LET

AQL proporciona el comando LET para crear variables para cada uno de los documentos en curso. Su sintaxis es como sigue:

```
LET variable = valor
```

Esta variable puede ser accedida a partir de su definición en los siguientes comandos de la consulta. Así pues, por ejemplo, si deseamos conocer los álbumes registrados para cada banda, podríamos hacer una consulta como la siguiente:

```
FOR b IN bands
LET albs = lenght(
  FOR a IN albums
  FILTER b._key == a.band
  RETURN a
)
RETURN merge(b, {albums: albs})
```

