Interrogazioni del database

**RICONTROLLARE QUESTO FILE, IN PARTICOLARE IL CODICE**

Questo documento riassume tutte le interrogazioni (transazioni e query) fatte al database. Per chiarezza, le query sono state scritte in questo documento come istruzioni SQL durante la fase di progettazione, ma la vera implementazione dipende dal DBMS utilizzato. In ogni caso, durante la progettazione delle query si è cercato di tenere conto del DBMS utilizzato, per minimizzarne il numero di accessi (costosi), ma anche il numero di indici. *Non* è stata utilizzata l’operazione di *JOIN*.

# Costi

**Costo di una query**: Numero di accessi in lettura: ( è il numero di entità ritornate )

# Query e procedure di interrogazione del database

## Ricerca di tutte le coppie (Uploader.identificativoAttore, File.identificativoFile) corrispondenti ad un file destinato ad un dato consumer

*Parametri:*

* *idConsumer [Consumer.identificativoAttore]*

*Query:*

SELECT Uploader AS idUploader, File

FROM RelazioneUploaderConsumerFile

WHERE RelazioneUploaderConsumerFile.Consumer = *idConsumer*

**Costo:** Numero di accessi in lettura:   
 è un Uploader che serve il particolare Consumer (la sommatoria ha il numero di addendi   
 pari al numero di Uploader distinti che servono il particolare Consumer) è il numero di file caricati dall’Uploader e destinati al particolare Consumer.

**Esempio numerico**

Si supponga che un Consumer riceva 2 documenti al mese da 10 Consumer distinti, allora, questa query, per il Consumer considerato, avrà il seguente costo:

* 21 accessi dopo un mese dalla registrazione del consumer;
* 41 accessi dopo due mesi dalla registrazione del consumer;
* 121 accessi dopo sei mesi dalla registrazione del consumer;
* 481 accessi dopo due anni dalla registrazione del consumer;
* 2401 accessi dopo dieci anni dalla registrazione del consumer.

**Osservazione:**

Se il valore di *Uploader* (idUploader) presente in *RelazioneUploaderConsumerFile* è lo stesso di quello presente nell’attributo marcato con l’annotation **@Id** (di Objectify) nella classe Java usata per rappresentare un *Uploader* e se quest’ultima classe *Uploader* (in Java) non eredita alcuna classe marcata dall’annotation **@Parent** (di Objectify), allora la chiave (di tipo **com.googlecode.objectify.Key<T>** ) dell’entità *Uploader* può essere calcolata come segue:

Key<Uploader> chiaveUploader = com.googlecode.objectify.Key.*create*(Uploader.class, idUploader);

dove idUploader è il valore in *RelazioneUploaderConsumerFile.Uploader* che corrisponde a quello dell’attributo marcato con @Id nella classe *Uploader*.

Quindi, a partire dal risultato di questa query si possono costruire immediatamente le chiavi, sia per collegarsi ad un file sia per collegarsi ad un uploader, associati al consumer dato, grazie alle quali si può accedere direttamente al corrispettivo valore salvato nel database (costo di 1 accesso), cioè, dato *idConsumer* (l’identificativo del consumer, parametro di partenza), la precedente query permette di ottenere le chiavi sia degli Uploader sia dei File associati e da queste ultime si riescono ad ottenere direttamente (al costo di un accesso) le entità referenziate:

*// Codice Java con Objectify (parametro:* idUploader*)*

*// Questo codice è solo una BOZZA di funzionamento (codice NON testato)*

List<RelazioneUplaoderConsumerFile> risultatiQuery;

Set<Key<Uploader>> setChiaviUploader = new HashSet<>(); // Set evita duplicate

Set<Key< File >> setChiaviFile = new HashSet<>(); // Set evita duplicati

List<Uploader> listaUploaderAssociati = new ArrayList<>();

List<File> listaFileAssociati = new ArrayList<>();

risultatiQuery = ofy().load().type(RelazioneUplaoderConsumerFile.class)  
 .filter(“Consumer”, String.valueOf(idConsumer))  
 .list(); *// esecuzione query [Q1]*

for( RelazioneUplaoderConsumerFile risultato : risultatiQuery ) {

Key<Uploader> chiaveUploader = com.googlecode.objectify.Key.*create*(Uploader.class, idUploader);

setChiaviUploader.add( chiaveUploader );

Key<File> chiavefile = com.googlecode.objectify.Key.*create*(File.class, idFile);

setChiaviFile.add( chiavefile );

}

listaUploaderAssociati = ofy().load().keys( chiaveUploader ).now();

## Ricerca di tutti gli Uploader che hanno caricato un file per il Consumer dato come parametro.

*Parametri:*

* *idConsumer [Consumer.identificativoAttore]*

*Procedura:*

1. esegui query [Q1] e salva i risultati (distinti) degli *Uploader* in *setChiaviUploader*
2. per ogni chiave in *setChiaviUploader* recupera l’entità *Uploader* corrispondente dal database

**Costo:** Numero di accessi in lettura:   
 è il numero di accessi in lettura dovuti alla query [Q1]  
 è il numero di Uploader distinti che servono il particolare Consumer (è la   
 dimensione di *setChiaviUploader*).

## Ricerca di tutti i File che sono stati caricati da uno specifico Uploader e destinati ad uno specifico Consumer.

*Parametri:*

* *idConsumer [Consumer.identificativoAttore]*
* *idUploader [Uploader.identificativoAttore]*

*Procedura (assumendo di aver già eseguito la query* [Q1][[1]](#footnote-1)*):*

1. esegui query [Q1] e salva i risultati (distinti) dei *File* in *setChiaviFile*
2. per ogni chiave in *setChiaviFile* recupera l’entità *File* corrispondente dal database

**Costo:** Numero di accessi in lettura:   
 è il numero di accessi in lettura dovuti alla query [Q1]  
 è il numero di Uploader distinti che servono il particolare Consumer (è la   
 dimensione di *setChiaviUploaders*).

1. Se *non* è già stata eseguita la query [Q1], non conviene eseguirla in questo contesto perché avendo *Consumer.identificativoAttore* e *Uploader.identificativoAttore* come parametri, si potrebbero filtrare i risultati già nella query, così da diminuire il numero di occorrenza restituite (e quindi di diminuire il costo degli accessi):

   SELECT File

   FROM RelazioneUploaderConsumerFile

   WHERE RelazioneUploaderConsumerFile.Consumer = *idConsumer* AND  
    RelazioneUploaderConsumerFile.Uploader = *idUploader*

   Il risultato della query sarà il *setChiaviUploaders* da cui ricavare le entità.

   **Costo:** Numero di accessi in lettura:   
    è il numero di file caricati dall’Uploader specificato e destinati al consumer specificati   
    (fattore 2 dovuto al recupero di chiavi date dalla query a cui si aggiunge il   
    costo ulteriore di accessi per recuperare effettivamente date le chiavi).

   Nota: il costo di quest’ultima query è sempre minore o uguale di [Q1]. [↑](#footnote-ref-1)