

IMPLEMENTASI DATA IMPORT APACHE SOLR UNTUK KEPERLUAN INDEXING DATA BUKU

Viranda Noratika Anwar

D3 Manajemen Informatika, Fakultas Teknik, Universitas Negeri Surabaya

viranda.noratika05@gmail.com

Abstrak

Pertumbuhan lalu lintas data dalam komunikasi *client-server* membutuhkan strategi khusus agar memberikan waktu respons rendah. Terlebih lagi pada proses pencarian yang seringkali melibatkan *query* yang rumit. Salah satu cara untuk mengurangi waktu proses ini adalah dengan melakukan proses *indexing*. Dalam proses *indexing* ini penulis menggunakan *Apache Solr* sebagai *platform*. Salah satu fitur dari *Apache Solr* adalah melakukan *import*, yaitu *full-import* dan *delta-import* dari basis data *MySQL* ke dalam *Solr*. Dalam penelitian ini dilakukan proses *import* seluruh atau parsial. *Full-import* atau impor seluruh adalah melakukan transaksi *import data* secara menyeluruh dari *database*. Sedangkan, *delta-import* atau impor parsial merupakan transaksi *import data* yang dilakukan dengan menarik *data* yang dimodifikasi. Penelitian ini dilakukan untuk mengetahui proses *indexing* pada *Apache Solr*. Hasil menunjukkan bahwa *data* berhasil di-*import* menggunakan kedua metode.

Kata kunci: *Apache Solr*, *Indexing*, Impor, Basis Data

Abstract

Data traffic growth in client-server communication requires special methods to provide low response time. Such technique is greatly needed in searching process that frequently involves complex queries. One attempt to reduce such process is indexing technology that arranges documents in systematic manner. This study uses Apache Solr as platform to perform the activity. Import process, full or delta, is required prior to locating document. While full import transfers all database contents to Solr, the latter only process partially and considers modified data. This paper discusses the steps and results these processes.

Keywords: *Apache Solr*, *Indexing*, *Import*, *Database*.

PENDAHULUAN

Lalu lintas data dalam komunikasi *client-server* telah mencakup banyak bidang termasuk pendidikan. Setiap instansi Pendidikan memiliki koleksi pustaka dalam jumlah besar dalam berbagai macam kategori. Upaya digitalisasi dilakukan untuk mempermudah pengelolaan dan pencarian literatur dalam koleksi tersebut. Namun, seiring dengan peningkatan jumlah koleksi, proses pencarian, yang seringkali melibatkan *query* kompleks, memakan waktu eksekusi tinggi. Salah satu metode yang digunakan untuk membantu proses pencarian adalah *indexing*. Proses ini akan mengelola dokumen-dokumen secara sistematis sehingga memudahkan proses pencarian. *Indexing* juga dapat membantu meningkatkan performa *database* dalam hal proses eksekusi *query*. [1]

Penelitian ini menguji proses *import database MySQL* ke dalam *Solr* dengan fitur *delta-import* dan *full-import* milik *platform Apache Solr*, dan melakukan pencarian *data* yang telah di-*import*. Pengujian juga dilakukan untuk mengetahui instalasi *Apache Solr*, proses dan hasil konfigurasi *import*.

Artikel ini diorganisasi sebagai berikut. Dasar teori dan literature terkait akan dibahas ada bab Kajian Pustaka.

Desain eksperimen serta pembahasan dari simulasi akan dibahas pada bab Rancangan Simulasi dan Hasil dan Pembahasan. Rangkuman capaian studi ini akan dituliskan pada bab Kesimpulan.

KAJIAN PUSTAKA

Penelitian Terdahulu

Penelitian yang telah banyak diusulkan oleh peneliti terdahulu melalui jurnal ilmiah dan *proceeding*. Jurnal-jurnal tersebut melakukan suatu implementasi proses *indexing* dimana pada umumnya melakukan pencarian *full text* menggunakan *database*. Penulis pada artikel [2] menerapkan *indexing* menggunakan *inverted index*, menuliskan bahwa proses *indexing* paling lama memakan waktu saat proses penulisan *index* yaitu sekitar 59,27% dari total waktu *indexing* dari hal tersebut diketahui *index* tersebut mempercepat proses pencarian hingga 3800 kali lipat dibandingkan dengan pencarian konvensional dengan menggunakan *full text search*.

Sedangkan pada artikel [3], penulisnya menerapkan pencarian *full text* pada data berita *online* yang menggunakan *sistem indexing* milik *Apache Solr*. Dituliskan bahwa hasil pengujian pencarian berita

diperoleh pencarian *full text* pada *Solr* lebih cepat 2,5 kali lipat dibandingkan dengan *MySQL*.

Apache Solr

Solr merupakan salah satu bentuk *search platform* yang bersifat *open-source* dari *Apache*. Dimana fitur utamanya adalah pencarian berbasis *text*, *hit highlight*, *faceted search*, *dynamic clustering*, *database integration* dan penanganan terhadap *rich document* (*word*, *pdf* dsb). Pencarian menggunakan *Solr* bisa menggunakan *URL* dan hasil pencariannya bisa berupa *XML*, *CSV*, *PHP*, *Ruby*, *Python*, maupun *JSON*. [4] *Apache Solr* menggunakan *Lucene library* untuk pencarian teks lengkap. *Apache Solr* adalah sebuah alat hebat dengan kemampuan pencarian yang luar biasa. Untuk mencari dokumen, ia melakukan operasi pengindeksan yaitu mengkonversi dokumen ke dalam format yang dapat dibaca oleh mesin. Juga *querying*, Memahami persyaratan permintaan yang diminta oleh pengguna. Istilah-istilah ini bisa berupa gambar, kata kunci, dan banyak lagi. Kemudian, pemetaan: *Solr* memetakan *query* pengguna ke dokumen yang disimpan dalam basis data untuk menemukan hasil yang sesuai. Lalu, setelah mesin mencari dokumen yang diindeks, ia memberi peringkat *output* sesuai relevansinya.

Beberapa kelebihan dari *solr* ini adalah fitur *full-import* dan *delta-import*. *Full-import* adalah ketika *SOLR* melakukan koneksi ke *database* dan mengambil semua data berdasarkan *query* yang sudah di atur konfigurasi di *SOLR*. Sedangkan *delta-import* adalah menarik sebagian data yang paling *update* berdasarkan waktu terakhir *import* data. [3]

Contoh *XML*, *config solr* untuk *dataimport* terlihat pada tabel 1. [5]

Tabel 1. Contoh Syntax Konfigurasi *DataImport*

```
<dataConfig>
  <dataSource type="JdbcDataSource"
    driver="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost/produk"
    user="root" password="rahasia"/>
  <document>
    <entity name="foodproducts"
      query="SELECT id, fCode, fName, fManufacture
      FROM foodproducts">
      <field column="id" name="id" />
      <field column="fCode"
        name="food_code" />
      <field column="fName"
        name="food_name" />
      <field column="fManufacture"
        name="food_man" />
    </entity>
  </document>
</dataConfig>
```

Indexing

Indexing adalah proses yang memperlakukan sebuah objek struktur yang menyimpan nilai spesifik tidak bergantung pada struktur *table* yang membuat pencarian data menjadi lebih cepat. [1] *Indexing* berfungsi untuk membantu mempercepat proses eksekusi sebuah *query* ke sebuah *database* yang sudah berisi banyak data. *Index* dari sebuah *database* dapat dianalogikan sebagai sebuah rak

didalam perpustakaan. Dimana satu rak itu merupakan satu jenis kelompok buku. Misalnya, dalam suatu perpustakaan terdapat berbagai macam buku. Buku-buku tersebut dikelompokkan berdasarkan jenisnya ke dalam masing-masing rak. Misalkan, buku pemrograman masuk ke dalam rak komputer, atau buku bahasa Indonesia masuk ke dalam rak bahasa, maka kelompok bahasa tersebut merupakan index bahasa, dan kelompok komputer merupakan index computer, dan seterusnya. Dari penggambaran *index* tersebut dapat dimanfaatkan, sehingga proses pembacaan *query* dapat terbantu oleh adanya *indexing* dan membuat waktu proses menjadi lebih cepat.

Full-Import dan Delta-Import

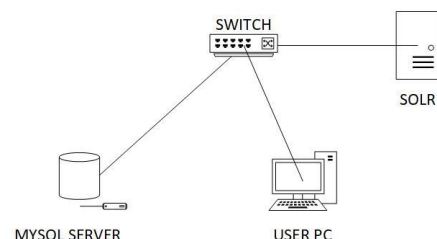
Saat operasi *full-import* dimulai *thread* baru, atribut status dalam respon akan menampilkan tanda sibuk. Operasi mungkin akan memerlukan beberapa waktu tergantung pada ukuran dataset. Ketika perintah *full-import* dijalankan, ia menyimpan waktu mulai operasi tersebut didalam file yang terletak di suatu dokumen '*conf/.....*' (file ini dapat dikonfigurasi). Ini akan secara otomatis tersimpan suatu *timestamp* dan akan berguna ketika operasi *delta-import* dijalankan. Perlu diketahui, bahwa *query* yang diperintahkan ke *solr* tidak akan diblokir selama operasi *full-import* berjalan

Sama dengan saat operasi *full-import* dimulai, *delta-import* pun dimulai pada *thread* baru, dan juga akan menampilkan tanda sibuk, serta lamanya waktu berlangsung bergantung pada besarnya ukuran *dataset*. Ketika perintah *delta-import* dijalankan, ini akan membaca waktu mulai yang tersimpan didalam '*conf/.....*' yang menggunakan *timestamp* milik *full-import* terakhir beroperasi. Setelah selesai, ia juga akan membuat *timestamp* baru yang disimpan pula didalam '*conf/...*' dan begitu seterusnya dalam memilih data yang akan di-*delta-import* kan berdasarkan *timestamp* terbaru. Operasi *delta-import* bisa dijalankan dengan perintah url '*http://<host>:<port>/solr/dataimport?Command=delta-import*'. [6]

RANCANGAN SIMULASI

Arsitektur Sistem

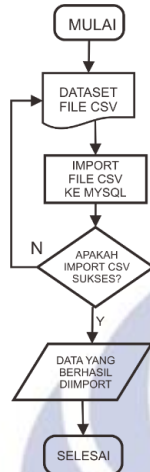
Pada tahapan ini, penulis akan menggambarkan dan menjelaskan mengenai *DataImport* dengan menggunakan *Apache Solr* dan topologi jaringan yang digunakan. Pada gambar 1 adalah topologi jaringan yang digunakan:



Gambar 1. Topologi Jaringan

Dari gambaran topologi pada gambar 1 dapat diketahui bahwa *Solr server* dan *MySQL Server* dapat diakses oleh *PC user*. *PC user* merupakan *desktop* milik penulis yang mana digunakan untuk mengakses *MySQL* dan *Solr*, dengan *switch* sebagai penghubung antara ketiganya. *MySQL* sebagai *database* dan *Solr* sebagai *peng-index*.

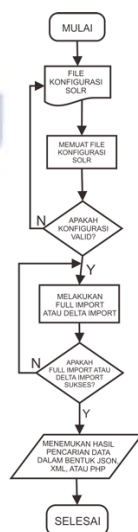
1) Perancangan Alur Kerja Import CSV ke MySQL



Gambar 2. Alur kerja untuk proses *insert CSV file*

File dataset berupa *CSV* yang didapat dari mengunduh di suatu *website*. Kemudian, *file dataset* tersebut diatur sedemikian rupa untuk dapat di-import ke dalam *MySQL*. Kemudian, *file CSV* di-import ke dalam *MySQL*. Apabila *import* sukses dilakukan, maka *data* berhasil di-insert ke dalam *MySQL* dan siap untuk digunakan untuk pengujian. Namun, apabila *import file CSV* tidak berhasil dilakukan dan terjadi *error*, maka perlu dilakukan pemeriksaan ulang terhadap *file CSV*. *File CSV* berhasil di-import, *insert file CSV* selesai dilakukan.

2) Perancangan Alur Kerja Full-Import dan Delta-Import pada Apache Solr



Gambar 3. Alur kerja untuk proses *full-import* dan *delta-import*

File konfigurasi yang dibuat berisi *syntax* yang menghubungkan *MySQL* dengan *Apache Solr*, serta *syntax* untuk mendeskripsikan *query data MySQL* yang akan ditarik ke *Solr*. Kemudian, konfigurasi yang dibuat akan dimuat saat menjalankan *Solr*. Setelah *Solr* dijalankan, dan dapat dilihat pada *dashboard Solr* pada *core* di *tab dataimport*, *config* di-load dan akan muncul isi dari konfigurasi pada *file konfigurasi* yang telah dihubungkan dengan *Solr*. Dapat dilihat apakah sudah *valid* dengan konfigurasi yang dibuat pada *file config*. Jika *valid*, maka *full-import* atau *delta-import* dapat dieksekusi. Jika tidak *valid*, maka dilakukan pengecekan ulang dengan *file konfigurasi* kembali.

Setelah konfigurasi *valid* maka *full-import* atau *delta-import* dieksekusi. Jika operasi *import* tidak sukses, maka dilakukan pengecekan kembali apakah konfigurasi *valid* atau tidak. Jika *full-import* atau *delta-import* berhasil dilakukan, maka *Solr* akan menampilkan dan menemukan hasil pencarian *data* yang di-import melalui *query* di *Solr*. Operasi selesai.

Skenario Pengujian

Dalam uji coba yang dilakukan pada implementasi ini menggunakan salah satu fitur dari *Apache Solr*, yaitu *dataImport*.

Skenario pengujian yang dilakukan yaitu mengacu pada implementasi proses *import* menggunakan *dataset* yang diunduh pada suatu *web*, sebagai bahan yang akan diuji.

- Import file *CSV* ke dalam *MySQL*
- Full-import* untuk data awal menguji konfigurasi
- Full-import* 10 data
- Delta-import* 1 record yang dimodifikasi pada *table parent*
- Delta-import* 3 record yang ditambahkan pada *table child*
- Delta-import* 1 record untuk menguji *import* pada modifikasi *data* di *table child*
- Delta-import* 1 record untuk menguji *import* pada *data* tambahan di *table child* apabila *id* sama.

HASIL DAN PEMBAHASAN.

Instalasi

Sebelum melakukan instalasi *Solr*, instal terlebih dahulu aplikasi server *XAMPP*, dan instalasi *Java*. Aktifkan *port MySQL*, dan *port Apache* pada *XAMPP* untuk menjalankan *Solr*. Kemudian, berikut adalah tahap-tahap instalasi *Solr*:

1) Unduh installer *Apache Solr*

Pada penelitian ini, *Apache Solr* dapat diunduh di situs *official*-nya pada 'lucene.apache.org/solr/' yang mana untuk sistem operasi *Windows* memilih *file* unduhan dengan *format .zip*. Setelahnya, ekstrak *file .zip* tersebut dan diletakkan pada direktori sesuai keinginan.

2) Mulai Solr

Menjalankan *Solr* melalui *command prompt*, masuk ke *disk* tempat direktori *file Solr* dipindahkan, dan ketik 'solr start' pada *command prompt*.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.16299.666]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd solr-7.5.0/bin
The system cannot find the path specified.

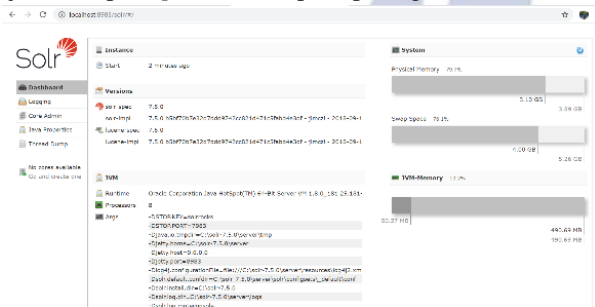
C:\Windows\system32>cd
C:\Windows\system32
C:\Windows\system32>cd C://
C:\>cd solr-7.5.0/bin

C:\solr-7.5.0\bin>solr start
INFO - 2018-10-10 14:23:26.505; org.apache.solr.util.configuration.SS
LCredentialProviderFactory; Processing SSL Credential Provider chain:
env;sysprop
Waiting up to 30 to see Solr running on port 8983
Started Solr server on port 8983. Happy searching!

C:\solr-7.5.0\bin>
```

Gambar 4. Memulai Solr

Pada gambar 4 menunjukkan *Solr* telah aktif dan dapat dijalankan pada *port* 8983 seperti pada gambar 5 berikut.



Gambar 5. Dashboard Solr

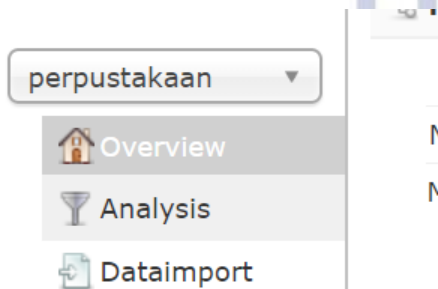
Dashboard Solr telah ditampilkan, maka *Solr* telah dipasang.

3) Buat core 'perpustakaan'

Selanjutnya membuat *document Solr* atau disebut *core* dengan 'solr create -c perpustakaan' pada *command prompt*, dimana 'perpustakaan' merupakan nama dari *core* yang akan dibuat.

```
D:\solr-7.7.1\bin>solr create -c perpustakaan
```

Gambar 6. Memuat core Solr



Gambar 7. Core telah dibuat

Gambar 7 merupakan tampilan *core* 'perpustakaan' pada *dashboard Solr*. *Solr* siap digunakan. Kemudian menginstal *JDBC MySQL* sebagai *connector MySQL* dengan *Solr*, dan menginstal *tools* yang digunakan untuk proses *dataimport* yaitu aplikasi *HeidiSQL*, dan *Notepad++* untuk *tools edit* teks konfigurasi.

Produksi Data

1) Mengunduh dataset

Untuk melakukan proses *import*, data yang digunakan untuk penelitian merupakan data yang sangat banyak yang diletakkan pada *database MySQL*. Data yang sangat banyak tersebut didapat dari mengunduh suatu *dataset*. Dalam *dataset* tersebut memiliki sekitar 200.000 lebih *record data* yang terdapat dalam bentuk *file CSV*.



Gambar 8. Copy data dari dataset

Gambar 8 merupakan contoh penulisan dari *file CSV*. Sebelum di-*import* ke dalam *MySQL*, *file CSV* diambil beberapa data dari keseluruhan data untuk menghindari *load* pada *database* terlalu lama pada saat *import*.

2) Membuat table

Membuat *table database* dilakukan menggunakan aplikasi *HeidiSQL*, seperti pada gambar 9 berikut.

Name:

data_buku

Comment:

Columns:

+

Add

-

Remove

▲

Up

▼

Down

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
1	id	VARCHAR	15	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
2	last_updated	TIMESTAMP		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	CURRENT_TIMESTAMP
3	file_name	VARCHAR	100	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
4	url_pic	VARCHAR	500	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
5	book_tit	VARCHAR	500	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
6	writer	VARCHAR	500	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
7	book_num	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default
8	book_type	VARCHAR	100	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default

Gambar 9. Table data_buku

Setelah dibuat *table* utama bernama 'data_buku', kemudian buat *table child* bernama 'review_user' yang saling berelasi dengan *table* induk, yakni *table child* memiliki *foreign key* dari *table* induk.

Host: 127.0.0.1 Database: db_test Table: review_user Data Query

Basic Options Indexes Foreign keys Partitions CREATE code ALTER code

Name: review_user

Comment:

Columns:

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Col
1	book_id	VARCHAR	15	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		
2	comment	VARCHAR	500	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
3	name_user	VARCHAR	300	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		
4	last_updated	TIMESTAMP		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	CURRENT_TIMESTAMP		

Help Discard Save

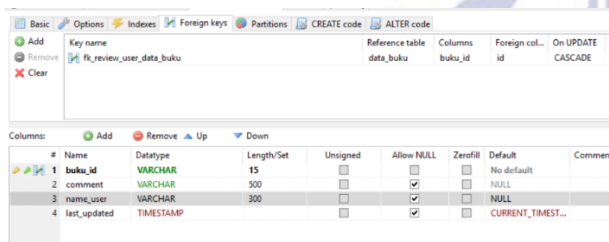
Gambar 10. Table review_user

Pada *table* 'data_buku', terdapat *column* dengan nama 'last_updated' yang memiliki tipe data 'TIMESTAMP' dan memiliki *default value* 'CURRENT_TIMESTAMP ON UPDATE' yang mana *value* tersebut membuat *column* 'last_updated' akan terisi otomatis sesuai dengan waktu ketika data diubah. *Column* 'last_updated' merupakan

kolom pada *table* 'data_buku' yang menunjukkan waktu terakhir mengubah *data*, dan akan digunakan sebagai acuan untuk konfigurasi melakukan *data-import* di *Apache Solr*.

Kemudian, pada *table* 'data_buku' diberikan satu buah *primary key* pada *column* 'id', yang nantinya akan dipakai untuk menarik data oleh *table* 'review_user'. Dan pada *table* 'review_user' diberikan *foreign key* pada *column* 'buku_id', yaitu *primary key* pada *table* 'data_buku'. Didalam *table* 'review_user' tidak terdapat *primary key* dikarenakan *user* tidak perlu melakukan *login* pada sistem pencarian, sehingga bisa terdapat lebih dari satu nama *user* yang sama, serta sebuah buku dapat memiliki lebih dari satu *comment* atau *review* yang sama atau berbeda.

Pada *table* 'review_user' dibuatkan *foreign key* seperti pada gambar 11 berikut.



Gambar 11. Membuat *Foreign Key*

Produksi *data* yang merupakan tahap awal implementasi, sebagian besar termasuk dalam tahap pengujian, sehingga dijelaskan langkah-langkahnya pada tahap pengujian selanjutnya karena merupakan bagian dari implementasi *dataimport*. Seperti modifikasi *data*, atau tambah *record data*.

Konfigurasi DataImport

Konfigurasi yang dituliskan merupakan teks yang berisi *syntax* dalam suatu *file* yang akan ditarik oleh sistem *Solr*.

1) Konfigurasi pada *file* 'solrconfig.xml'

Tabel 2. *Syntax* konfigurasi 'solrconfig.xml'

```
<lib dir="${solr.install.dir:../../../../dist/"
  regex="solr-dataimporthandler-.*\.jar" />
<lib dir="${solr.install.dir:../../../../dist/"
  regex="mysql-connector-java-.*\.jar" />
<requestHandler
  name="/dataimport"
  class="org.apache.solr.handler.dataimport.Data
    ImportHandler"
    >
    <lst name="defaults">
      <str name="config">data-
        config.xml</str>
    </lst>
  </requestHandler>
```

Pada konfigurasi tabel 2, dijelaskan bahwa *file* 'solrconfig.xml' memanggil *request* pada fungsi /dataimport yang mana *class* utamanya memiliki konfigurasi untuk menarik *requestHandler* yang melayani permintaan pada /dataimport, yaitu berupa *document* yang bernama 'data-config.xml'. Sedangkan, *file* 'solrconfig.xml' sendiri merupakan *file* utama yang ditarik oleh *Solr* sebagai *file* yang memanggil fungsi *requestHandler* dan fungsi lain yang dijalankan oleh *Solr*.

2) Konfigurasi 'data-config.xml'

Di dalam *file* 'data-config.xml', berisi konfigurasi berikut.

Tabel 3. *Syntax* konfigurasi 'data-config.xml'

```
<dataConfig>
  <dataSource type="JdbcDataSource"
    driver="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost:3306/db_booklist"
    user="root" password="" />
  <document>
    <entity name="data_buku" pk="id"
      query="SELECT * FROM data_buku "
      deltaQuery="select id from data_buku where
        last updated >
          '${dataimporter.last_index_time}'"
    >
    <entity name="review_user" pk="buku_id"
      query="SELECT comment,name_user FROM
        review_user WHERE
          buku_id='${data_buku.id}'"
      deltaQuery="SELECT buku_id FROM
        review_user WHERE last updated >
          '${dataimporter.last_index time}'"
      parentDeltaQuery="SELECT id FROM
        data_buku WHERE
          id='${review_user.buku_id}'"
    <field name="review" column="comment" />
    <field name="uname" column="name_user" />
  </entity>
</entity>
</document>
```

Data-config merupakan sebuah *file* konfigurasi untuk memberikan spesifikasi proses yang dilakukan dalam *DataImportHandler*. Pada *syntax* yang ditulis pada tabel 3, dijelaskan bahwa *file* tersebut menggunakan *JDBCDataSource* sebagai jenis sumber *data* untuk *dataimport*. Serta menggunakan *driver database* dengan *url* 'com.mysql.jdbc.Driver' untuk melakukan koneksi ke *database*. Kemudian, menarik *url* yang merupakan alamat untuk koneksi ke *database* yang bernama 'db_booklist'.

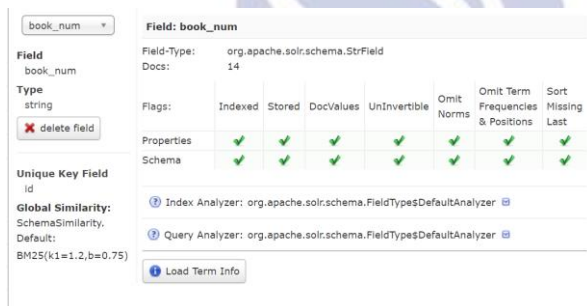
Isi dari konfigurasi berikutnya merupakan konfigurasi yang digunakan dalam *document Solr*. Yaitu, mengenalkan pada *Solr* bahwa nama dari *table* pada *database* yang akan ditarik adalah 'data_buku' yang memiliki *primary key* bernama 'id'. Untuk mengambil data dari semua *field* pada *table* 'data_buku', dituliskan pada *parameter* 'query' pada tabel 3. Kemudian, *parameter* 'deltaQuery' memiliki peran penting dalam berjalannya proses *delta-import*, dimana *syntax* menjelaskan bahwa *Solr* mengambil *data* dari *column* 'id' pada *table* 'data_buku' yang memiliki nilai *timestamp* pada kolom 'last_updated' lebih besar daripada nilai *last index time* yang disimpan oleh *dataImporter* di *Solr*. Kemudian, *table* 'data_buku' memiliki *child* yang bernama *table* 'review_user' yang memiliki *foreign key* pada kolom 'id_buku'. *Parameter query* pada *entiy* 'review_user' memanggil *data* di kolom 'comment', 'name_user' dari *table* 'review_user' yang mana nilai pada kolom 'buku_id' sama dengan *id* pada *table* 'data_buku'. *Parameter deltaQuery* merupakan *syntax delta-import* yang mengambil *data* dari 'buku_id' pada *table* 'review_user' yang memiliki nilai *timestamp* 'last_updated' lebih besar dibanding nilai *last index time* yang disimpan oleh *dataImporter* di *Solr*. *Parameter parentDeltaQuery* merupakan parameter yang menjelaskan

bahwa *id* pada *column* 'buku_id' di *table* 'review_user' merupakan *id* milik 'id' di *table* 'data_buku'. Kemudian, *tag field* menjelaskan bahwa terdapat *field* di *database* dengan nama 'comment' yang di-mapping ke Solr dan diberi nama 'review, dan *field* dengan nama 'name_user' yang di-mapping ke Solr dan diberi nama 'uname'.

Mapping Field

Memetakan *field database* yang dipanggil dalam konfigurasi untuk meletakkan *record*-nya pada *field* di Solr dilakukan dengan menambahkan *field* pada *Schema* di Solr. *Field* yang ditambahkan harus memiliki nama yang sama dengan *field* kolom yang dituliskan pada konfigurasi, serta memiliki tipe *data* 'String'. Dalam penelitian ini menggunakan dua buah *table*, dalam kedua *table* tersebut kolom-kolomnya harus didefinisikan ke dalam Solr agar ketika melakukan *import* dapat dikenali oleh Solr, *field* manakah yang akan diisi oleh *record* yang di-import ke dalam Solr.

Untuk menambahkan *field*, masuk kedalam 'schema' pada *document Solr*, kemudian pilih *add field* dan beri nama yang sesuai pada konfigurasi. Berikut merupakan salah satu deskripsi *field* yang telah dibuat pada *document Solr*.

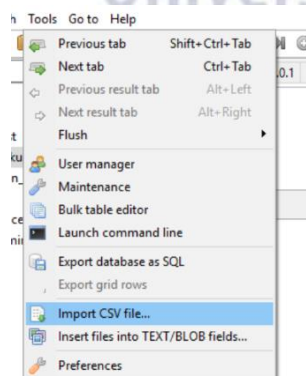


Gambar 12. Field yang telah dibuat

Pengujian Full Import

1) Full-import data awal diMySQL

Full import dilakukan dengan meng-import seluruh *data* yang ada di *database*. Sebelum itu, melakukan *import file CSV* ke dalam *database* terlebih dahulu, sebagai *data awal* pada *database*.



Gambar 13. Load file CSV

```
43 /* Affected rows: 0 Found rows: 1 Warnings: 0 Duration: 0.0001 sec
44 SHOW CREATE DATABASE `db_booklist`;
45 LOAD DATA LOW_PRIORITY LOCAL INFILE 'D:\\book-database\\book-database.csv'
46 /* 6.120 rows imported in 1,250 seconds. */
47 SHOW WARNINGS;
```

Gambar 14. Import CSV berhasil

Pada gambar 14, menunjukkan *error message* menunjukkan bahwa sekitar 1.250 *data* berhasil di-insert ke *database*. Demi menghindari *overload* pada MySQL pada saat *import file CSV*, maka *import* dilakukan secara berangsur-angsur dengan *data* yang dipisah-dipisah. Sehingga *data* tidak di-import seluruhnya secara langsung. Melainkan *data* di-import per 1000 *data* hingga mencapai angka 60,238 *data* di *database*.

Kemudian, dilakukan *full-import* untuk menguji apakah konfigurasi dapat berjalan. Full-import dilakukan karena diperlukan untuk meng-import keseluruhan *data* dalam suatu *database*. Maka untuk memastikan jumlah *data* yang di-import dari *database MySQL* ke Solr sama, seluruh *data* di Solr harus dihapus terlebih dahulu saat melakukan *full-import*. Yaitu, dengan klik *check* pada *checkbox* 'Clean' sebelum mengeksekusi *full-import*, *check* 'Commit' untuk *commit* setelah melakukan *full-import*. *Check* 'Verbose' untuk menampilkan informasi lebih detail. Kemudian, *check* juga pada 'Auto-Refresh Status' agar sistem *refresh status* secara otomatis, lalu klik pada tombol 'Execute' untuk memulai operasi. Berikut merupakan hasil dari *full-import* pertama.

```
Last Update: 11:16:14
Indexing completed. Added/Updated: 60238 documents. Deleted 0 documents.
Requests: 1, Fetched: 60,238 5,476/s, Skipped: 0, Processed: 60,238 5,476/s
Started: less than a minute ago
```

Gambar 15. Full-import 60,238 data.

Full-import berhasil dilakukan dan jumlah *data* yang diproses sama dengan jumlah *data* yang ada di *database*, yaitu 60,238 *data*. Pada gambar 15 tertera bahwa *last update* atau operasi terakhir dijalankan pada pukul 11:16:14, dimana waktu ini akan tersimpan pada *file* 'dataimport.properties' di *document Solr*.

2) Full-import 10 data di MySQL

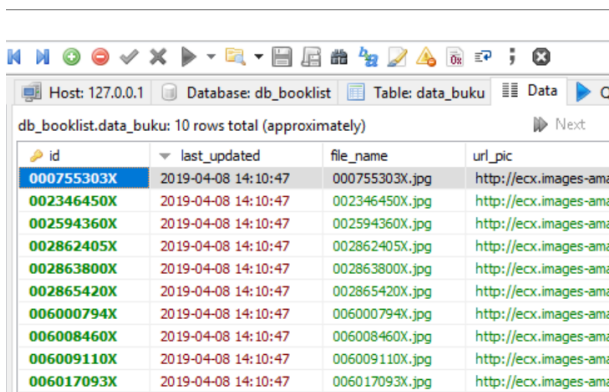
Pada pengujian ini, *data* pada MySQL dihapus seluruhnya pada *table* 'data_buku' untuk mengetahui hasil pengujian *full-import* selanjutnya. Kemudian, Melakukan *insert 10 record data* di MySQL dengan *query* seperti pada gambar 16.

```
Host: 127.0.0.1 Database: db_booklist Table: review_user Data Query
1 insert into data_buku(id,last_updated,file_name,url_pic,book_tit,writer,book_num,book_type)
2 select id,now() as last_updated,file_name,url_pic,book_tit,writer,book_num,book_type
3 from data_buku_all limit 10 offset 0;
```

Gambar 16. Insert 10 record

Query pada gambar 16 merupakan perintah untuk menambah *record* ke dalam *table* 'data_buku' sebanyak 10

record yang diambil dari *table back-up*, yaitu *table 'data_buku_all'*.

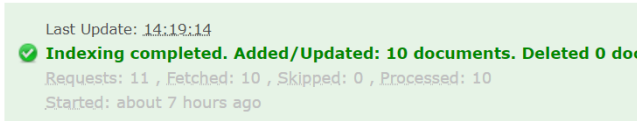


id	last_updated	file_name	url_pic
000755303X	2019-04-08 14:10:47	000755303X.jpg	http://ecx.images-amz
002346450X	2019-04-08 14:10:47	002346450X.jpg	http://ecx.images-amz
002594360X	2019-04-08 14:10:47	002594360X.jpg	http://ecx.images-amz
002862405X	2019-04-08 14:10:47	002862405X.jpg	http://ecx.images-amz
002863800X	2019-04-08 14:10:47	002863800X.jpg	http://ecx.images-amz
002865420X	2019-04-08 14:10:47	002865420X.jpg	http://ecx.images-amz
006000794X	2019-04-08 14:10:47	006000794X.jpg	http://ecx.images-amz
006008460X	2019-04-08 14:10:47	006008460X.jpg	http://ecx.images-amz
006009110X	2019-04-08 14:10:47	006009110X.jpg	http://ecx.images-amz
006017093X	2019-04-08 14:10:47	006017093X.jpg	http://ecx.images-amz

Gambar 17. record di table 'data_buku'

Pada gambar 17 tertera pada kolom 'last_updated' bahwa waktu *timestamp* terbesar di *MySQL* atau waktu terakhir di-updatenya data di *MySQL*, yakni dihari yang sama dengan operasi sebelumnya tanggal **8 April 2019** pukul **14:10:47**.

Selanjutnya melakukan *full-import* ke dalam *Solr*.



Gambar 18. Full-import 10 data sukses.

Solr berhasil *import* seluruh data dari database sebanyak 10 data, serta berikut gambar dari beberapa data yang berhasil di-import pada hasil pencarian di *Solr*.

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 87,
    "params": {
      "q": "",
      "start": "1554707823822"
    }
  },
  "response": {
    "numFound": 10, "start": 0, "docs": [
      {
        "book_type": "Biographies & Memoirs",
        "book_num": "1",
        "file_name": "006000794X.jpg",
        "book_tit": "Ruby Ridge: The Truth and Tragedy of the Randy Weaver Family",
        "id": "006000794X",
        "writer": "Jess Walter",
        "_version_": 1630229171634438144,
        "url_pic": "http://ecx.images-amazon.com/images/I/514W#PJ2eL.jpg"
      }
    ]
  }
}
```

Gambar 19. Hasil pencarian query seluruh data di document *Solr*.

Pada gambar 19 terdapat tulisan "numFound":10, mengartikan jumlah hasil record yang didapat dari hasil pencarian query seluruh data di document *Solr*.

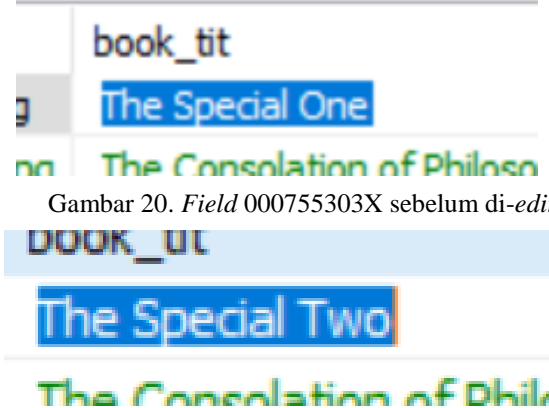
Full-import berhasil dilakukan sesuai permintaan, yaitu 'Processed: 10', yang diartikan *Solr* memproses *full-import* sejumlah 10 record.

Pengujian Delta Import

1) Delta-import 1 data termodifikasi pada 'data_buku'

Pada pengujian ini, terlebih dahulu melakukan perubahan 1 data di *MySQL*. Percobaan yang dilakukan yakni, pada kolom 'book_tit' pada id '000755303X',

mengubah kata 'The Special One' menjadi 'The Special Two' seperti pada gambar 20.

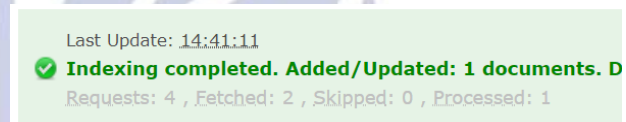


Gambar 20. Field 000755303X sebelum di-edit.

Gambar 21. Field 000755303X setelah di-edit.

Data berhasil diubah dan di-update dengan *timestamp* menunjukkan waktu pukul **14:30:12**.

Selanjutnya, melakukan *delta-import* dengan mencentang 'Verbose' dan 'Commit', serta 'Auto-Refresh Status'. Untuk *delta-import*, 'Clean' tidak dicentang karena jika dicentang akan menghapus data lain yang ada di document *Solr*.



Gambar 22. Delta-import edit 1 data sukses

Delta-import sukses dan memproses secara delta atau sebagian data, yakni 1 data sesuai jumlah data yang di-update. *Solr* mendeteksi adanya perubahan atau adanya data baru yang ada di *MySQL* dengan membandingkan waktu antara *Solr* dengan *MySQL*, yakni waktu terakhir atau terbaru milik *Solr* maupun *MySQL* menjalankan operasi atau meng-update record data. Dimana, *Solr* menyimpan *timestamp* tersebut di dalam document bernama 'dataimport.properties' dan akan ter-update secara *auto* setelah operasi *Solr* dijalankan. Untuk pengujian langkah ini, *delta-import* membandingkan *timestamp* dimana saat terakhir dilakukannya operasi *Solr* yaitu menjalankan *full-import* dengan waktu terakhir pukul **14:10:47**. Sedangkan, di *MySQL* tertera waktu terbaru pukul **14:30:12** yang menunjukkan *timestamp* yang lebih baru atau lebih besar (menurut konfigurasi) dibanding *timestamp* yang ada di *Solr*, sehingga system *Solr* berhasil mendeteksi adanya data baru dan *delta-import* berhasil dieksekusi. Kini, waktu terakhir import yang disimpan *Solr* sebagai last update tertera pada gambar 22 yakni pukul **14:41:11**.

Berikut merupakan hasil pencarian data yang telah diubah, menggunakan pencarian dengan query pada id: **000755303X**.

```
{
  "responseHeader":{
    "status":0,
    "QTime":2,
    "params":{
      "q":"id:000755303X",
      "_:":"1559205225828"}},
  "response":{"numFound":1,"start":0,"docs":[
    {
      "book_type":["Biographies & Memoirs"],
      "book_num":"1",
      "file_name":"000755303X.jpg",
      "book_tit":"The Special Two",
      "id":"000755303X",
      "writer":"Diego Torres",
      "_version_":"1634945200031793153",
      "url_pic":"http://ecx.images-amazon.com/images/I/51AKpivMfFL.
    }
  ]}
}
```

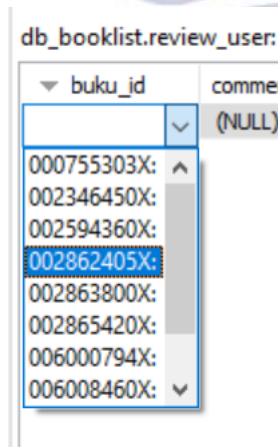
Gambar 23. Hasil pencarian *data record* berdasarkan *id*.

Hasil dapat dilihat pada gambar 23 tepat pada *field* 'book_tit' memiliki *record* berisi 'The Special Two' yang merupakan hasil dari *update data* yang dilakukan di *MySQL*.

2) Delta-import 3 data yang ditambahkan pada *table child*

Selanjutnya, adalah menguji *delta-import* jika dilakukan penambahan *data* baru ke *table child*. Pengujian ini dilakukan untuk mengetahui apakah konfigurasi *Solr* yang telah ditulis dapat bekerja apabila terjadi perubahan bukan di *table* utama, melainkan di *table child*.

Dalam *table child*, dilakukan *insert data* sebanyak 3 *record*.



Gambar 24. Memilih buku yang akan di-review, berdasarkan *id*.

db_booklist.review_user: 3 rows total (approximately)			
buku_id	comment	name_user	last_updated
002862405X	comment1	Rudi	2019-04-08 15:00:50
002346450X	comment2	Hadi	2019-04-08 15:03:09
006008460X	comment3	Mira	2019-04-08 15:04:14

Gambar 25. Insert 3 *comment*.

3 *data* telah di-*insert* dan kolom 'last_updated' ter-*update* dengan *timestamp* terbesar adalah 15:04:14. Kemudian, melakukan *delta-import*.

```
Last Update: 15:11:02
Indexing completed. Added/Updated: 3 documents.
Requests: 11 11/s, Fetched: 12 12/s, Skipped: 0, Process
```

Gambar 26. Delta-import 3 *comment* sukses diproses

Delta-import untuk menambahkan *data* di *table child* sukses diproses, dan dapat terdeteksi oleh *Solr*, maka konfigurasi dapat berjalan dengan baik. Sebelum *delta-import* dieksekusi, *timestamp* terbaru di *Solr* pukul 14:41:11, sedangkan *data* di *MySQL* memiliki *timestamp* yang lebih besar yakni 15:04:14. Dan pada operasi terakhir *Solr* memiliki *timestamp last update* pukul 15:11:02 yang dapat dilihat pada gambar 26.

```
"book_type":["Politics & Social Sciences"],
"book_num":"19",
"file_name":"002346450X.jpg",
"book_tit":"The Consolation of Philosophy",
"id":"002346450X",
"writer":"Richard H. Green",
"uname":["Hadi"],
"review":["comment2"],
"_version_":"1630232702197170176",
"url_pic":"http://ecx.images-amazon.com/images/I/51DCAT3GFFL.jpg"},
{
  "book_type":["Test Preparation"],
  "book_num":"28",
  "file_name":"002862405X.jpg",
  "book_tit":"SSAT & ISEE 7E EAEA",
  "id":"002862405X",
  "writer":"Arco",
  "uname":["Rudi"],
  "review":["comment1"],
  "_version_":"1630232702280007680",
  "url_pic":"http://ecx.images-amazon.com/images/I/71HT405VCJL.gif"},
{
  "book_type":["Humor & Entertainment"],
  "book_num":"13",
  "file_name":"006008460X.jpg",
  "book_tit":"Cheaper by the Dozen",
  "id":"006008460X",
  "writer":"Frank B. Gilbreth",
  "uname":["Mira"],
  "review":["comment3"],
  "_version_":"1630232702280007681",
  "url_pic":"http://ecx.images-amazon.com/images/I/5170U5NLTVL.jpg"}]
```

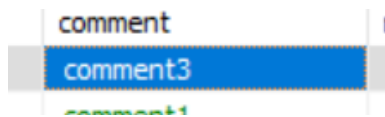
Gambar 27. Hasil *Query* pencarian 3 *comment* yang ditambahkan

Gambar 27 merupakan hasil pencarian dari *data* yang telah di-*import* dengan *delta-import*, dan berhasil ditemukan 3 *data* dengan masing-masing 2 *field* ditambahkan, sesuai dengan *insert* yang dilakukan di *database*, yang kemudian di-*import* ke dalam *Solr*. Hasil *import* diberi tanda dengan *highlight* kuning.

3) Delta-import 1 data di-update pada *table child*

Pengujian ini dilakukan untuk mengetahui apakah bila terjadi perubahan *record data* pada *table child*, *delta-import* dapat mendeteksinya seperti pada saat menambah *record* di *table child*. Sebelumnya, melakukan modifikasi pada *database* terlebih dahulu.

Edit dilakukan pada *column* 'comment' untuk *id* 006008460X. yaitu, mengubah kata 'comment3' menjadi 'comment3 edit'.

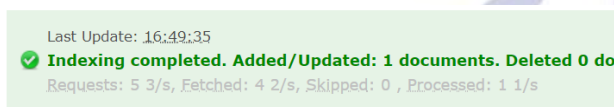


Gambar 28. Comment sebelum di-edit.



Gambar 29. Comment setelah di-edit

Update data telah dilakukan dan data memiliki *timestamp last updated* pukul 16:46:55. Kemudian, melakukan *delta-import* pada Solr.



Gambar 30. Status *delta-import* edit 1 comment.

Delta-import untuk data termodifikasi sukses dilakukan dan data *processed* sebanyak 1 data sesuai banyaknya yang di-edit. Pada operasi *delta-import* sebelumnya Solr memiliki *timestamp last update* pukul 15:11:02, sedangkan MySQL memiliki data yang *last update* nya pukul 16:46:55 yang artinya *timestamp* lebih besar dibanding sebelumnya, sehingga *delta-import* berhasil dilakukan. Kemudian, pada gambar 30 tertera 'Last Update: 16:49:35' yaitu waktu terakhir *DataImport* dilakukan dan disimpan sebagai waktu yang paling baru di Solr.

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 289,
    "params": {
      "q": "id:006008460X",
      "_": "1554717038943"
    }
  },
  "response": {
    "numFound": 1, "start": 0, "docs": [
      {
        "book_type": "Humor & Entertainment",
        "book_num": "13",
        "file_name": "006008460X.jpg",
        "book_tit": "Cheaper by the Dozen",
        "id": "006008460X",
        "writer": "Frank B. Gilbreth",
        "uname": "Mira",
        "review": ["comment3 edit"],
        "_version_": 1630238902466904064,
        "url_pic": "http://ecx.images-amazon.com/images/I/517C"
      }
    ]
  }
}
```

Gambar 31. Hasil pencarian *query* edit 1 comment berdasarkan *id*.

Pada gambar 32 terdapat hasil yang tulisannya ber-highlight kuning, yakni comment yang sudah di-edit dan telah di-import ke Solr menggunakan *delta-import*. Artinya *delta-import* telah berhasil mendeteksi adanya perubahan pada *table child* yaitu 'review_user'.

4) *Delta-import* 1 data yang ditambahkan dengan *id* sama.

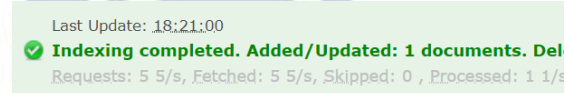
Pengujian ini dilakukan untuk mengetahui apakah Solr dapat mendeteksi adanya *record* baru di salah satu *column* pada *table child* untuk *id* yang sudah digunakan oleh data lain di *table child* yang sama. Dapat dikatakan sebuah *id* buku memiliki 2 buah *review*.

Melakukan *insert* data pada *table* 'review_user' dengan menggunakan *id* yang sama dengan salah satu *id* yang sudah dipakai dalam *table* 'review_user'.

buku_id	comment	name_user	last_updated
006008460X	comment3 edit	Mira	2019-04-08 16:46:55
006008460X	comment4	Suga	2019-04-08 18:18:04
002862405X	comment1	Rudi	2019-04-08 15:00:50
002346450X	comment2	Hadi	2019-04-08 15:03:09

Gambar 32. Insert 1 comment dengan *id* sama.

Data telah ditambah dan tersimpan dengan *timestamp* terbaru, yakni masih hari yang sama pukul 18:18:04. Data yang di-insert adalah kolom 'comment' berisi 'comment4', dan kolom 'name_user' berisi 'Suga'. Kemudian, melakukan *delta-import* untuk *record* tersebut.



Gambar 33. Status *delta-import* 1 data *id* sama

Delta-import berhasil dilakukan, waktu terakhir di *database* pukul 18:18:04, sedangkan waktu di Solr yang terakhir sebelumnya adalah 16:49:35. Kemudian, setelah dilakukan eksekusi *delta-import* untuk data yang *id*-nya sudah digunakan, gambar 34 menunjukkan bahwa *last update* di Solr adalah pukul 18:21:00, maka Solr dapat mendeteksi data *table child* yang memiliki perubahan atau tambahan yang menyimpan waktu lebih baru dibanding waktu terbaru di Solr, walaupun *id* pada data tersebut memiliki *field column* yang sama yang ada di Solr.

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 5,
    "params": {
      "q": "id:006008460X",
      "_": "1554717038943"
    }
  },
  "response": {
    "numFound": 1, "start": 0, "docs": [
      {
        "book_type": "Humor & Entertainment",
        "book_num": "13",
        "file_name": "006008460X.jpg",
        "book_tit": "Cheaper by the Dozen",
        "id": "006008460X",
        "writer": "Frank B. Gilbreth",
        "uname": "Mira",
        "review": ["comment3 edit", "comment4"],
        "_version_": 1630244651934941184,
        "url_pic": "http://ecx.images-amazon.com/images/I/5170U5NLTVL.jpg"
      }
    ]
  }
}
```

Gambar 34. Hasil pencarian *query* untuk data *id* yang sama

Pada gambar 35 terdapat tulisan yang diberi *highlight* kuning, yang artinya *data* yang telah berhasil di-*import* ke *Solr*, yakni ‘Suga’ dan ‘comment4’ hasil dari *data* di *MySQL* yang telah dimodifikasi.

PENUTUP

Simpulan

Kesimpulan yang didapat dari penelitian yang dilakukan berdasarkan rumusan masalah yang ada, yaitu Proses *import* pada *full-import* di *Solr*, yaitu meng-*import* *data* tanpa membandingkan hal satu dengan yang lain dengan menarik seluruh *data* yang ada pada *database* sesuai konfigurasi telah didefinisikan dalam *file* ‘*data-config*’. *Full-import* dikatakan berhasil apabila pada *status Solr* terdapat jumlah *data* yang dieksekusi sesuai dengan jumlah keseluruhan *data* pada *database* yang dipanggil. Sedangkan proses *delta-import*, yaitu menarik atau mengambil sebagian (*delta*) *data* pada *MySQL* dengan mendefinisikannya sebagai *data* baru yang memiliki nilai *last update* lebih besar, dibandingkan dengan nilai *last index time* milik *Solr*, sesuai dengan konfigurasi yang telah dibuat pada *file* ‘*data_config*’. Operasi *delta-import* berhasil dilakukan, apabila pada *status Solr* setelah mengeksekusi *delta-import* terdapat jumlah *data* yang sesuai dengan jumlah *data* yang di-*import*, serta *data*-nya ditemukan pada *document Solr* saat melakukan pencarian.

Saran

Penelitian selanjutnya dapat mengimplementasikan *full-import* atau *delta-import* secara otomatis *scheduler*. Untuk meng-*import* *data* secara terjadwal agar sistem dapat melakukan *import* secara otomatis.

UCAPAN TERIMA KASIH

Ucapan terima kasih kepada **Bapak Ibnu Febry Kurniawan, S.Kom., M.Sc.** selaku Pembimbing untuk penelitian dan pengerjaan artikel ini.

DAFTAR PUSTAKA

- [1] Admin, "Inovasi Informatika Indonesia," 7 October 2016. [Online]. Available: <https://www.i-3.co.id/2016/10/07/index-pada-database/>. [Accessed 9 April 2018].
- [2] H. Rusfandi, "Electronic Theses & Dissertations (ETD) Gajah Mada," 2016. [Online]. Available: http://etd.repository.ugm.ac.id/index.php?mod=penelitian_detail&sub=PenelitianDetail&act=view&typ=html&buku_id=97086&obyek_id=4. [Accessed 9 April 2018].
- [3] A. Kurniawan, "e-Proc," 19 June 2015. [Online]. Available: <https://eproc.lkpp.go.id/news/read/30/aplikasi-e-katalog-gunakan-solr-untuk-optimasi-performa>. [Accessed 12 April 2018].
- [4] T. Tjoen, "Catatan Tomy Tjoen," 22 April 2016. [Online]. Available: <https://tomytjoen.id/2016/04/22/install-config-dan-query-solr-di-debian-8/>. [Accessed 9 April 2018].
- [5] A. Fajri, "Medium," 13 April 2017. [Online]. Available: <https://medium.com/@adnanfajr/tutorial-apache-solr-indexing-mysql-25dc211453b7>. [Accessed 12 April 2018].