Migrasi Basis Data Non-Relasional MongoDB ke MySQL Menggunakan Pentaho Data Integration

by Mahisa Agni Satria Panatagama

Submission date: 20-Nov-2019 10:19PM (UTC+0700)
Submission ID: 1217913213
File name: Makalah_Migrasi_Pentaho_205.pdf (920.06K)
Word count: 2436
Character count: 15427

Migrasi Basis Data Non-Relasional MongoDB ke MySQL Menggunakan Pentaho Data Integration

(Studi kasus: Sistem Pengembangan dan Pembinaan Sumber Daya Manusia Lembaga Kebijakan Pengadaaan Barang / Jasa Pemerintah)

***Abstract*— Dalam menangani sebuah basis data yang memiliki ukuran data yang sangat besar hingga jutaan data, tentunya diperlukan penanganan khusus untuk mengolahnya, terlebih dalam melakukan migrasi dan pemetaan data yang berasal dari berbagai sumber data ke sebuah *Database Management System* (DBMS). Adapun tahapan yang biasa dilakukan dalam melakukan migrasi basis data diantaranya dimulai dari mengumpulkan data, memilah data (*extract)*, transformasi (*transform)*, dan memuat setiap data ke dalam DBMS (*load*). Proses migrasi dan pemetaan basis data dapat dilakukan dengan menggunakan bahasa pemrograman PHP atau Java, namun hal tersebut akan menyulitkan dikarenakan proses yang tidak mudah dan memakan waktu cukup lama. Hal tersebut menjadi lebih sulit dikarenakan saat dilakukan proses migrasi, terkadang ditemukan kondisi data yang tidak seharusnya atau tidak sesuai dengan kondisi DBMS yang dituju, seperti dalam hal tipe data, penamaan atribut, *value*, dan juga struktur data, sehingga perlu digunakan *tool* khusus untuk menanganinya. Salah satu *tool open source* terbaik yang ada di pasar saat ini untuk menangani kasus tersebut adalah Pentaho Data Integration. Pentaho Data Integration adalah sebuah *tools* yang memiliki kemampuan untuk *extract, tranform, dan load* (ETL) data pada *multi platform database*. Adapun di dalam Pentaho terdapat cukup banyak fitur untuk membuat sebuah *workflow control* atau *job*, dan *data workflow* atau *tranformation*. Dengan menggunakan Pentaho Data Integration, diharapkan dapat menunjang kemudahan dalam melakukan pembaharuan sistem dimana data pada sistem lama masih dibutuhkan.**

Keywords—Pentaho Data Integration; database; DBMS; ETL;

I. PENDAHULUAN

Memasuki era teknologi informasi yang berkembang sangat pesat saat ini, sebuah basis data menjadi hal yang sangat penting untuk dikelola baik oleh sebuah organisasi atau perusahaan. Sebuah data dapat menjadi pengaruh perkembangan dan keberlanjutan suatu perusahaan dikarenakan dengan menggunakan data, perusahaan dapat mengevaluasi dan mendapatkan dukungan dalam pengambilan keputusan untuk perkembangan perusahaan. Data tersebut dihasilkan dari sebuah sistem atau aplikasi yang dalam penggunaannya akan terus bertambah setiap harinya, hingga membentuk sebuah kumpulan data yang berukuran cukup besar. Dengan jumlah data yang besar, pemrosesan data tentu akan menjadi hal yang rumit bagi perusahaan, seperti dalam hal analisis data, migrasi data, dan juga pemetaan data.

Sebuah data dalam perkembanganya memiliki karakteristik diantaranya *volume, variety,* dan *velocity* [1]. Ukuran *(volume)* dalam data yang dihasilkan akan terus bertambah dari waktu ke waktu, dikarenakan hampir semua interaksi yang ada di dalam sistem melibatkan data, sehingga dapat menghasilkan ukuran yang sangat besar. Selain ukuran, variasi (*varietas*) yang dihasilkan dalam data juga terdiri dari berbagai jenis dan variasi baik itu dari format *file* seperti teks, gambar, video dan juga dari struktur isi data seperti terstruktur, semi-terstruktur, dan tidak terstruktur. Data tersebut kemudian harus dapat dianalisa dengan cepat dan secara *real-time* untuk menghasilkan informasi, sehingga kecepatan (*velocity*) untuk menyelesaikan proses komputasi data menjadi hal yang perlu diperhatikan [2]. Data-data atau informasi tersebut kemudian dikumpulkan secara sistematik dan saling berhubungan satu dengan yang lainnya sehingga terbentuklah sebuah basis data.

Basis data memiliki konsep atau mekanisme dalam mengorganisasikan dan mengolah data yang ada di *Database Management System* (DBMS), salah satunya yaitu konsep basis data relasional dan basis data non-relasional. Konsep basis data relasional telah menjadi pilihan yang sering digunakan dalam hal pengelolaan data dan penyimpanan informasi pada sebuah aplikasi [3]. Integrasi yang mudah dan pengelolaan relasi yang sederhana membuat konsep relasional *database* menjadi banyak digunakan. Adapun contoh dari *Relational Database Management System* (RDBMS) antara lain MySQL, PostgreSQL, Oracle, SQLite, dll.

Dalam penggunaan basis data relasional, dapat ditemui suatu kebutuhan atau kondisi dimana dalam menyimpan sebuah data tidak seharusnya berbentuk tabel yang terikat pada relasi-relasi yang kaku. Dari hal tersebut maka dapat digunakan konsep basis data non-relasional. Basis data non-relasional menyimpan data dalam bentuk *collection* yang berisikan *document* di dalamnya, serta relasi antar *collection* yang tidak terikat sehingga data yang disimpan dapat menjadi lebih dinamis. Adapun contoh dari DBMS yang menggunakan konsep non-relasional antara lain MongoDB, Casandra, Redis, dll.

Perubahan yang terjadi pada basis data suatu sistem merupakan sesuai yang tidak dapat dihindari, terlebih jika sistem tersebut membutuhkan sebuah fitur baru, perbaikan dari segi performa ataupun *re-write* sistem. Pada contoh kasus makalah ini, dilakukan perubahan *database* pada proyek Pengembangan dan Pembinaan Sumber Daya Manusia Lembaga Kebijakan Barang / Jasa Pemetinah (PPSDM LKPP) dari yang awalnya pada sistem sebelumnya (Aplikasi PPSDM LKPP 1) menggunakan *database* non-relasional MongoDB menuju ke sistem baru (Aplikasi PPSDM LKPP 2) yang menggunakan *database* relasional MySQL. Perubahan ini dilakukan dengan beberapa alasan, yakni: peningkatan fitur dan optimasi sistem lama yang masih ditemukan beberapa kekurangan, performa

pada saat memuat data dari MongoDB berjalan cukup lama saat diintregasikan dengan *database* lainnya, dan kebutuhan akan *query* yang lebih kompleks.

Proses perubahan pada DBMS dilakukan dengan mengimplementasikan *tools* Pentaho Data Integration. Adapun perubahan yang dilakukan yaitu migrasi dan pemetaan basis data dari MongoDB ke MySQL. Pemilihan Pentaho Data Integration sebagai *tools* adalah karena fitur-fitur yang disediakan sangat memungkinkan untuk melakukan proses tersebut, terlebih Pentaho Data Integration merupakan aplikasi *open source* terbaik dalam hal *data integration*.

Makalah ini disusun dengan bagian-bagian bab sebagai berikut: Bab II menyajikan konsep-konsep teori mengenai non-relasional *database*, MongoDB, dan Pentaho Data Integration, kemudian Bab III menyajikan proses analisis, perancangan algoritma, dan metode yang dilakukan. Bab IV menyajikan hasil dari masalah yang dikaji. Adapun kesimpulan dan penyelesaian ditunjukkan pada Bab V.

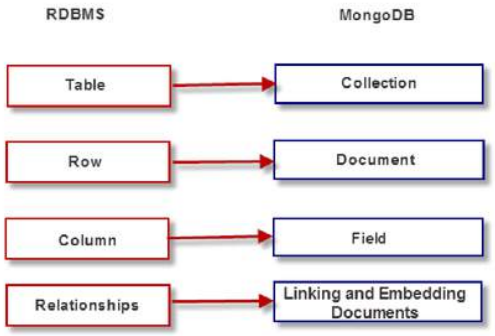
II. LANDASAN TEORI

A. *Non Relasional Database*

Non relasional *database* (NoSQL) merupakan suatu DBMS tidak terstruktur yang memiliki konsep penyimpanan data yang berbeda dari model relasional, seperti penyimpanan data yang dinamis, tidak menggunakan relasi antar tabel, dan juga secara umum memiliki performa pengaksesan data yang lebih cepat dibandingkan *database* relasional [3]. Data yang disimpan di dalam *database* NoSQL disimpan dalam bentuk dokumen JSON. Konsep *database* NoSQL cocok diimplementasikan pada konsep *Big Data* yang mana membutuhkan penyimpanan data (*volume*) yang besar sedangkan sumber daya (server) yang disediakan terbatas, penyimpanan data yang tidak terstruktur (*variety*), dan proses *insert/select* yang tidak rumit dan cepat (*velocity*). Adapun contoh dari *database* yang menggunakan konsep non-relasional antara lain MongoDB, Casandra, Redis, dll.

B. *MongoDB*

MongoDB merupakan sebuah basis data *cross*-platform yang berorientasi pada dokumen yang sangat populer saat ini dimana *database* tersebut memiliki konsep non-relasional dan juga tidak terstruktur. MongoDB menyediakan layanan *database* yang memiliki performa tinggi, ketersediaan tinggi, dan skalabilitas yang mudah. Data di dalam MongoDB disimpan dalam bentuk *collection* yang berisikan kumpulan *document* di dalamnya. Sebuah *collection* merupakan representasi bentuk tabel di dalam *database* relasional begitu juga *document* yang merupakan representasi dari baris dan *field* representasi dari kolom.



Gambar 1 Representasi *database* relasional ke MongoDB

Document di dalam sebuah *collection* tidak terikat pada format yang kaku. Data di dalam sebuah *collection* bersifat dinamis, sehingga setiap *document* tersebut tidak harus berisi *field* yang memiliki struktur data dan jumlah yang sama, seperti pada **Gambar 2**. Sebuah *field* dapat menampung berbagai jenis data (*variety*).

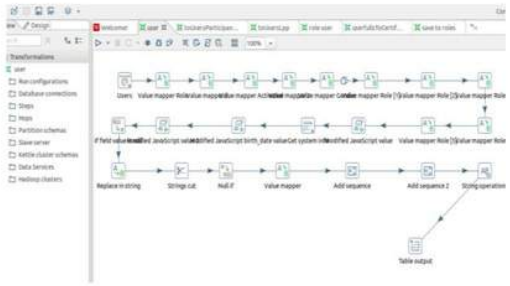
Gambar 2 *Document* di dalam sebuah *collection* MongoDB

C. *Pentaho Data Integration*

Pentaho merupakan sebuah perusahaan yang mempunyai fokus pada produk dan solusi dalam hal *bussiness intelligence*. Pentaho sendiri memiliki beberapa produk (aplikasi) yang memberikan kemudahan dalam mengelola data, analisis data, *data-mining*, dan *reporting*. Salah satu aplikasi dari Pentaho yang dapat digunakan untuk mengelola data dan mampu melakukan proses *extract, transform,* dan *load* (ETL) data adalah Pentaho Data Integration. Aplikasi tersebut dapat digunakan untuk migrasi data, membersihkan data, memuat dari file ke basis data atau sebaliknya dalam volume yang besar [4].

Pentaho Data Integration dapat melakukan proses *transformation* dan *job* dengan cara yang sangat sederhana dan intuitif, sehingga memudahkan pengguna dalam merancang skema dan memeliharanya. Adapun maksud dari *transformation* adalah sekumpulan instruksi untuk mengubah *input* data menjadi sebuah *output* yang telah disesuaikan dengan kebutuhan. Sedangkan *job* adalah kumpulan instruksi yang diberikan untuk menjalankan proses *transformation*. Fitur utama yang dimiliki Pentaho Data Integration yaitu *spoon, pan,* dan

kitchen. *Spoon* merupakan bagian aplikasi atau *interface* untuk membuat rancangan atau instruksi untuk menjalankan proses *transformation*. *Pan* yaitu utilitas untuk menjalankan transformasi dalam tampilan *console*. *Kitchen* yaitu utilitas untuk menjalankan *job* dalam tampilan *console*. Adapun utilitas-utilitas yang digunakan di dalam *job* disebut *step*.



Gambar 3 Tampilan antarmuka Pentaho Data Integration

III. METODOLOGI

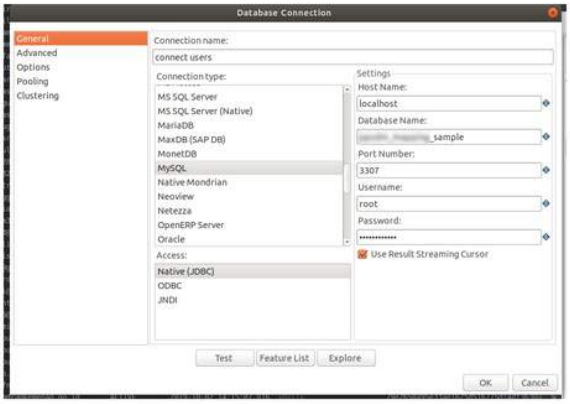
Dalam proses migrasi dan pemetaan *database,* dilakukan beberapa tahapan. Tahapan tersebut dimulai dari proses pengumpulan data yang dibutuhkan dari *datasource* (*extract*), perubahan pada data yang telah di *extract* (*transform*), dan proses penyimpanan atau pemuatan data yang telah ditransformasi (*load*)

Pada makalah ini, data yang akan digunakan adalah data koleksi *users* yang berasal dari aplikasi PPSDM LKPP 1 (MongoDB). Data tersebut kemudian akan dipindahkan ke *database* aplikasi PPSDM LKPP 2 tabel *userfull* (MySQL). Proses migrasi dan pengolahan dilakukan dengan menggunakan *tools* Pentaho Data Integration. Adapun tahapan yang dilakukan adalah sebagai berikut:

A. Pengumpulan Data (Extract)

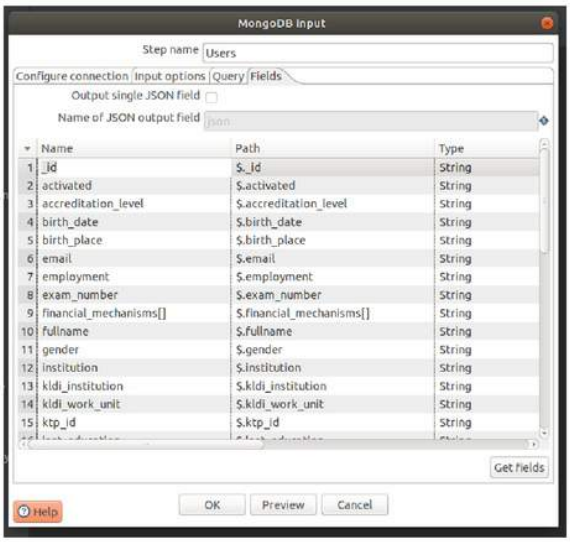
Data yang akan dimigrasi berasal dari *database* PPSDM LKPP 1 yang mana menggunakan DBMS MongoDB. Data tersebut terletak di server, sehingga data tersebut harus diakses secara *remote*. Akses *database* tersebut dapat dilakukan dengan membuat konfigurasi pada *SSH Tunnel* (untuk sistem operasi Linux), dimana *port* pada mesin lokal akan dihubungkan dengan *port* pada server *database remote*, sehingga mesin dapat berkomunikasi dengan baik.

Setelah mesin lokal dapat terbuhung dengan *database* server, kemudian di dalam Pentaho Data Integration, proses pengumpulan dan ekstraksi data dilakukan dengan menggunakan *step* "*MongoDB Input*". Pada *step* tersebut, dilakukan konfigurasi untuk menghubungkan Pentaho ke *database remote* melalui *port tunnel* yang sudah dibuat, seperti ditunjukan pada **Gambar 4**.



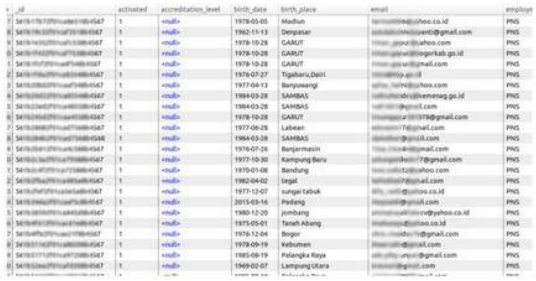
Gambar 4 Konfigurasi *Database Connection*

Ekstraksi dilakukan dengan menyeleksi nama *field* dan *path* JSON yang dibutuhkan atau hapus *field* yang tidak dibutuhkan, seperti pada **Gambar 5**.



Gambar 5 Daftar field yang diambil

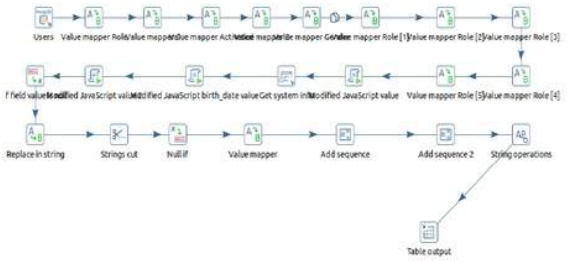
Field yang telah diseleksi kemudian dapat di *preview* **(Gambar 6)** sehingga menampilkan data dari *field* yang telah diseleksi agar memastikan bahwa data tersebut berhasil diambil.



Gambar 6 Data hasil ekstraksi

B. Transformasi Data (Transform)

Data yang telah diekstraksi selanjutnya dilakukan proses transformasi untuk disesuaikan dengan kondisi *database* yang hendak dituju. Transformasi dilakukan agar data tersebut dapat digunakan dengan baik pada aplikasi yang dituju. Pada tahap ini, rancangan, alur dan logika dalam menyusun data dilakukan, seperti yang tunjukan pada **Gambar 7**.



Gambar 7 Skenario *job* dalam menjalankan transformasi

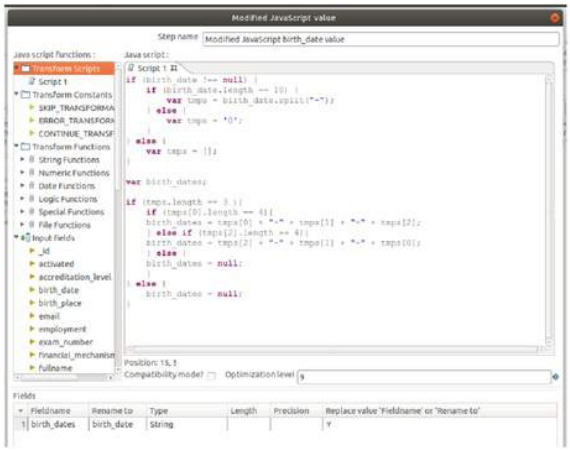
Pada proses transformasi data kasus ini, perubahan yang dilakukan terletak pada nilai data dan tipe data yang digunakan. Perubahan ini diperlukan karena kondisi data dari sumber *database* (MongoDB) tidak cocok atau tidak sesuai apabila langsung dimasukan ke *database* MySQL. Untuk perubahan pada nilai data, salah satu langkah yang digunakan adalah dengan menggunakan *step "Value Mapper"*. Di dalam *step* tersebut kemudian dimasukan nilai data yang hendak diubah pada kolom "*source value"* dan nilai yang hendak dituju pada kolom "*target value"*, seperti pada **Gambar 8**.

Field values:

	Source value	Target value
1	superadmin	Superadmin
2	admin	Admin
3	peserta	Participant
4	lpp	Lpp
5	fasilitator	Admin Fasilitasi
6	assessor	Asesor Akreditasi
7	narasumber	Fasilitator PBJ
8	superadmin_sertifikasi	Superadmin Sertifikasi
9	pengawas	Pengawas Sertifikasi
10	kasi_tata_kelola	Kasi Tata Kelola (Non Jabfung)
11	kasitatakelola	Kasi Tata Kelola (Non Jabfung)
12	admin_penjadwalan	Admin Penjadwalan Ujian Dasar
13	admin_penjadwalan_kompetensi	Admin Penjadwalan Kompetensi
14	admin_sarana	Admin Sarana

Gambar 8 *Step Value Mapper*

Untuk perubahan data yang cukup rumit, seperti mengubah format tanggal yang sebelumnya memiliki format "15-08-1987" ke format tanggal yang dibutuhkan oleh MySQL yaitu "1987-08-15" maka langkah yang dilakukan adalah dengan menggunakan *step "Modified JavaScript"*. *Step* tersebut berfungsi untuk melakukan perubahan nilai data dimana nilai data tersebut tidak dapat diubah dengan *step* yang ada pada Pentaho (kustomisasi). Perubahan dilakukan dengan menggunakan bahasa pemrograman *JavaScript,* seperti pada **Gambar 9**.



Gambar 9 *Modified JavaScript* untuk mengubah format tanggal

Selain menggunakan *step Value Mapper* dan *Modified JavaScript,* proses transformasi pada kasus ini juga menggunakan *step* lain, diantaranya seperti yang ditunjukan pada tabel I.

TABLE I. NAMA DAN FUNGI *STEP* YANG DIGUNAKAN

Nama *step*	Fungsi
If field value null	Pengkondisian jika nilai data berisi <null>
Null if	Pengkondisian null jika nilai data = <data>
Get system info	Mendapatkan variabel dari sistem, seperti zona waktu, waktu, lokasi, dll.
Replace in string	Menggantikan nilai *string* yang ada dengan *string* lain
Strings cut	Memotong jumlah karakter *string*
Add sequence	Menghasilkan angka urutan
String operations	Melakukan operasi string seperti *trim, upper case, lower case, padding, etc.*

C. Memuat Data (Load)

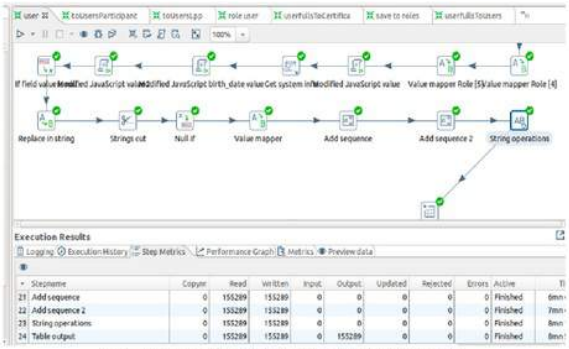
Setelah melakukan proses transformasi, proses terakhir yang dilakukan adalah memuat (*loading*) data yang telah diolah ke *database* yang hendak dituju. *Step* yang dapat digunakan adalah dengan mengunakan *"Table Output"*. Pada kasus ini, lokasi *database* yang akan dituju terletak di server, sehingga perlu menggunakan konfigurasi pada *SSH Tunnel* untuk menghubungkan mesin lokal dengan server *remote*. Setelah berhasil terhubung, data-data yang telah diolah tersebut

kemudian dipetakan sesuai dengan kolom yang tersedia di tabel yang dituju seperti pada **Gambar 10.** Proses pemetaan ini memerlukan analisis mengenai perbandingan nama *field* di *database* lama terhadap nama kolom di *database* baru, karena hal ini berkaitan kemana data dari *database* lama berpindah.

Gambar 10 Pemetaan data dari MongoDB ke MySQL

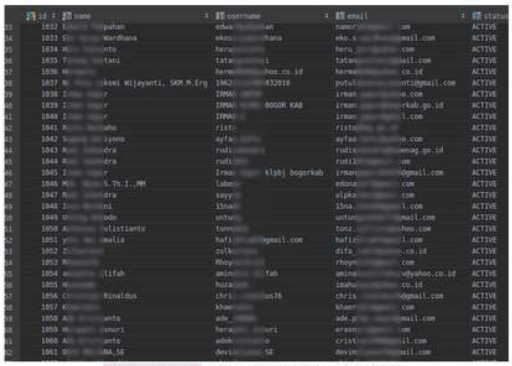
IV. HASIL

Rancangan transformasi *job* yang telah dilakukan sebelumnya kemudian dijalankan di Pentaho Data Integration. Secara ringkas, skema *job* yang dilakukan (**Gambar 11**) menunjukan keberhasilan dalam melakukan migrasi dan pemetaan *database* MongoDB ke *database* MySQL Seluruh data yang dimasukan berhasil diproses oleh semua *step* yang tersedia di dalam *job.* Adapun jumlah data yang berhasil diolah sebanyak 155.289 baris data dengan proses yang memakan waktu selama 8 menit. Waktu tersebut dapat dibilang cukup cepat mengingat data yang diambil berasal dari *database remote,* begitu juga dengan data yang disimpan.



Gambar 11 Skenario job berhasil dijalankan

Data yang telah berhasil dimigrasi kemudian dapat langsung digunakan di aplikasi baru tanpa perlu lagi dilakukan konfigurasi oleh *developer* aplikasi tersebut. Data-data yang tadinya tersimpan secara tidak terstruktur telah berubah menjadi data terstruktur yang mana memiliki jumlah kolom yang kaku dan memiliki relasi antar tabel, seperti yang ditunjukan pada **Gambar 12**.



Gambar 12 Data di database MySQL

V. KESIMPULAN

Kebutuhan dalam pengembangan sebuah sistem aplikasi baru atau *re-write* aplikasi dari aplikasi lama tidak lagi menjadi hal yang menyulitkan, terutama dalam hal menangani perpindahan *database* dari aplikasi lama ke aplikasi baru. Perbedaan dalam hal DBMS, nama data, nilai data, dan tipe data dapat diselesaikan dengan proses migrasi dan pemetaan data. Proses migrasi dan pemetaan data dapat dilakukan dengan menggunakan *tools* Pentaho Data Integration dengan proses dimulai dari pengumpulan data yang dibutuhkan, transformasi data, dan memuatnya ke dalam *database* yang dituju.

Proses migrasi dan pemetaan *database* dengan menggunakan *tools* Pentaho Data Integration dikatakan lebih mudah karena jika dibandingkan dengan menyusun kode (PHP atau Java) untuk melakukan migrasi, maka akan memakan waktu cukup lama dan kesulitan lebih besar dibandingkan dengan menggunakan *tools* Pentaho. Hal tersebut akan menjadi sangat terasa apabila data yang akan diolah berjumlah ribuan data.

Adapun data yang telah berhasil dimigrasi dapat diakses oleh pengguna di aplikasi baru dengan baik sehingga memudahkan *developer* dalam mengembangkan aplikasi baru tersebut, tanpa harus melakukan integrasi dengan *database* aplikasi lama.

REFERENSI

[1] N. Lozada, Jose Arias-Perez, and G. Charry (2019), "*Big data analytics capability and co-innovation: An empirical study*". University of Antiouqia: Research, pp.1-2.

[2] M.I. Baig and Elaheh Yadegaridehkordi (2019), "Big dataadoption: state of the art and research challenges" vol. 6, Uniersity of Malaya: Information Processing & Management, pp. 3.

[3] N. Darsono, "Sistem Basis Data", Medan:Universitas Pelita Harapan.

[4] Wibisono, Y (2014), "Pengantar Pentaho Data Intefraion (Kettle)", http://file.upi.edu/Direktori/FPMIPA/PROD I._ILMU_KOMPUTER/Yudi%20Wibisono/datamining/Modul_Praktikum_Pentaho_Kett le.pdf | diunduh 19 November 2019.

Migrasi Basis Data Non-Relasional MongoDB ke MySQL Menggunakan Pentaho Data Integration

ORIGINALITY REPORT

5%	3%	1%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Universitas Brawijaya Student Paper	1%
2	Meza Silvana, Ricky Akbar, - Derisma. "Pengembangan Model Business Intelligence Manajemen Rumah Sakit untuk Peningkatan Mutu Pelayanan (Studi Kasus : Semen Padang Hospital)", Jurnal Edukasi dan Penelitian Informatika (JEPIN), 2017 Publication	1%
3	cintaimabar.blogspot.com Internet Source	<1%
4	Submitted to Universidad Nacional Abierta y a Distancia, UNAD,UNAD Student Paper	<1%
5	pupangkep.wordpress.com Internet Source	<1%
6	Submitted to Unika Soegijapranata Student Paper	<1%

7	mutiaracintaa.blogspot.com Internet Source	<1%
8	ahlimesin.com Internet Source	<1%
9	ojs.poltek-kediri.ac.id Internet Source	<1%
10	es.scribd.com Internet Source	<1%
11	www.bimbie.com Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		