ISSN: 2598-0256, E-ISSN: 2598-0238

Transformasi Teknologi *Database* Untuk Mendukung Peningkatan Daya Saing Organisasi

Sutedi[1a,*]

[1] Institut Informatika dan Bisnis Darmajaya
[a] sutedi@darmajaya.ac.id

Abstract

Competition between organizations in this digital era is getting tighter and tougher. Organizations must be able to take advantage of information technology to increase productivity and produce superior products. More than that, organizations must also be able to utilize this technology to build business intelligence to increase their competitiveness. The phenomenon of overflowing data that occurs in this digital era must be able to be converted into knowledge that is useful for organizational progress. Therefore, organizations must be able to apply big data technology to be able to manage large, diverse, and very fast amounts of data. One of the big data technology ecosystems that plays an important role in data management is the NoSQL database. This engine has very different characteristics compared to the RDBMS which has been used to manage internal organizational data. Broadly speaking, NoSQL databases are divided into four categories. Two of the most popular are document-oriented NoSQL databases and column-oriented NoSQL databases. In order for big data technology to process internal organizational data, a good database transformation method is needed to migrate data from RDBMS to NoSQL databases. Several studies have been conducted for this. Graph Transforming V2 Algorithm has been successfully developed to effectively transform data from RDBMS to document-oriented NoSQL databases. Multiple Nested Schema has also been developed to transform data from an RDBMS to a column-oriented NoSQL database. However, some special conditions in RDBMS cannot be handled properly. This study proposes the development of adjacency metrics to support the development of Multiple Nested Schema in order to transform data from an RDBMS that contains multiple relationships and recursive relationships into an effective CoNoSQLDB schema.

Keywords: *Transformation*; RDBMS; CoNoSQLDB; *Multiple Relationships*; *Recursive Relationships*; *Business Intelligence.*

Abstrak

Persaingan antar organisasi di era digital ini semakin ketat dan berat. Organisasi harus mampu memanfatkan teknologi informasi untuk meningkatkan produktivitas dan menghasilkan produk-produk unggulannya. Lebih dari itu, organisasi juga harus mampu memanfaatkan teknologi tersebut untuk membangun kecerdasan bisnis untuk meningkatkan daya saingnya. Fenomena luapan data yang terjadi di era digital ini harus dapat ubah menjadi pengetahuan yang berguna bagi kemajuan organisasi. Oleh karena itu organisasi harus mampu menerapkan teknologi *big data* untuk dapat mengelola data dalam jumlah besar, beragam, dan sangat cepat. Salah satu ekosistem teknologi *big data* yang berperan penting dalam pengelolaan data adalah NoSQL *database*. *Engine* ini memiliki karakteristik yang sangat berbeda dibandingkan dengan RDBMS yang selama ini digunakan untuk mengelola data internal organisasi. Secara garis besar, NoSQL *database* terbagi menjadi empat katagori. Dua diantara yang paling popular adalah *document-oriented* NoSQL *database* dan *column-oriented* NoSQL *database*. Agar teknologi *big data* dapat mengolah data internal organisasi, maka diperlukan adanya metode transformasi *database* yang baik untuk memigrasi data dari RDBMS ke NoSQL *database*. Beberapa penelitian telah dilakukan untuk hal tersebut. *Graph Transforming V2 Algorithm* telah berhasil dikembangkan dengan efektif untuk transformasi data dari RDBMS ke *document-oriented* NoSQL *database*. *Multiple Nested Schema* juga telah dikembangkan untuk dapat transformasi data dari RDBMS ke *column-oriented* NoSQL *database*. Namun demikian, beberapa kondisi khusus dalam RDBMS belum dapat ditangani dengan baik. Penelitian ini mengusulkan pengembangan metriks *adjacency* untuk menunjang pengembangan *Multiple Nested Schema* agar dapat melakukan transformasi data dari RDBMS yang mengandung *multiple relationship* dan *recursive relationship* ke dalam skema CoNoSQLDB yang efektif.

Keywords : Transformasi; RDBMS; CoNoSQLDB; *Multiple Relationship; Recursive Relationship; Kecerdasan Bisnis.*

`

1. PENDAHULUAN

Pada era digital ini, banyak organisasi telah memanfaatkan berbagai teknologi informasi untuk mendukung berbagai aktifitas dalam proses bisnisnya. Namun demikian, hal tersebut tidaklah cukup bagi organisasi untuk memenangkan persaingan yang kini semakin ketat saat ini. Selain digunakan untuk meningkatkan produktivitas dan menghasilkan produk-produk unggulan, organisasi juga harus mampu mengimplementasi teknologi informasi guna membangun kecerdasan bisnisnya. Perkembangan dan implementasi teknologi informasi saat ini, mengakibatkan

ISSN: 2598-0256, E-ISSN: 2598-0238

pertumbuhan data yang terus menerus dan terjadi secara *massive* (Lee & Zheng, 2015a). Tsunami data tersebut dapat berupa teks, gambar, foto, video, atau jenis data lainnya (Hsu et al., 2014) yang jumlahnya bahkan telah melampaui batas kemampuan *storage* serta *database engine* yang selama ini biasa digunakan (Rocha et al., 2015). Jika fenomena ini tidak disikapi dengan baik, maka organisasi hanya akan menjadi kaya data namun tetap miskin informasi. Oleh karenanya, organisasi harus mampu mengeksplorasi dan mengekstrak data-data tersebut menjadi *knowledge* yang berguna bagi kemajuan bisnisnya (Chung et al., 2014).

Agar dapat memanfaatkan luapan berbagai jenis data tersebut, organisasi perlu mengimplementasi teknologi *big data*. Teknologi ini mampu mengelola *dataset* super besar (Russom, 2013) serta mampu menampung kombinasi *structured data*, *semi-structured data*, dan *unstructured data* (Goyal et al., 2016; Y. Li & Manoharan, 2013; Saraladevi et al., 2015). Teknologi *big data* minimal harus memenuhi kriteria 3V, yaitu: *volume, variety, serta velocity* (Durham et al., 2014; Gandomi & Haider, 2015; George & Mathew, 2015; Miloslavskaya & Tolstoy, 2016; Srivastava et al., 2015). Dengan demikian teknologi tersebut mampu mengelola data dengan *storage* yang tak terbatas, mengakomodir berbagai jenis data, dan memprosesnya dengan sangat cepat. Salah satu bagian dari ekosistem teknologi *big data* adalah NoSQL (*not only* SQL) *database* (Lee & Zheng, 2015a)(Srivastava et al., 2015)(Bhogal & Choksi, 2015; Gadepally et al., 2015; D. Han & Stroulia, 2013; Hu & Dessloch, 2014; Karnitis & Arnicans, 2015; Lawrence, 2014; Lee & Zheng, 2015b; Mpinda et al., 2015; Nitnaware & Khan, 2015) yang berperan penting dalam pengelolaan data. NoSQL *database* memiliki kemudahan akses, kecepatan, dan skalabilitasnya yang sangat baik (Y. Li & Manoharan, 2013)(Lawrence, 2014)(Huang et al., 2012). Hal ini yang membuat *database engine* tersebut semakin popoler (Kanade et al., 2014)(Mior et al., 2016) dan banyak digunakan untuk mendukung berbagai sistem berbasis *cloud computing* (Lee & Zheng, 2015b)(Lombardo et al., 2012) yang membutuhkan kapasitas *storage* dan pengelolaan data yang besar(Okman et al., 2011), kecepatan *read-write* yang lebih tinggi, serta skema yang lebih fleksibel untuk beradaptasi dengan berbagai jenis data (X. Li et al., 2014). Sistem tersebut sangat efisien dalam penggunaan sumber daya serta ketersediaanya dapat diperoleh dengan biaya yang rendah (Karnitis & Arnicans, 2015). Tingginya kinerja dan skalabilitas pada sistem berbasis *cloud computing* (C. Li, 2010) tidak terlepas dari dukungan NoSQL *database* (J. Han et al., 2011).

Berdasarkan metode penyimpanan datanya, NoSQL *database* terbagi dalam empat katagori, yaitu: *key-value oriented, column oriented, document oriented, dan graph oriented* (Lee & Zheng, 2015b) (Bhogal & Choksi, 2015)(Abiri et al., 2014; Dai, 2019; Hecht & Jablonski, 2011; Kamal et al., 2019; Kaur & Rani, 2013; Mohammadpur & Zeid abadi, 2015). *Database engine* memiliki karakteristik yang sangat berbeda dibandingkan dengan *relational database management system* (RDBMS) yang selama ini biasa digunakan untuk mengelola data internal organisasi (X. Li et al., 2014)(J. Han et al., 2011)(Jia et al., 2016)(Tauro et al., 2013). Karakteristik utama yang membedakan keduanya adalah bahwa NoSQL *database* sejak awal didesain untuk menangani masalah partisi dan replikasi data (Rocha et al., 2015), sedangkan pada RDBMS kemampuan tersebut merupakan kemampuan tambahan yang membutuhkan proses instalasi dan pemeliharaan yang lebih komplek. Selain itu, NoSQL *database* menganut prinsip BASE, yaitu: *basic availability*, *soft state*, dan *eventual consistency* (George & Mathew, 2015)(Chandra, 2015)(Carniel et al., 2012), sedangkan RDBMS mengikuti prinsip ACID, yaitu: *atomicity, consistency, isolation,* dan *durability* (Lee & Zheng, 2015b)(Fatima & Wasnik, 2016; Liyanaarachchi et al., 2016; Nikam et al., 2016). Oleh karena adanya perbedaan karakteristik yang signifikan antara keduanya, serta pentingnya posisi RBDMS sebagai sumber data internal organisasi dalam teknologi *big data*, maka hal tersebut mendorong perlu adanya metode transformasi data yang baik untuk memigrasi data dari RDBMS ke NoSQL *database* (Zhao, Lin, et al., 2014). Metode tersebut sangat dibutuhkan agar organisasi dapat menerapkan teknologi *big data* secara efektif guna meningkatkan daya saingnya (Ghule & Vadali, 2017). Seiring dengan berkembangnya kebutuhan tersebut, maka pengembangan metode transformasi data dari RDBMS ke NoSQL *database* menjadi salah satu isu penting dalam bidang teknologi informasi (Lee & Zheng, 2015b)(Seshagiri et al., 2016).

2. PENELITIAN TERKAIT

Beberapa penelitian yang berkaitan dengan pengembangan metode transformasi data dari RDBMS ke NoSQL *database* telah dilakukan. Sebagian besar diantaranya membahas masalah konversi data ke *document oriented* NoSQL *database* dan *column oriented* NoSQL *database* atau CoNoSQLDB (Hu & Dessloch, 2014). Kedua katagori NoSQL *database* tersebut memiliki kemiripan dengan RDBMS yang lebih dekat dibandingkan katagori *key-value oriented* dan *graph oriented*.

Konversi data ke *document-oriented* NoSQL *database* dibahas dalam beberapa penelitian, antara lain: penelitian Zhao, Lin, *et al.* mengembangkan model konversi data menggunakan *Graph Transforming Algorithm*. Algoritme tersebut menyajikan urutan *nested* yang tepat guna meningkatkan kecepatan *query* pada NoSQL *database* (Zhao, Lin, et al., 2014). Namun demikian, redudansi pada skema NoSQL *database* yang dihasilkan penelitian tersebut masih terlalu besar. Berikutnya, penelitian Hanine *et al.* mengembangkan *prototype* perangkat lunak untuk mengkonversi data ke NoSQL *database* (Hanine et al., 2015). Meskipun prosesnya sudah dilakukan secara otomatis, namun kinerja *query* pada skema NoSQl yang dihasilkan belum optimal. Selanjutnya, penelitian Rocha *et al.* mengusulkan

ISSN: 2598-0256, E-ISSN: 2598-0238

framework NoSQLayer untuk mempermudah proses migrasi data (Rocha et al., 2015). Namun, implementasinya hanya efisien untuk migrasi data yang bervolume besar. Penelitian lainnya yang membahas masalah yang sama dilakukan oleh Potey *et al.* yang memanfaatkan *Amazon cloud services* untuk memigrasi data ke NoSQL *database* (Potey et al., 2015). Hubungan antar *table* RDBMS yang dapat diselesaikan dalam penelitian tersebut belum mengakomodir bentuk-bentuk khusus yang dapat terjadi pada RDBMS.

Konversi data ke *column-oriented* NoSQL *database* dibahas dalam beberapa penelitian, antara lain: penelitian (C. Li yang mengubah RDBMS ke skema Hbase (C. Li, 2010). Solusi konkret untuk menangani situasi *multiple nested* pada penelitian ini belum ada. Topik serupa dilakukan pada penelitian Vajk *et al.* yang mengusulkan algoritme denormalisasi *database* dan mengubah *query* sesuai dengan skema tersebut (Vajk et al., 2013). Jumlah skema yang dihasilkan pada penelitian ini masih terlalu banyak dan kinerja *query*-nya belum optimal. Berikutnya, penelitian Dagar *et al.* mengusulkan lima jenis translasi dari *entity relation model* (ERM) ke *column-bases database schema* (CDBS) (Dagar et al., 2013). Jumlah *table* dan *column family* yang dihasilkan belum optimal dalam menunjang kinerja *query*. Selanjutnya, penelitian Zhao, Li, *et al.* mengusulkan konsep *join key reference relationship*. Meskipun berhasil meningkatkan efisiensi *query* pada kondisi *join query* (Zhao, Li, et al., 2014), namun desain skemanya masih banyak mengandung redundansi data. Penelitian lainnya dilakukan oleh Lee & Zheng yang melakukan denormalisasi menggunakan prinsip desain DDI (*denormalization*, *duplication*, dan *intelligent keys*) untuk mengkonversi data ke NoSQL *database* (Lee & Zheng, 2015a)(Lee & Zheng, 2015b). Lee & Zheng menyempurnakan penelitiannya dengan melakukan otomatisasi proses dalam penentuan *row key* (Lee & Zheng, 2015a). Namun demikian, jumlah *column family* yang terbentuk pada kedua penelitian tersebut belum optimal. Berikutnya, penelitian Mohammadpur & Zeid abadi mengusulkan *Graph Traversing Algorithm* untuk konversi data ke skema CoNoSQLDB (Mohammadpur & Zeid abadi, 2015). Jumlah *table* dan *column family* yang dihasilkan belum menunjang untuk memaksimalkan kinerja *query*-nya.

Berdasarkan hasil penelaahan, diketahui bahwa *Graph Transforming Algorithm* dan *Multiple Nested Schema* menghasilkan skema NoSQL *database* yang lebih baik dibandingkan dengan beberapa metode yang dikembangkan dalam penelitian lainnya. Keduanya menghasilkan peningkatan kinerja *query* secara signifikan. Namun demikian, keduanya masih perlu ditingkatkan untuk mengurangi redundansi pada skema NoSQL-nya. Selain itu, keduanya perlu dikembangkan agar dapat menangani konversi data dari RDBMS yang memiliki variasi hubungan antar tabel yang lebih beragam dan komplek. Pada tahun 2017, Sutedi *et al.* mengembangkan *Graph Transforming Algorithm* agar dapat mentransformasi RDBMS yang mengandung *transitive dependency* secara efektif ke skema *document oriented* NoSQL *database* (Sutedi et al., 2017). Kemudian, pada tahun 2018 Sutedi *et al.* mengembangkan *Multiple Nested Schema* untuk dapat menangani masalah yang sama dengan penelitian sebelumnya sehingga dapat menekan tingkat redudansi data pada skema CoNoSQLDB (Sutedi et al., 2018). Selain itu, penelitian ini juga mengembangkan matriks *adjacency* (Mohammadpur & Zeid abadi, 2015) untuk memudahkan proses identifikasi *leaf node* dan *root* pada *table* RDBMS. Selanjutnya, Sutedi *et al.*pada tahun 2020 menyempurnakan *Graph Transforming Algorithm* menjadi *Graph Transforming V2 Algorithm* sehingga metode tersebut dapat menyelesaikan transformasi *database* dari RDBMS yang membentuk *simple graph* maupun *non-simple graph/multigraph* (*digraph* yang mengandung *multiple edge* dan atau *loop*) dengan efektif ke dalam skema *document oriented* NoSQL *database* (Sutedi et al., 2020).

3. METODE YANG DIUSULKAN

Penelitian ini mengusulkan pengembangan *rules* pada matriks *adjacency* untuk mengidentifikasi kondisi *multiple relationships* pada RDBMS. Hal ini menunjang pengembangan *Multiple Nested Schema* agar dapat mentransformasi RDBMS yang mengandung *mulltiple relationships* ke dalam skema *column-oriented* NoSQL *database*. Kasus *multiple relationships* pada RDBMS terjadi apabila sebuah *table* memiliki lebih dari satu *binary relationship* dengan *table* lainnya. Model matematis dari kondisi tersebut dapat digambarkan dalam Persamaan 1.

$$|A \cap B| \geq 2 \quad (1)$$

Pada Persamaan 1 hubungan antara Himpunan A dan Himpunan B yang diwakili oleh minimal dua buah elemen irisan menunjukkan adanya *multiple foreign keys* di antara kedua *table*. Hal tersebut menunjukkan bahwa di antara kedua himpunan yang merepresentasikan *table* relasional tersebut terdapat *multiple relationships*. Kondisi ini juga dapat terjadi pada RDBMS apabila sebuah *table* memiliki lebih dari satu *unary relationship* dengan dirinya sendiri. Model matematis dari kondisi tersebut dapat digambarkan dalam Persamaan 2.

$$A = \{a1, a2, a3, a4, a1'\} \quad (2)$$

Pada Persamaan 2 terdapat elemen a1’ yang merupakan alias dari elemen a1. Hal tersebut menunjukkan adanya *foreign keys* dalam bentuk atribut alias pada *table* relasional yang membentuk kondisi *recursive relationship* (Silberschatz et al., 2011).

ISSN: 2598-0256, E-ISSN: 2598-0238

Pembentukan matriks *adjacency* pada penelitian ini mengikuti pola yang sudah yang digunakan pada penelitian sebelumnya (Mohammadpur & Zeid abadi, 2015), yaitu dengan memasukkan setiap *table* relasional yang terkait menjadi indeks dimensi baris dan dimensi kolom dalam matriks. Namun demikian, cara pengisian elemen aij matriksnya dimodifikasi dan dikembangkan dengan ketentuan berikut ini.

a. Elemen a_{ij} matriks bernilai "0", jika tidak ada hubungan antara *Table* i dan *Table* j.
b. Elemen a_{ij} matriks bernilai "m", jika ada hubungan antara *Table* i dengan *Table* j, dimana *Table* i berada pada sisi *many*.
c. Elemen a_{ij} matriks bernilai "1", jika ada hubungan *one to one* antara *Table* i dan *Table* j.
d. Elemen a_{ij} matriks dapat diberi nilai jamak (lebih dari satu nilai dipisahkan dengan koma) mengikuti ketentuan di atas apabila antara *Table* i dengan *Table* j memiliki lebih dari satu hubungan.

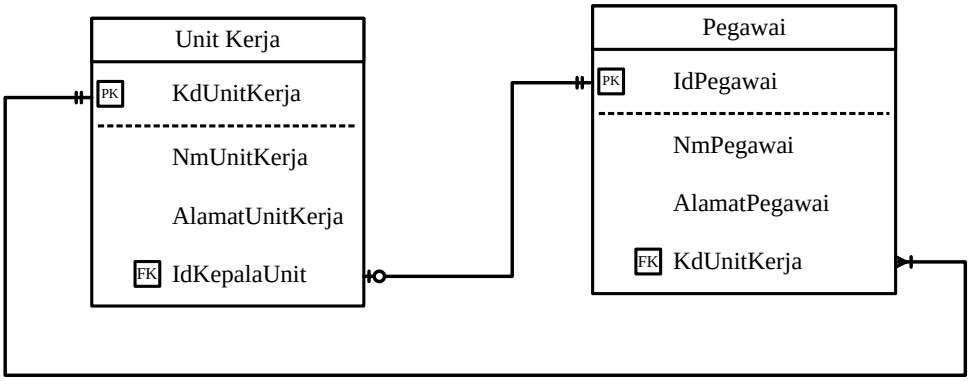
Berikut adalah beberapa *rules* yang dikembangkan agar matriks *adjacency* dapat mengidentifikasi *multiple relatioships* atau *recursive relationship* pada suatu RDBMS.

a. *Table* A dan *Table* B teridentifikasi memiliki *multiple relationships* dua arah, apabila perpotongan dimensi baris dan dimensi kolom dari *Table* A dan *Table* B serta sebaliknya pada matriks *adjacency* bernilai "m' dan "1" atau "m' dan "m" atau "1' dan "1" tergantung kardinalitas datanya.
b. *Table* A dan *Table* B teridentifkasi memiliki *multiple relationships* searah, apabila salah satu perpotongan dimensi baris dan dimensi kolom antara keduanya pada matriks *adjacency* bernilai jamak: "m,m" atau "m,1" atau "1,1" tergantung kardinalitas datanya.
c. Sebuah *table* teridentifikasi memiliki *recursive relationship*, apabila perpotongan dimensi baris dan dimensi kolom antara *table* tersebut dengan dirinya sendiri pada matriks *adjacency* bernilai "m" atau "1" tergantung kardinalitas datanya.

4. HASIL DAN PEMBAHASAN

Berikut adalah pengujian *rules* yang diusulkan ke dalam kasus nyata dalam struktur RDBMS.

a. Struktur RDBMS pada Gambar 1 mencontohkan kondisi *multiple relationships* dua arah yang terjadi di antara dua buah *table*.



Gambar 1. Contoh kasus *multiple relationships* dua arah.

Pada Gambar 1 terdapat dua buah *table*, yaitu: *Table* Unit Kerja dan *Table* Pegawai yang memiliki *multiple relationship* dua arah. *Relationship* pertama merepresentasikan hubungan pegawai dengan unit kerjanya, dimana seorang pegawai hanya menjadi bagian dari satu dan hanya satu unit kerja dan sebuah unit kerja dapat memiliki satu atau beberapa pegawai di dalamnya. Hubungan tersebut dijembatani dengan atribut *foreign key* KdUnitKerja pada *Table* Pegawai. *Relationship* kedua merepresentasikan hubungan antara unit kerja dengan pegawai, dimana sebuah unit kerja dikepalai oleh satu dan hanya satu orang pegawai, sedangkan seorang pegawai dapat menjadi kepala di suatu unit kerja tertentu atau tidak menjadi kepala unit dimana pun. Hubungan tersebut dijembatani dengan atribut *foreign key* IdKepalaUnit pada *Table* Unit Kerja yang merupakan atribut alias dari IdPegawai pada *Table* Pegawai. Matriks *adjacency* dari kasus tersebut disajikan pada Gambar 2.

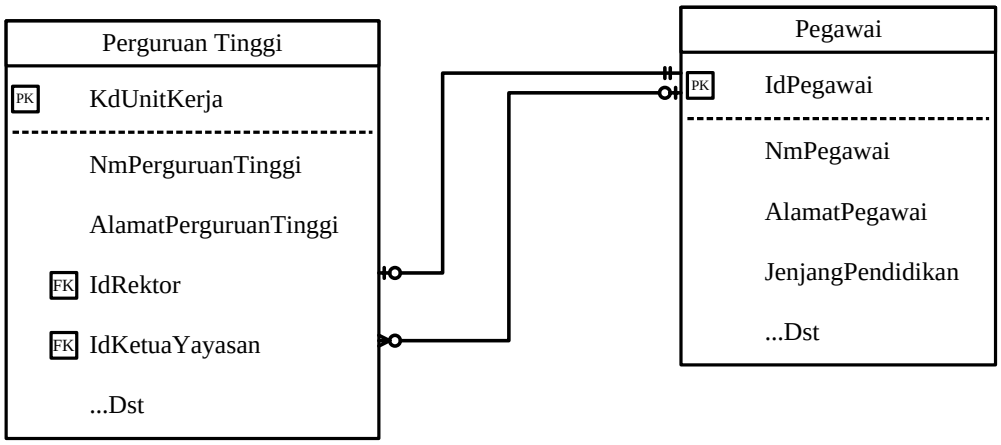
ISSN: 2598-0256, E-ISSN: 2598-0238

		Reference	
		Table Unit Kerja	Table Pegawai
Refer	*Table* Unit Kerja	0	1
	Table Pegawai	m	0

Gambar 2. Matrik *adjacency multiple relationships* dua arah.

Berdasarkan Gambar 2 terlihat bahwa perpotongan dimensi baris dan dimensi kolom antara *Table* Unit Kerja dan *Table* Unit Kerja, serta *Table* Pegawai dan *Table* Pegawai bernilai "0". Hal ini menunjukkan bahwa tidak ada *recursive relationship* pada *table-table* tersebut. Pada satu sisi, perpotongan dimensi baris dan dimensi kolom antara *Table* Unit Kerja dan *Table* Pegawai bernilai "1" yang menunjukkan adanya hubungan *one to one*, dimana pada *Table* Unit Kerja bersifat *optional* sedangkan pada *Table* Pegawai bersifat *mandatory* (*Table* Unit Kerja merujuk ke *Table* Pegawai). Pada sisi lain, perpotongan dimensi baris dan dimensi kolom antara *Table* Pegawai dan *Table* Unit Kerja bernilai "m" yang menunjukkan adanya hubungan dengan kardinalitas *many to one*, dimana *Table* Pegawai berada pada sisi *many* dan *Table* Unit Kerja berada pada sisi *one* (*Table* Pegawai merujuk ke *Table* Unit kerja).

b. Struktur RDBMS pada Gambar 3 mencontohkan kondisi *multiple relationships* searah yang terjadi di antara dua buah *table*.



Gambar 3. Contoh kasus *multiple relationships* searah.

Pada Gambar 3 terdapat dua buah *table*, yaitu: *Table* Perguruan Tinggi dan *Table* Pegawai yang memiliki *multiple relationship* searah. *Relationship* pertama merepresentasikan hubungan perguruan tinggi dengan pegawai, dimana sebuah perguruan tinggi dipimpin oleh satu dan hanya satu orang pegawai yang menjadi rektor, dan seorang pegawai pada satu saat hanya bisa menjadi rektor di satu perguruan tinggi atau tidak menjadi rektor dimana pun. Hubungan tersebut dijembatani dengan atribut *foreign key* IdRektor (atribut alias dari IdPegawai) pada *Table* Perguruan Tinggi. *Relationship* kedua masih merepresentasikan hubungan yang sama, namun dalam kaitannya sebuah perguruan tinggi dapat memiliki seorang pegawai yang menjadi ketua yayasan atau dapat juga tidak memiliki ketua yayasan, sedangkan seorang pegawai dalam satu saat dapat menjadi ketua yayasan untuk beberapa perguruan tinggi atau tidak sama sekali. Hubungan tersebut dijembatani dengan atribut *foreign key* IdKetuaYayasan pada *Table* Perguruan Tinggi yang merupakan atribut alias dari IdPegawai pada *Table* Pegawai. Matriks *adjacency* dari kasus tersebut disajikan pada Gambar 4.

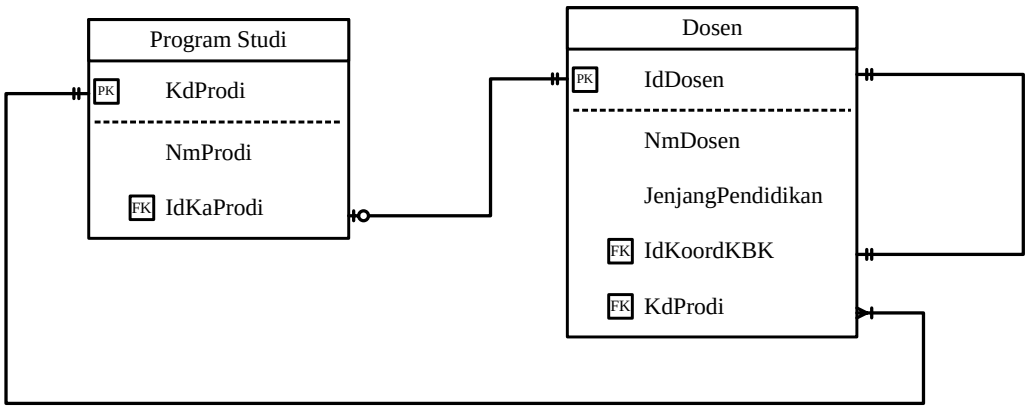
Seminar Nasional Hasil Penelitian dan Pengabdian Masyarakat 2023
Institut Informatika dan Bisnis Darmajaya, 3 Agustus 2023
ISSN: 2598-0256, E-ISSN: 2598-0238

		Reference	
		Table Perguruan Tinggi	Table Pegawai
Refer	*Table* Perguruan Tinggi	0	1,m
	Table Pegawai	0	0

Gambar 4. Matrik *adjacency multiple relationships* searah.

Berdasarkan Gambar 4 terlihat bahwa perpotongan dimensi baris dan dimensi kolom antara *Table* Perguruan Tinggi dan *Table* Perguruan Tinggi, serta *Table* Pegawai dan *Table* Pegawai bernilai "0". Hal ini menunjukkan bahwa tidak ada *recursive relationship* pada *table-table* tersebut. Pada satu sisi, perpotongan dimensi baris dan dimensi kolom antara *Table* Perguruan Tinggi dan *Table* Pegawai bernilai "1,m", sedangkan pada sisi lain, perpotongan dimensi baris dan dimensi kolom antara *Table* Pegawai dan *Table* Tinggi, bernilai "0". Hal ini menunjukkan adanya *multiple relationship* searah, dimana sebuah perguruan tinggi harus memiliki satu dan hanya satu pegawai yang menjadi rektor (*mandatory*), namun seorang pegawai tidak harus menjadi rektor diperguruan tinggi mana pun, sedangkan seorang pegawai dimungkinkan menjadi ketua yayasan di beberapa perguruan tinggi atau tidak sama sekali (*optional*), dan sebuah perguruan tinggi tidak harus memiliki pegawai yang dijadikan ketua yayasan (*optional*).

c. Struktur RDBMS pada Gambar 5 mencontohkan kondisi *recursive relationship* yang terjadi pada suatu *table.*



Gambar 5. Contoh kasus *recursive relationship*.

Pada Gambar 5 terlihat dua buah *Table*, yaitu: *Table* Program Studi dan *Table* Dosen. Dalam kondisi tersebut *Table* Program studi dan *Table* Dosen memiliki *multiple relationship*, dimana sebuah program studi dikepalai oleh satu dan hanya satu orang dosen, sedangkan seorang dosen dapat memimpin satu program studi atau tidak sama sekali. Hubungan tersebut dijembatani dengan adanya atribut *foreign key* IdKaprodi di *Table* Program Studi. Selain itu, sebuah program studi dapat memiliki satu atau beberapa orang dosen di dalamnya. Hubungan tersebut dijembatani dengan adanya atribut *foreign key* IdProdi di *Table* Dosen. Kondisi *recursive relationship* pada Gambar 5 terjadi pada Table Dosen, dimana seorang dosen memiliki seorang koordinator kelompok bidang keilmuan (koordinator KBK) yang juga merupakan seorang dosen. Hubungan tersebut ditandai dengan atribut *foreign key* IdKoordKBK pada *Table* Dosen yang merupakan atribut alias dari IdDosen. Matriks *adjacency* dari kasus tersebut disajikan pada Gambar 6.

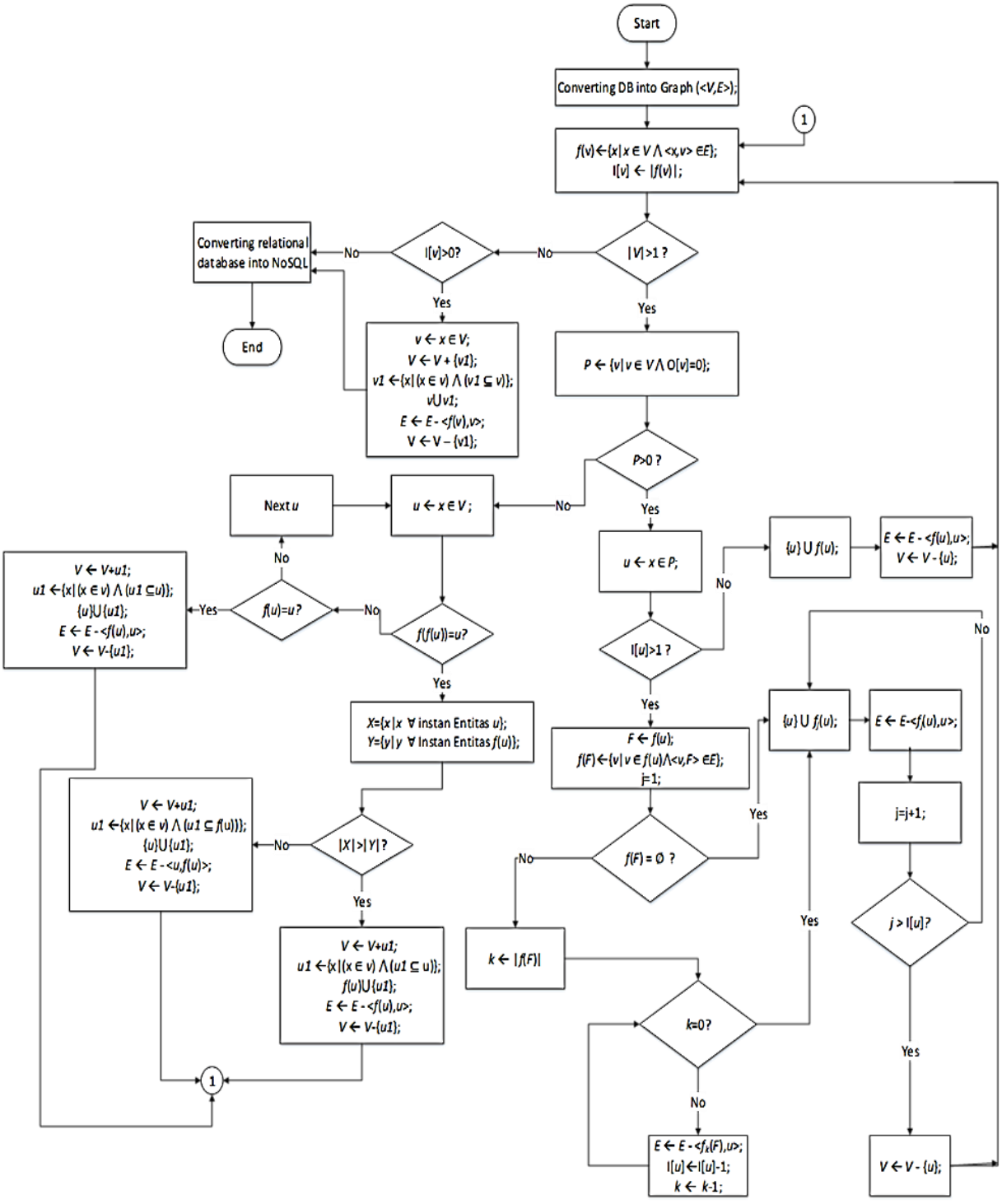
		Reference	
		Table Program Studi	Table Dosen
Refer	*Table* Program Studi	0	1
	Table Dosen	m	1

Gambar 6. Matriks *adjacency recursive relationship*.

ISSN: 2598-0256, E-ISSN: 2598-0238

Berdasarkan Gambar 6, kondisi *recursive relationship* dapat teridentifikasi dari nilai "1" pada perpotongan dimensi baris dan dimensi kolom dari *Table* Dosen dengan dirinya sendiri, sedangkan kondisi multiple relationship antara *Table* Dosen dan *Table* Program Studi teridentifikasi dari nilai "m" pada perpotongan dimensi baris dan dimensi kolom *Table* Dosen dan *Table* Program Studi, serta nilai "1" pada perpotongan dimensi baris dan dimensi kolom *Table* Program Studi dan *Table* Dosen.

Setelah kondisi *multiple relationships* dan *recursive relationships* pada RDBMS teridentifikasi dengan baik, selanjutnya pola penanganan yang digunakan pada *Graph Transforming V2 Algorithm* sebagaimana tersaji pada Gambar 7 dapat diadopsi sebagai dasar pengembangan *Multiple Nested Schema* untuk penanganan kondisi yang serupa.



Gambar 7. *Graph Transforming V2 Algorithm* (Sutedi et al., 2020)

ISSN: 2598-0256, E-ISSN: 2598-0238

Apabia dalam RDBMS teridentifikasi adanya *multiple relationship* maka diidentifikasi jumlah variasi instan entitas yang lebih besar diantara kedua *table* yang terkait, kemudian dilakukan pembetukan *column family* pada skema CoNoSQLDB. Sedangkan jika terjadi kondisi *recursive relationship* pada RDBMS maka dilakukan identifikasi atribut-atribut yang terkait dengan *foreign key* pembentuk *recursive relationship*, kemudian dilanjutkan dengan pembuatan *column family* pada skema CoNoSQLDB.

5. KESIMPULAN

Agar dapat memenangkan persaingan, maka organisasi perlu membangun kecerdasan bisnismya dengan cara ekstraksi *knowledge* dari data internal maupun data eksternal organisasi. Implementasi teknologi *big data* memungkinkan organisasi melakukan hal tersebut. Tingginya kinerja dan skalabilitas teknologi *big data* tidak terlepas dari peran NoSQL *database* yang merupakan salah satu bagian penting teknologi tersebut. Perbedaan karakteritik yang signifikan antara NoSQL *database* dengan RDBMS yang selama ini digunakan dalam pengelolaan data internal organisasi, pada akhirnya mendorong perlunya metode transformasi *database*. Secara umum, *Multiple Nested Schema* dapat digunakan untuk transformasi data dari RDBMS ke dalam skema CoNoSQLDB yang merupakan salah satu tipe NoSQL *database* yang populer. Pengembangan perlu dilakukan agar metode tersebut dapat menangani kondisi-kondisi khusus RDBMS. Matriks *adjacency* dapat dikembangkan untuk mendukung *Multiple Nested Schema* dalam identifikasi RDBMS yang mengandung *multiple relationship* dan *recursive relationship*. Selanjutnya, pola transformasi pada *Graph Transforming V2 Algorithm* dapat diadopsi untuk penyelesaian permasalahan yang serupa.

Beberapa *future works* pada penelitian ini, antara lain: mengidentifikasi kondisi-kondisi khusus lainnya pada RDBMS dan mengembangkan metode *Multiple Nested Schema* untuk menangani kondisi tersebut. Mengembangkan metode transformasi *database* untuk skema *key-value oriented* dan *graph oriented* NoSQl *database*. Mengembangkan teknik *big data mining* untuk membangun kecerdasan bisnis.

UCAPAN TERIMA KASIH

Terima kasih kepada Bapak Teguh Bharata Adji, S.T., M.T., M. Eng., Ph.D. dan Bapak Ir. Noor Akhmad Setiawan, S.T., M.T., Ph.D., IPM. yang telah banyak memberikan arahan dan wawasan tentang penelitian ini serta penulisan publikasi ilmiahnya.

DAFTAR PUSTAKA

Abiri, F., Kahani, M., & Zarinkalam, F. (2014). An Entity Based RDF Indexing Schema Using Hadoop And HBase. *4th International Conference on Computer and Knowledge Engineering (ICCKE)*, 1–6.

Bhogal, J., & Choksi, I. (2015). Handling Big Data using NoSQL. *29th International Conference on Advanced Information Networking and Applications Workshops*, 393–398. https://doi.org/10.1109/WAINA.2015.19

Carniel, A. C., Sá, A. D. A., Brisighello, V. H. P., Ribeiro, M. X., Bueno, R., Ciferri, R. R., & Ciferri, C. D. de A. (2012). Query Processing over Data Warehouse using Relational Databases and NoSQL. *XXXVIII Conferencia Latinoamericana EnInformatica (CLEI)*, 1–9.

Chandra, D. G. (2015). BASE analysis of NoSQL database. *Future Generation Computer Systems*, *52*, 13–21. https://doi.org/10.1016/j.future.2015.05.003

Chung, W. C., Lin, H. P., Chen, S. C., Jiang, M. F., & Chung, Y. C. (2014). JackHare : a framework for SQL to NoSQL translation using MapReduce. *Automated Software Engineering*, *21*(4), 489–508. https://doi.org/10.1007/s10515-013-0135-x

Dagar, M., Mittal, S., & Singh, M. (2013). Conversion from Relational-Based Database to Column-Based Database. *International Journal of Scientific Research in Computer Science*, *1*(1), 29–35.

Dai, J. (2019). SQL to NoSQL : What to do and How. *IOP Conference Series: Earth and Environmental Science*. https://doi.org/10.1088/1755-1315/234/1/012080

Durham, E. E. A., Rosen, A., & Harrison, R. W. (2014). A Model Architecture for Big Data applications using Relational Databases. *IEEE International Conference on Big Data*, 9–16.

Fatima, H., & Wasnik, K. (2016). Comparison of SQL , NoSQL and NewSQL Databases for Internet of Things. *IEEE Bombay Section Symposium (IBSS)*.

Gadepally, V., Bolewski, J., Hook, D., Hutchison, D., Miller, B., & Kepner, J. (2015). Graphulo : Linear Algebra Graph Kernels for NoSQL Databases. *IEEE International Parallel and Distributed Processing Symposium Workshops*, 822–830. https://doi.org/10.1109/IPDPSW.2015.19

Gandomi, A., & Haider, M. (2015). Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, *35*(2), 137–144. https://doi.org/10.1016/j.ijinfomgt.2014.10.007

ISSN: 2598-0256, E-ISSN: 2598-0238

George, K., & Mathew, T. (2015). Big Database Stores A review on various big data datastores. *International Conference on Green Computing and Internet of Things (ICGCIoT)*, 567–573.

Ghule, S., & Vadali, R. (2017). Transformation of SQL system to NoSQL system and performing Data Analytics using SVM. *International Conference on Trends in Electronics and Informatics ICEI 2017*, 883–887.

Goyal, A., Swaminathan, A., Pande, R., & Attar, V. (2016). Cross Platform (RDBMS to NoSQL) Database Validation Tool using Bloom Filter. *FIFTH INTERNATIONAL CONFERENCE ON RECENT TRENDS IN INFORMATION TECHNOLOGY*.

Han, D., & Stroulia, E. (2013). HGrid : A Data Model for Large Geospatial Data Sets in HBase. *IEEE Sixth International Conference on Cloud Computing HGrid*, 910–917. https://doi.org/10.1109/CLOUD.2013.78

Han, J., Haihong, E., Le, G., & Du, J. (2011). Survey on NoSQL database. *6th International Conference on Pervasive Computing and Applications (ICPCA)*, 363–366. https://doi.org/10.1109/ICPCA.2011.6106531

Hanine, M., Bendarag, A., & Boutkhoum, O. (2015). Data Migration Methodology From Relational to NoSQL Database. *International Journal of Computer, Electrical, Automation, Control and Information Enginering*, *9*(12), 2541–2545.

Hecht, R., & Jablonski, S. (2011). NoSQL Evaluation A Use Case Oriented Survey. *International Conference on Cloud and Service Computing*, 336–341.

Hsu, J., Hsu, C., Chen, S., & Chung, Y. (2014). Correlation Aware Technique for SQL to NoSQL Transformation. *IEEE 7th International Conference on Ubi-Media Computing and Workshops Correlation*, 4–7. https://doi.org/10.1109/U-MEDIA.2014.27

Hu, Y., & Dessloch, S. (2014). Extracting deltas from column oriented NoSQL databases for different incremental applications and diverse data targets. *Data & Knowledge Engineering*, *93*, 42–59. https://doi.org/10.1016/j.datak.2014.07.002

Huang, J., Ouyang, X., Jose, J., Wang, H., Luo, M., Subramoni, H., Murthy, C., & Panda, D. K. (2012). High-Performance Design of HBase with RDMA over InfiniBand. *IEEE 26th International Parallel and Distributed Processing Symposium High-Performance*, 774–785. https://doi.org/10.1109/IPDPS.2012.74

Jia, T., Zhao, X., Wang, Z., Gong, D., & Ding, G. (2016). Model Transformation and Data Migration from Relational Database to MongoDB. *IEEE International Congress on Big Data*, 60–67. https://doi.org/10.1109/BigDataCongress.2016.16

Kamal, S. H., Elazhary, H. H., & Hassanein, E. E. (2019). A Qualitative Comparison of NoSQL Data Stores. *International Journal of Advanced Computer Science and Applications*, *10*(2), 330–338.

Kanade, A., Gopal, A., & Kanade, S. (2014). A Study of Normalization and Embedding in MongoDB. *IEEE International Advance Computing Conference(IACC)*, 416–421.

Karnitis, G., & Arnicans, G. (2015). Migration of Relational Database to Document-Oriented Database : Structure Denormalization and Data Transformation. *7th Nternational Conference on Computational Intelligence, Communication Systems and Networks (CICSyN)*, 113–118. https://doi.org/10.1109/CICSyN.2015.30

Kaur, K., & Rani, R. (2013). Modeling and Querying Data in NoSQL Databases. *IEEE International Conference on Big Data*, 1–7.

Lawrence, R. (2014). Integration and Virtualization of Relational SQL and NoSQL Systems including MySQL and MongoDB. *International Conference on Computational Science and Computational Intelligence*, 285–290. https://doi.org/10.1109/CSCI.2014.56

Lee, C. H., & Zheng, Y. L. (2015a). Automatic SQL-to-NoSQL Schema Transformation over the MySQL and HBase Databases. *International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, 426–427.

Lee, C. H., & Zheng, Y. L. (2015b). SQL-to-NoSQL Schema Denormalization and Migration : A Study on Content Management Systems. *IEEE International Conference on Systems, Man, and Cybernetics*, 2022–2026. https://doi.org/10.1109/SMC.2015.353

Li, C. (2010). Transforming Relational Database into HBase : A Case Study. *International Conference on Software Engineering and Service Sciences (ICSESS)*, 683–687.

Li, X., Ma, Z., & Chen, H. (2014). QODM : A Query-Oriented Data Modeling Approach for NoSQL Databases. *IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA)*, 338–345.

Li, Y., & Manoharan, S. (2013). A Performance Comparison of SQL and NoSQL Databases. *IEEE Pacific RIM Conference on Communications, Computers, and Signal Processing*, 15–19. https://doi.org/10.1109/PACRIM.2013.6625441

Liyanaarachchi, G., Kasun, L., Nimesha, M., Lahiru, K., & Karunasena, A. (2016). MigDB – Relational to NoSQL mapper. *IEEE International Conference*.

Lombardo, S., Nitto, E. Di, & Ardagna, D. (2012). Issues in Handling Complex Data Structures with NoSQL databases. *IEEE 14th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, 443–448. https://doi.org/10.1109/SYNASC.2012.59

Miloslavskaya, N., & Tolstoy, A. (2016). Application of Big Data , Fast Data and Data Lake Concepts to Information Security Issues. *4th International Conference on Future Internet of Things and Cloud Workshops*, 148–153. https://doi.org/10.1109/W-FiCloud.2016.41

ISSN: 2598-0256, E-ISSN: 2598-0238

Mior, M. J., Salem, K., Aboulnaga, A., & Liu, R. (2016). NoSE : Schema Design for NoSQL Applications. *IEEE Transactions on Knowledge and Data Engineering*.

Mohammadpur, D., & Zeid abadi, A. (2015). Column-Oriented Storage Optimization in Multi-Table Queries. *Revista Tecnica de La Facultad de Ingenieria Universidad Del Zulia*, *38*(2), 62–68.

Mpinda, S. A. T., Maschietto, L. G., & Bungama, P. A. (2015). From Relational Database to Column-Oriented NoSQL Database : Migration Process. *International Journal of Engineering Research & Technology*, *4*(05), 399–403.

Nikam, P., Patil, T., Hungund, G., Pagar, A., Talegaonkar, A., & Pawar, M. S. (2016). Migrate and Map : A Framework to Access Data from Mysql , Mongodb or Hbase Using Mysql Queries. *IOSR Journal of Computer Engineering*, *18*(3), 13–17. https://doi.org/10.9790/0661-1803041317

Nitnaware, C., & Khan, A. (2015). A Multidimensional Data Storage Model for Location based application on HBase. *2nd International Conference on Innovations in Information Embedded and Communication Systems ICIIECS'15*, 1–5.

Okman, L., Gal-oz, N., Gonen, Y., Gudes, E., & Cassandra, A. (2011). Security Issues in NoSQL Databases. *IEEE International Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11*, 541–547. https://doi.org/10.1109/TrustCom.2011.70

Potey, M., Digrase, M., Deshmukh, G., & Nerkar, M. (2015). Database Migration from Structured Database to non-Structured Database. *International Journal of Computer Applications*, 1–3.

Rocha, L., Vale, F., Cirilo, E., Barbosa, D., & Mouroa, F. (2015). A Framework for Migrating Relational Datasets to NoSQL. *ICCS 2015 International Conference On Computational Science*, *51*, 2593–2602. https://doi.org/10.1016/j.procs.2015.05.367

Russom, P. (2013). Managing Big Data. In *TDWI Research*. TDWI research.

Saraladevi, B., Pazhaniraja, N., Paul, P. V., Basha, M. S. S., & Dhavachelvan, P. (2015). Big Data and Hadoop-A Study in Security Perspective. *Procedia Computer Science*, *50*, 596–601. https://doi.org/10.1016/j.procs.2015.04.091

Seshagiri, V., Vadaga, M. L., Shah, J. J., & Karunakaran, P. (2016). Data Migration Methodology From SQL To Column Oriented Databases (HBase). *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, *5*(11), 2631–2635.

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2011). Database System Concepts. In *Database* (6th ed.). McGrawHill. https://doi.org/10.1145/253671.253760

Srivastava, P. P., Goyal, S., & Smieee, A. K. (2015). Analysis of Various NoSql Database. *IEEE International Conference OnGreen Computing and Internet of Things (ICGCIoT)*, 539–544.

Sutedi, Adji, T. B., & Setiawan, N. A. (2018). Developing Multiple Nested Schema to Reduce Data Redundancy in CONoSQLDB Schema. *Proceedings of ICAITI 2018 - 1st International Conference on Applied Information Technology and Innovation: Toward A New Paradigm for the Design of Assistive Technology in Smart Home Care*, 12–18. https://doi.org/10.1109/ICAITI.2018.8686736

Sutedi, Adji, T. B., & Setiawan, N. A. (2017). Enhanced Graph Transforming Algorithm to Solve Transitive Dependency between Vertices. *IEEE International Conference on Signals and Systems (ICSigSys)*, 250–255.

Sutedi, S., Setiawan, N. A., & Adji, T. B. (2020). Enhanced Graph Transforming V2 Algorithm for Non-Simple Graph in Big Data Pre-Processing. *IEEE Transactions on Knowledge and Data Engineering*, *32*(1). https://doi.org/10.1109/TKDE.2018.2880971

Tauro, C. J. M., Ganesan, N., Easo, A. A., & Mathew, S. (2013). Convergent Replicated Data Structures that Tolerate Eventual Consistency in NoSQL Databases. *IEEE Third International Conference on Advances in Computing and Communications*, 70–75. https://doi.org/10.1109/ICACC.2013.109

Vajk, T., Fehér, P., Fekete, K., & Charaf, H. (2013). Denormalizing Data into Schema-free Databases. *IEEE 4th International Conference on Cognitive Infocommunications*, 747–752.

Zhao, G., Li, L., Li, Z., & Lin, Q. (2014). Multiple Nested Schema of HBase for Migration from SQL. *Ninth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, 338–343. https://doi.org/10.1109/3PGCIC.2014.127

Zhao, G., Lin, Q., Li, L., & Li, Z. (2014). Schema Conversion Model of SQL Database to NoSQL. *Ninth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, 355–362. https://doi.org/10.1109/3PGCIC.2014.137