ISSN : 2085-3688; e-ISSN : 2460-0997

Teknik Migrasi Data Lintas DBMS dengan Menggunakan Metadata

Wahyu Hidayat[1], Muhammad Dwi Aldhi[2], Dahliar Ananda[3]
[1,2,3]Program Studi D3 Manajemen Informatika, Fakultas Ilmu Terapan, Universitas Telkom
[1,2,3]Bandung Technoplex, Jl Telekomunikasi, Bandung 40257, Indonesia
wahyuhidayat@tass.telkomuniversity.ac.id[1], muhammaddwialdhi@students.telkomuniversity.ac.id[2], ananda@tass.telkomuniversity.ac.id[3]

Abstrak - Proses migrasi data biasanya dibutuhkan saat adanya perubahan sistem, format, atau tipe *storage*. Saat ini telah dikenal beberapa teknik dan kakas untuk melakukan migrasi data, misalnya CSV file, ODBC, SQLDump dan sebagainya. Sayangnya tidak semua teknik tersebut dapat diimplementasikan untuk migrasi data antara dua DBMS yang berbeda. Dalam penelitian ini dipaparkan sebuah teknik migrasi data yang dapat digunakan untuk migrasi data lintas DBMS. Teknik migrasi data yang dipaparkan memanfaatkan metadata yang ada di masing-masing DBMS. Proses migrasi data yang dipaparkan di sini melalui tiga tahap yaitu *capture*, *convert* dan *construct*. Sebuah *prototype* dibangun untuk menguji teknik migrasi data ini. Dengan menggunakan *schema* HR dilakukan uji coba migrasi data lintas DBMS antara Oracle dan MySQL. Dengan menggunakan teknik ini, migrasi data *full-schema* membutuhkan waktu rata-rata 20,43 detik dari DBMS Oracle ke MySQL dan 12,96 detik untuk skenario sebaliknya. Adapun untuk migrasi data parsial dibutuhkan waktu rata-rata 5,95 detik dari DBMS Oracle ke MySQL dan 2,19 detik untuk skenario sebaliknya.

Kata kunci – migrasi data; DBMS; metadata; tipe data

Abstract **-** Data migration process is usually required when there is a change in system, format or storage type. Nowadays, we are familiar with some technique and tools to perform data migration, for example CSV file, ODBC, SQLDump etc.Unfortunately, not all of this technique are applicable when it comes to migrating data between two different DBMS. This research give an overview about one particular data migration technique that is applicable for cross DBMS data migration. This technique utilizes metadata that exist in each DBMS. The data migration process described here consist of three stages, namely capture, convert and construct. A prototype is developed to test the proposed data migration technique. Using HR schema, a cross DBMS data migration test between Oracle and MySQL is conducted. Using this technique, full schema data migration requires an average time of 20,43 seconds to migrate data from Oracle to MySQL or an average time of 12,96 seconds for opposite scenario. As for patial data migration, an average time of 5,59 seconds is required to migrate data from Oracle to MySQL or an average time of 2,19 seconds for opposite scenario.

Keywords – data migration; DBMS; metadata; data type

I. PENDAHULUAN

Saat ini ada begitu banyak pilihan Database Management System (DBMS) yang dapat digunakan. Setiap DBMS memiliki keunggulan dan pasarnya masing-masing. Dalam penggunaannya, sangat memungkinkan terjadi proses migrasi data antara DBMS yang satu dengan DBMS yang lain. Proses migrasi data biasanya dibutuhkan saat adanya perubahan sistem, format, atau tipe *storage*. Saat ini telah dikenal beberapa teknik dan kakas untuk melakukan migrasi data, misalnya CSV file, ODBC, SQLDump dan sebagainya. Sayangnya tidak semua teknik tersebut dapat diimplementasikan untuk migrasi data antara dua DBMS yang berbeda.

Format file CSV dinilai sangat universal karena dapat dibaca oleh semua DBMS, namun sayangnya CSV tidak memberikan informasi tentang tipe data dan dinilai tidak praktis jika dihadapkan pada skenario memindahkan data dari beberapa tabel dalam satu *schema*. Oleh karena itu walaupun universal tapi SCV hanya cocok untuk memindahkan data dalam skala tabel. Open Database Connectivity (ODBC) menawarkan kemudahan memindahkan data dalam skala yang lebih besar karena ODBC mampu memindahkan data dari beberapa tabel yang berada dalam satu *schema* secara sekaligus. Sayangnya *driver* ODBC tidak selalu kompatibel dan tidak selalu tersedia. Demikian pula halnya dengan SQLDump yang hanya bisa digunakan untuk migrasi data antar DBMS yang sejenis karena ketergantungannya yang kuat terhadap tipe data.

Oleh karena itu dibutuhkan sebuah teknik migrasi data yang tidak hanya bersifat universal, tidak terpengaruh pada perbedaan tipe data, namun juga dapat digunakan dalam skala yang besar, memiliki kompatibilitas yang baik, tidak bergantung pada

ketersediaan *driver*, dan memberikan kontrol penuh kepada pengguna dalam proses migrasi data.

II. MIGRASI DATA DENGAN METADATA

Ada beberapa definisi tentang migrasi data di berbagai referensi. Dalam [1] disebutkan bahwa Migrasi data adalah teknik atau proses pemindahan data, untuk data yang mengalami perubahan karena alasan tertentu seperti adanya perubahan sistem, dimana sistem baru yang akan diimplementasikan membutuhkan data dari sistem yang lama. Sementara itu dalam [2] disebutkan bahwa migrasi data adalah proses memindahkan data yang mengalami perubahan tipe storage, format data, maupun sistem pengolah data. Biasanya dilakukan dengan bantuan komputer untuk meminimalkan proses manual. Migrasi Data dilakukan karena organisasi melakukan upgrade atau pergantian sistem. Oleh karena itu migrasi data dapat didefinisikan sebagai proses atau teknik pemindahan data yang dilakukan dengan bantuan komputer di mana sistem yang lama mengalami perubahan baik dari tipe storage, format data maupun sistem pengolah data sedemikian rupa sehingga data dari sistem yang lama masih dapat digunakan pada sistem yang baru.

Salah satu tantangan terbesar dalam migrasi data adalah jika migrasi data dilakukan karena perubahan sistem pengolah data atau *database management system* (DBMS). Untuk memenuhi tantangan ini, teknik migrasi data yang paling cocok digunakan adalah dengan memanfaatkan metadata. Metadata adalah data tentang data. Metadata ada di hampir semua DBMS. Biasanya metadata menyimpan informasi tentang data-data yang tersimpan di database tersebut, seperti struktur, tipe maupun lokasi penyimpanan data.

Teknik untuk melakukan migrasi data cukup beragam, misalnya menggunakan SQL Dump [3], Comma Separated Values (CSV) [4], Open Database Connectivity (ODBC) [5], eXtended Markup Language (XML) [6], database reengineering [7] dan sebagainya. Tiap teknik dan metode memiliki kelemahan dan keunggulan masing-masing. Pada Tabel 1 ditunjukkan perbandingan kemampuan antara beberapa teknik migrasi data

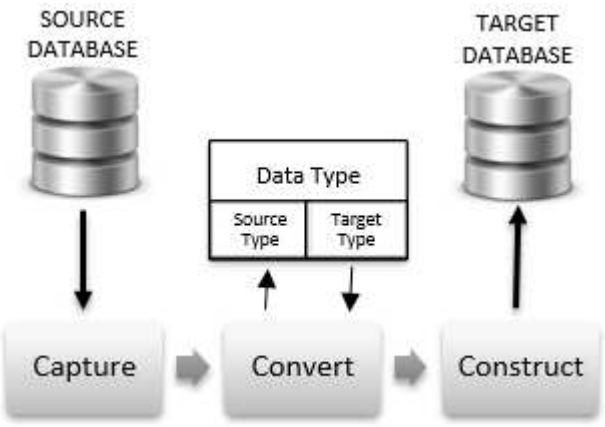
Tabel 1. Perbandingan Teknik Migrasi data

Kemampuan	Teknik Migrasi Data			
	SQl Dump	*CSV*	*ODBC*	*Metadata*
Lintas DBMS		√	√	√
Lihat Skema			√	√
Lihat Tabel			√	√
Memilih Skema	√			√
Memilih Tabel	√	√	√	√
Memilih Kolom				√
Konversi Tipe Data		√	√	√
Log Migrasi				√

Migrasi data lintas DBMS dengan metadata dilakukan melalui tiga tahap yaitu:

1. *Capture*
2. *Convert*
3. *Construct*

Berikut ini adalah ilustrasi dan penjelasan dari tiga tahap pada migrasi data lintas DBMS dengan menggunakan metadata.



Gambar 1. Tahap-Tahap Migrasi Data dengan Metadata

A. *Capture*

Pada tahap *capture*, yang dilakukan adalah membangun koneksi ke database sumber data kemudian melakukan *query* untuk membaca metadata yang ada pada database sumber data. Hasil dari proses *capture* adalah informasi tentang struktur data yang akan dipindahkan, meliputi user pemilik data, daftar table, kolom, tipe data tiap kolom, *constraint* dan *index.*

Untuk mendapatkan informasi ini digunakan lebih dari satu metadata. Masing-masing DBMS juga memiliki metadata yang berbeda-beda. Daftar metadata yang digunakan baik pada DBMS Oracle maupun MySQL terlihat pada Tabel 2 dan Tabel 3 berikut ini.

Tabel 2. Daftar Metadata Yang Digunakan Pada DBMS Oracle

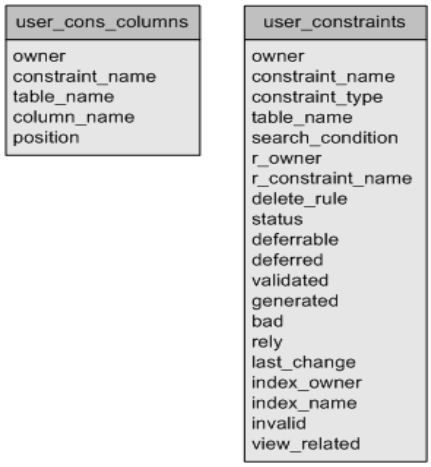
Metadata	Fungsi
DBA_USERS, USER_ROLE_PRIVS	Mengidentifikasi daftar user dan *schema* yang dimiliki beserta privilege masing-masing user
USER _TABLES	Mengidentifikasi daftar tabel yang dimiliki

Metadata	Fungsi
USER_TAB_COLUMNS	Mengidentifikasi daftar kolom dan tipe data, panjang dan presisi nya
USER_CONS_COLUMNS, USER_*CONSTRAINT*S	Mengidentifikasi struktur dan jenis *constraint*
USER_IND_COLUMNS	Mengidentifikasi daftar dan jenis dan lokasi *index*

Tabel 3. Daftar Metadata Yang Digunakan Pada DBMS MySQL

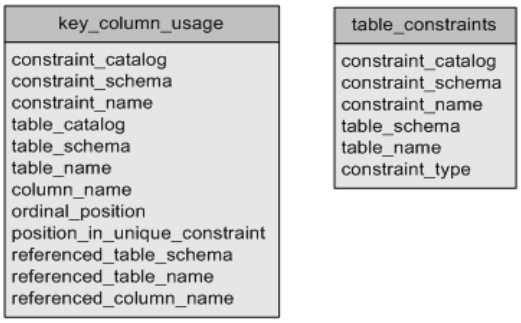
Metadata	Fungsi
*SCHEMA*TA	Mengidentifikasi daftar user dan *schema* yang dimiliki
TABLES	Mengidentifikasi daftar tabel yang dimiliki
COLUMNS	Mengidentifikasi daftar kolom dan tipe data, panjang dan presisi nya
COLUMN_USAGE, TABLE_*CONSTRAINT*	Mengidentifikasi struktur dan jenis *constraint*

Metadata yang saling beririsan juga dimanfaatkan untuk mendapatkan informasi lebih, contohnya untuk mendapatkan informasi tentang *constraint* pada Oracle dibutuhkan dua metadata yang saling beirisan seperti ditunjukkan pada Gambar 2.



Gambar 2. Metadata untuk Identifikasi *Constraint* pada Oracle

Demikian pula halnya dengan metadata pada MySQL, untuk mendapatkan informasi tentang *constraint* digunakan dua metadata yang saling beririsan seperti ditunjukkan pada Gambar 3.



Gambar 3. Metadata untuk Identifikasi *Constraint* pada MySQL

Adapun baris-baris data disalin ke temporary file sebelum diubah tipe datanya pada tahap berikutnya yaitu tahap *convert*.

B. Convert

Pada tahap *convert*, hasil pembacan metadata dari tahap *capture* dianalisis secara otomatis. Setelah itu sebuah script berisi rangkaian perintah Data Definition Language (DDL) juga secara otomatis dibuat untuk menciptakan user, membangun *schema*, membuat tabel dan menyusun keterhubungan antar tabel lengkap dengan *constraint* dan *index*nya di database tujuan. Script ini dibuat berdasarkan hasil pembacaan metadata pada database sumber.

Konversi tipe data dibuat untuk mengakomodir tipe data yang berbeda antara satu DBMS dengan DBMS yang lain. Adapun pemetaan konversi tipe data dilakukan berdasarkan kemiripan karakteristik tipe data seperti yang dijelaskan pada [8] dan dapat dilihat pada Tabel 4 berikut ini.

Tabel 4. Pemetaan Konversi Tipe Data

Tipe Data MySQL	Tipe Data Oracle
BIGINT	NUMBER(19, 0)
INT	NUMBER(10, 0)
MEDIUMINT	NUMBER(7, 0)
SMALLINT	NUMBER(5, 0)
TINYINT	NUMBER(3, 0)
YEAR	NUMBER
DECIMAL	FLOAT (24)
DOUBLE	FLOAT (24)
REAL	FLOAT (24)
FLOAT	FLOAT
DATE	DATE
DATETIME	DATE
TIME	DATE
TIMESTAMP	DATE
TEXT	VARCHAR2

Tipe Data MySQL	Tipe Data Oracle
TINYTEXT	VARCHAR2
VARCHAR	VARCHAR2
CHAR	CHAR

Selain itu sebuah script berisi rangkaian perintah Data Manipulation Language (DML) juga secara otomatis dibuat. Perintah DML ini bertugas memasukkan data ke tabel pada database target sesuia dengan baris-baris data yang tersimpan di temporary file hasil pembacaan data pada proses *capture* yang telah dilakukan sebelumnya.

Script DDL dan DML yang telah dibuat tidak langsung dieksekusi pada tahap *convert*. Namun demikian, baik script DDL maupun DML disusun sedemikian rupa sehingga urutan eksekusi dalam script dapat menjamin keberhasilan pada tahap selanjutnya yaitu tahap *construct*.

C. *Construct*

Pada tahap ini, dibangun sebuah koneksi ke database target, kemudian script DDL yang dihasilkan dari tahap *convert* dieksekusi di database target. Script DDL dieksekusi secara bertahap untuk memastikan agar user, *schema* dan tabel berhasil dibangun pada database target lengkap dengan relasi antar tabel, *constraint* dan *index*nya masing-masing.

Setelah memastikan bahwa script DDL dieksekusi dengan sempurna, barulah script DML yang diperoleh dari tahap *convert* juga diekskusi untuk menyalin data dari temporary file ke database target. Dengan demikian baik struktur database maupun datanya berhasil disalin dari DBMS sumber ke DBMS target.

III. UJI PERFORMA MIGRASI DATA DENGAN METADATA

Sebuah *prototype* dikembangkan untuk melakukan migrasi data lintas DBMS dengan menggunakan metadata sesuai dengan tiga tahap yang telah dijelaskan sebelumnya, yaitu *capture, convert* dan *construct*. Protoype ini dibangun dengan bahasa pemrograman VB.NET dan memiliki kemampuan sebagai berikut.

1. Mengatur konfigurasi koneksi untuk DBMS asal maupun tujuan
2. Menampilkan informasi struktur tabel seperti data kolom, *constraints* dan *index*.
3. Memilih level migrasi data baik secara penuh atau parsial (migrasi spesifik tabel).
4. Melakukan proses migrasi data dan konversi tipe data dari DBMS asal ke DBMS target.
5. Menyajikan informasi hasil proses migrasi data dalam sebuah *log file*

Prototype yang dibangun kemudian diuji dengan serangkaian percobaan migrasi data lintas DBMS antara DBMS Oracle dan MySQL. Pengujian ini bertujuan untuk mengukur efektivitas dan efisiensi teknik migrasi data dngan metadata ditinjau dari waktu eksekusi yang dibutuhkan.

A. *Lingkungan Pengujian*

Prototype diuji pada lingkungan uji berikut ini:

1. *Hardware*
   Processor Corei3 2,53GHz, RAM 2GB dan HDD Space 320GB
2. Software
   Sistem operasi Windows 8.1 Pro 32-bit, Oracle 11g XE dan MySQL 5.0.8

B. *Skenario Pengujian*

Prototype diuji dengan 2 skenario, yaitu migrasi data full *schema* dan migrasi data parsial.

1. Migrasi data full *schema*

   Prototype diuji untuk memindahkan data dari sebuah *schema*, lengkap dengan keterhubungan antar tabel, contraint dan *index*nya masing-masing. Dalam pengujian ini digunakan *schema* "HR" yang merupakan sample *schema* dari DBMS Oracle.

2. Migrasi data parsial

   Prototype diuji untuk memindahkan data dari sebagian *schema*, yaitu dari salah satu tabel, namun lengkap dengan contraint dan *index* yang melekat pada tabel tersebut. Dalam pengujian ini digunakan table "Employees" yang merupakan salah satu tabel dari *schema* "HR".

Detail skenario pengujian migrasi data dapat dilihat pada Tabel 5 berikut ini.

Tabel 5. Detail Skenario Pengujian Migrasi Data

Parameter	Jenis Migrasi Data	
	Full Schema	***Parsial***
Jumlah Tabel	7 (*Schema* HR)	1 (Employees)
Jumlah Baris	215	107
Jumlah Kolom	35	11
Jumlah *Constraint*	19	4
Jumlah *Index*	21	7

Baik pada migrasi data full *schema* maupun migrasi data parsial, *prototype* diuji untuk melakukan migrasi data dari Oracle ke MySQL lalu sebaliknya, dari MySQL ke Oracle. Masing-masing percobaan diulang sebanyak 10 kali dan waktu eksekusi yang dibutuhkan pada tiap percobaan dicatat kemudian dirata-ratakan.

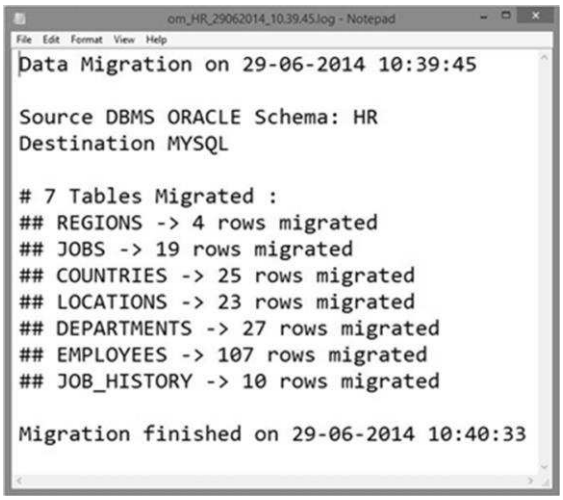
Dengan demikian pada pengujian ini terdapat total 40 percobaan migrasi data dengan rincian seperti yang ditunjukkan pada Tabel 6.

Tabel 6. Rincian Percobaan Pengujian Migrasi Data

Skenario	Jumlah Percobaan	
	Dari Oracle ke MySQL	*Dari MySQl ke Oracle*
Full *Schema*	10	10
Parsial	10	10
TOTAL	**40**	

C. Hasil Pengujian

Prototype yang diuji berhasil melakukan migrasi data baik untuk skenario migrasi data *full schema* maupun migrasi data parsial. Migrasi data lintas DBMS berhasil dilakukan baik dari Oracle ke MySQL maupun sebaliknya. Berikut ini adalah contoh *log file* yang dihasilkan saat tahap pengujian.



Gambar 4. Contoh *Log file*

Log file secara detail menampilkan informasi tentang:

1. DBMS sumber
2. DBMS target
3. Nama *schema* yang dimigrasikan
4. Jumlah dan daftar tabel yang dimigrasikan,
5. Jumlah baris data dari tiap-tiap tabel yang berhasil dimigrasikan.

Selain itu tiap *log file* memiliki catatan waktu dimulainya migrasi data yaitu di awal tahap *capture* sampai waktu selesainya proses migrasi data yaitu pada akhir tahap *construct*. Selisih waktu keduanya dicatat sebagai waktu eksekusi migrasi data. Waktu eksekusi yang dibutuhkan pada tiap skenario dicatat dan kemudian dirata-ratakan.

Berikut ini adalah Tabel 7 yang menampilkan hasil pengujian pada skenario migrasi data full *schema*.

Tabel 7. Waktu Eksekusi Migrasi Data Full Schema

Percobaan ke	Waktu Eksekusi (detik)	
	dari Oracle ke MySQL	*dari MySQl ke Oracle*
1	48,78	15,60
2	24,90	12,25
3	18,65	11,71
4	12,00	11,57
5	10,78	11,43
6	14,16	11,20
7	33,20	11,24
8	15,23	11,71
9	12,26	22,16
10	14,38	10,73
Rata-rata	**20,434**	**12,96**

Adapun hasil pengujian skenario migrasi data parsial dapat dilihat pada Tabel 8 berikut ini.

Tabel 8. Waktu Eksekusi Migrasi Data Parsial

Percobaan ke	Waktu Eksekusi (detik)	
	dari Oracle ke MySQL	*dari MySQl ke Oracle*
1	7,43	3,61
2	4,93	1,47
3	3,81	1,49
4	4,54	2,41
5	4,12	1,94
6	7,47	3,13
7	10,28	1,62
8	5,48	1,57
9	5,21	2,30
10	6,27	2,39
Rata-rata	**5,958**	**2,193**

Hasil pengujian menunjukkan bahwa untuk skenario migrasi data *full-schema* membutuhkan waktu rata-rata 20,43 detik dari DBMS Oracle ke MySQL dan 12,96 detik untuk skenario sebaliknya. Adapun untuk skenario migrasi data parsial dibutuhkan waktu rata-rata 5,95 detik dari DBMS Oracle ke MySQL dan 2,19 detik untuk skenario sebaliknya.

IV. PENUTUP

A. *Kesimpulan*

Teknik migrasi data dengan metadata dapat digunakan untuk melakukan pemindahan data antara dua DBMS yang berbeda tanpa kehilangan *constraint-contraint* pada data. Teknik ini juga dapat digunakan untuk migrasi data secara penuh maupun secara parsial.

Hasil pembacaan *log file* pada saat pengujian menunjukkan bahwa untuk skenario migrasi data *full-schema* membutuhkan waktu rata-rata 20,43 detik dari DBMS Oracle ke MySQL dan 12,96 detik untuk skenario sebaliknya. Adapun untuk skenario migrasi data parsial dibutuhkan waktu rata-rata 5,95 detik dari DBMS Oracle ke MySQL dan 2,19 detik untuk skenario sebaliknya. Hal ini menunjukkan bahwa ditinjau dari sisi waktu yang dibutuhkan, performa teknik migrasi data dengan metadata masih menyediakan banyak ruang untuk perbaikan.

B. *Saran*

Karena teknik migrasi data yang diusulkan baru memanfaatkan beberapa metadata saja maka untuk pengembangan lebih lanjut, perlu diteliti dan dipetakan metadata-metadata lain yang dapat digunakan untuk migrasi data. Selain itu mekanisme konversi tipe data yang lebih baik juga perlu terus dikembangkan, khususnya untuk tipe data yang memiliki kemampuan menampung data berukuran besar seperti tipe data Binary Large Object (BLOB) dan Character Large Oject (CLOB).

DAFTAR PUSTAKA

[1] J. Morris, Practical Data Migration, Swindon: The British Computer Society, 2009.

[2] Trish Rose-Sandler, "Introduction to Data Migration," in Visual Resources Association Conference, Kansas, 2007.

[3] K. Rich, Oracle Database Utilities, 10g Release 2, Oracle, 2005.

[4] Y. Shafranovich, October 2005. Common Format and MIME Type for Comma-Separated-Values[Online]. Available: http://tools.ietf.org/html/rfc4180.

[5] Kingsley Idehen, "Open Database Connectivity," OpenlinkSoftware, Whitepaper 1993. [Online]. http://www.openlinksw.com/info/docs/odbcwhp/tableof/

[6] R. Setiawan and A. Nugroho, “Sistem Pertukaran Data antar Basis Data dengan XML” in Seminar Nasional Aplikasi Teknologi Informasi (SNATI), Yogyakarta, 2005

[7] D.H. Putra and H.A. Wibawa, “Implementasi Interpretive Transformer Approach dalam Migrasi Data sebagai Rangkaian Database Reengineering”, *Jurnal Masyarakat Informatika* Vol. 5 No. 9 p 53-61, 2014.

[8] C. Murray, Oracle SQL Developer Supplementary Information for MySQL Migrations, California: Oracle Corporation, 2008.