Kajian Pendekatan *Binary Log* dalam *Change Data Capture*

Muhammad Febrian Rachmadhan Amri, I Made Sukarsa I Ketut Adi Purnawan
Program Studi Teknologi Informasi Universitas Udayana
Bukit Jimbaran, Bali, Indonesia, telp. (0361) 701806
e-mail : rama.febrian@hotmail.com, sukarsa@ee.unud.ac.id, adipurnawan@unud.ac.id

Abstrak

Era bisnis yang serba online menyebabkan transaksi terjadi sangat cepat sehingga memungkinkan informasi yang tersimpan dalam data warehouse menjadi tidak valid. Data Warehouse digunakan sebagai media repository data yang mempunyai sifat berorientasi subjek, terintegrasi, time-variant, dan bersifat tetap. Data Warehouse dapat dibangun pengelolaannya menjadi bersifat real time dengan pemanfaatan Change Data Capture. Change Data Capture merupakan teknik yang bisa dijadikan sebagai solusi permasalahan untuk membangun real time data warehousing. Pendekatan binary log dalam change data capture dibuat untuk merekam segala aktivitas manipulasi data yang terjadi pada tingkat On-Line Transactional Processing dan dikelola kembali sebelum disimpan ke dalam Data Warehouse (loading process). Pendekatan binary log dapat meningkatkan kualitas pengelolaan data sehingga tercipta informasi yang valid, karena informasi yang tersedia senantiasa diperbarui. Pengujian menunjukkan bahwa pendekatan Binary Log dalam Change Data Capture mampu menghasilkan pengelolaan data secara real time, informasi terkini yang valid, komunikasi antar sistem yang dinamis, dan pengelolaan data tanpa harus kehilangan satupun informasi dari manipulasi data.

Kata kunci: *Change Data Capture, Real Time, Data Warehouse, Database Management System, Binary Log*

Abstract

The online business era causes the form of transactions to occur so quickly that the information stored in the data warehouse becomes invalid. Companies are required to have a strong system, which is a system that is real time in order to be able to perform data loading into the media repository that resides on different hosts in the near-real time. Data Warehouse is used as a media repository of data that has the nature of subject-oriented, integrated, time-variant, and is fixed. Data Warehouse can be built into real time management with the advantages possessed and utilize Change Data Capture. Change Data Capture (CDC) is a technique that can be used as problem solution to build real time data warehousing (RTDW). The binary log approach in change data capture is made to record any data manipulation activity that occurs at the OLTP level and is managed back before being stored into the Data Warehouse (loading process). This can improve the quality of data management so that the creation of the right information, because the information available is always updated. Testing shows that Binary Log approach in Change Data Capture (BinlogCDC) is able to generate real time data management, valid current information, dynamic communication between systems, and data management without losing any information from data manipulation.

Keyword: *Change Data Capture, Real Time, Data Warehouse, Database Management System, Binary Log*

1. Pendahuluan

Era bisnis yang serba *online* menyebabkan segala bentuk transaksi bergerak dengan cepat tanpa mengenal waktu. Proses transaksi yang berlangsung terlalu cepat memungkinkan sumber data yang telah mengalami perubahan belum tercatat ke dalam *data warehouse* saat diperlukan [1]. Data yang tidak *up to date* menyebabkan informasi yang tersimpan dalam *data warehouse* sudah tidak valid untuk dijadikan objek analisis [2]. Manipulasi data yang terjadi

pada sistem dapat direkam dan dimanfaatkan untuk mengatasi permasalahan data yang tidak valid akibat transaksi yang terjadi sangat cepat, sehingga dilakukan suatu kajian mengenai pendekatan *binary log* dalam membangun *change data capture.*

Change data capture (CDC) merupakan suatu teknik yang bisa dijadikan sebagai solusi permasalahan untuk membangun *real time data warehousing* (RTDW). CDC bekerja dengan merekam segala aktivitas yang terjadi pada tabel di level *on-line transaction processing* (OLTP) sebagai sumber data. CDC dimanfaatkan untuk merekam setiap perubahan yang terjadi pada sumber dalam waktu yang nyata, sehingga data yang mengalami perubahan pada sumber data tiba di *data warehouse* pada waktu yang mendekati waktu sebenarnya (*near-real time*) [3]. Mekanisme CDC dikaji untuk diterapkan dalam membangun proses *loading* data ke dalam *data warehouse* secara *real time.* Suatu perusahaan yang tidak menerapkan RTDW memungkinkan tidak menemukan solusi atau inovasi dalam membuat strategi baru yang dapat diterapkan saat pengambilan keputusan [1].

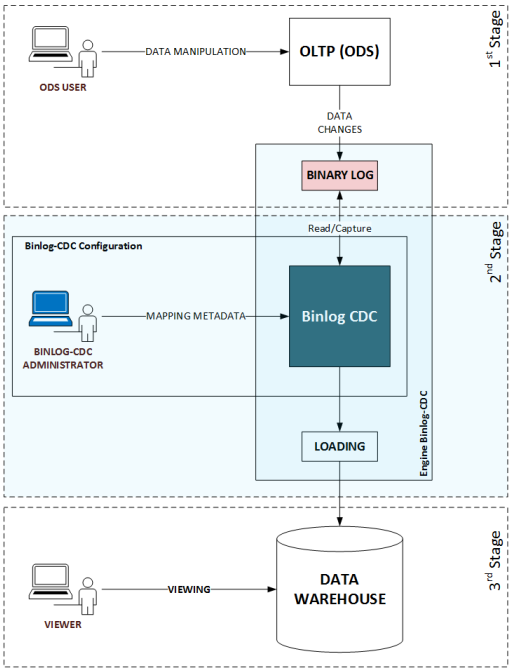
Binary log merupakan suatu *plugin* yang terdapat pada MySQL yang bisa diaktifkan untuk menjadi *log* atau catatan dari segala aktivitas yang terjadi pada *database. Query* digunakan untuk membuka dan membaca informasi yang ada dalam *binary log*. Informasi yang terkandung dalam *binary log* merupakan teks yang terenkripsi menjadi bentuk biner. *Binary log* merupakan catatan riwayat transaksi dan manipulasi yang terjadi pada *database* [4]. Catatan atau *log* dapat dimanfaatkan untuk menghindari hilangnya informasi mengenai manipulasi data karena semua catatan aktivitas yang terjadi pada *database* tercatat secara rinci [5].

2. Metodologi Penelitian

Metode penelitian membahas tentang perancangan dan metode yang digunakan dalam penelitian.

2.1. Gambaran Umum BinlogCDC

Engine BinlogCDC mencatat dan merekam segala manipulasi data yang terjadi pada OLTP. Keluaran utama dari sistem yang dibuat adalah data yang mengalami perubahan. BinlogCDC harus mampu merekam segala informasi penting yang terkandung dalam *binary log.* Gambaran umum sistem secara lengkap ditunjukkan pada Gambar 1.



Gambar 1. Gambaran Umum BinlogCDC

Administrator BinlogCDC memegang peranan penting dalam sistem yang dirancang seperti yang digambarkan oleh Gambar 1. Administrator BinlogCDC harus mengerti tentang

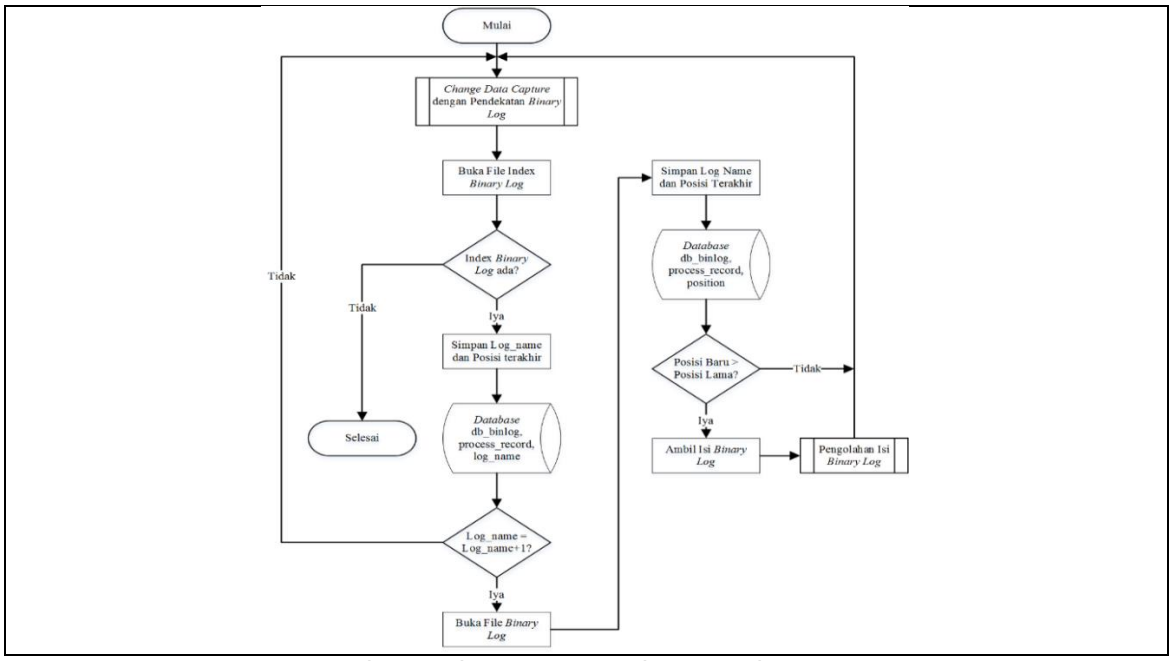
rancangan dimensi fakta dalam *Data Warehouse* serta memiliki pemahaman untuk memetakan data sumber (ODS) dari OLTP dengan tabel dimensi atau fakta yang menjadi tujuannya (*Data Warehouse*). Semua pemahaman tentang *Data Warehousing* diperlukan agar proses *loading* data dapat dilakukan dengan baik.

2.2. Kajian Proses BinlogCDC

BinlogCDC memiliki empat tahapan dalam melakukan *capture* terhadap perubahan data yang terjadi, berikut merupakan urutan tahapan proses BinlogCDC.

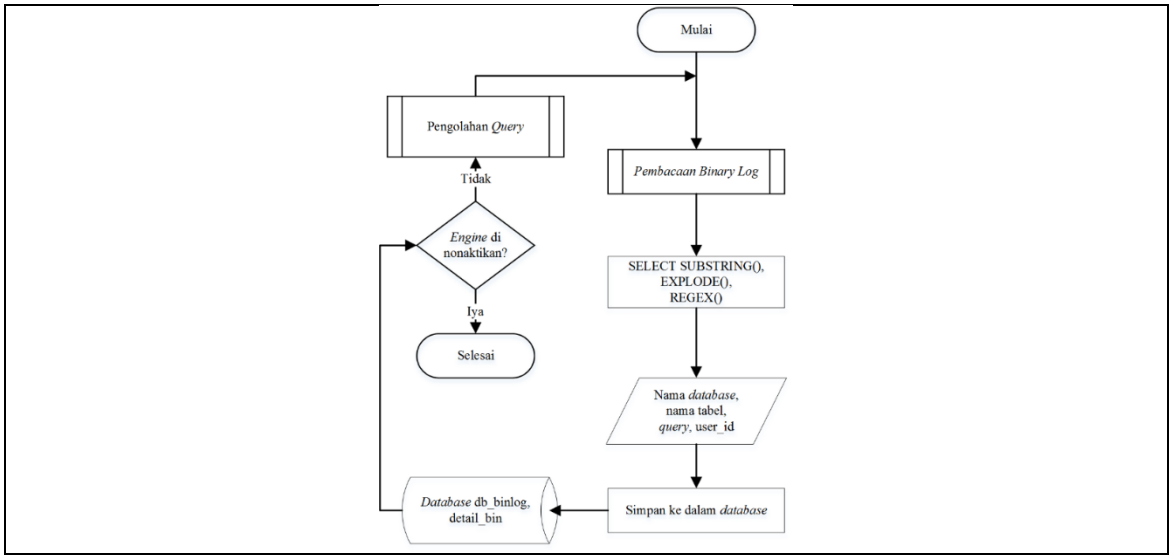
1. Proses Pembacaan *Binary Log*
2. Proses Pengolahan Isi *Binary Log*
3. Proses Pengolahan *Query*
4. Proses *Loading Data*

Proses pembacaan *file binary log* digambarkan dengan diagram alir seperti yang terlihat pada Gambar 2.



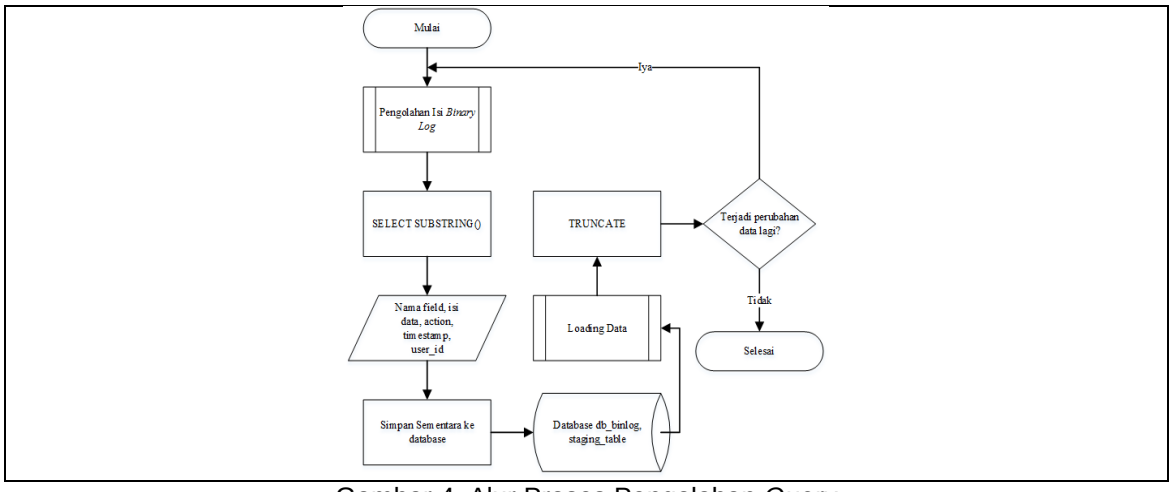
Gambar 2. Alur Proses Pembacaan *Binary Log*

BinlogCDC berfungsi untuk membaca dan membuka *file index* dari *binary log*. Sistem menyimpan satu per satu nama *file index* beserta posisi terakhir yang ada di dalamnya. Proses pembacaan dilakukan secara berulang hingga sistem menemukan posisi yang bertambah. Posisi yang bertambah menunjukkan adanya manipulasi atau perubahan data. Manipulasi data menjadi pemicu sistem untuk mengambil informasi yang terkandung di dalam *binary log* dan melakukan proses pengolahan isi dari *binary log.*



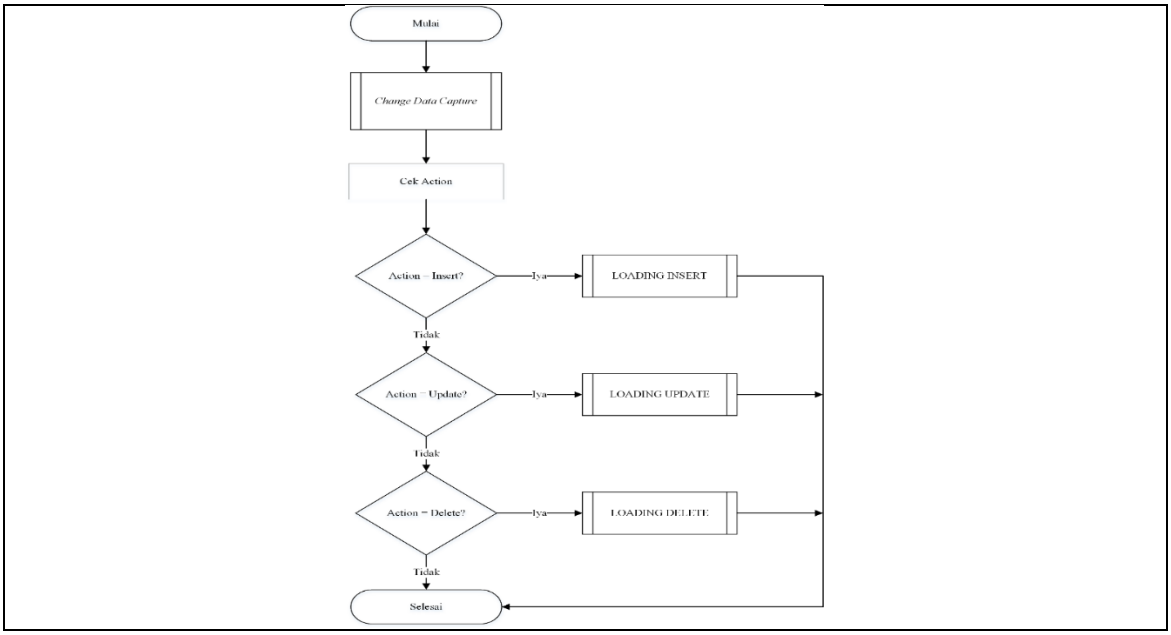
Gambar 3. Alur Proses Pengolahan Isi *Binary Log*

Pengolahan isi dari *binary log* memiliki *output* informasi yang terkandung dalam *binary log* seperti *query*, nama *file*, posisi dimana *query* dieksekusi, dan beberapa informasi lainnya seperti yang terlihat pada Gambar 3. Fungsi yang digunakan dalam mengolah isi yang ada dalam *binary log* adalah fungsi *string* seperti `SELECT SUBSTRING()`, `EXPLODE()`, serta `REGEX()`. Fungsi *string* digunakan untuk memecah *query* yang ada di dalam *binary log.* Tujuan dilakukan pemecahan yaitu untuk mendapatkan nama *database* dan tabel yang mengalami proses manipulasi. Pemecahan *query* juga digunakan untuk mendapatkan *action* atau jenis proses yang dieksekusi, misalnya proses *insert, update,* atau *delete*. Informasi yang didapatkan kemudian disimpan ke dalam *database* sistem.



Gambar 4. Alur Proses Pengolahan *Query*

Proses pada Gambar 4 dinamakan proses pengolahan *query.* Informasi dari isi *binary log* yang telah tersimpan, kemudian diolah kembali dengan menggunakan fungsi `SELECT SUBSTRING()`. Fungsi ini bekerja pada hasil pengolahan *query* yang sebelumnya dilakukan, sehingga didapatkan informasi mengenai *field* beserta *value*. Proses ini menghasilkan informasi yang siap untuk dijadikan acuan dalam melakukan pemetaan metadata sebelum dilanjutkan dengan proses pengiriman data ke dalam *data warehouse* (*loading*).



Gambar 5. Alur Proses *Loading Data*

Informasi dari hasil pengolahan akhir tersimpan secara sementara dalam sebuah tabel *staging* dan dihapus kembali ketika proses *loading* telah selesai dilakukan seperti yang terlihat pada Gambar 5. Tabel *staging* menyimpan informasi mengenai jenis proses apa yang terjadi (*action*). Terdapat tiga jenis *action* yaitu *insert*, *update*, dan *delete,* yang masing-masing memiliki mekanisme tersendiri dalam menangani proses *loading* ke dalam tujuannya.

2.3. Kajian Proses dalam Binary Log

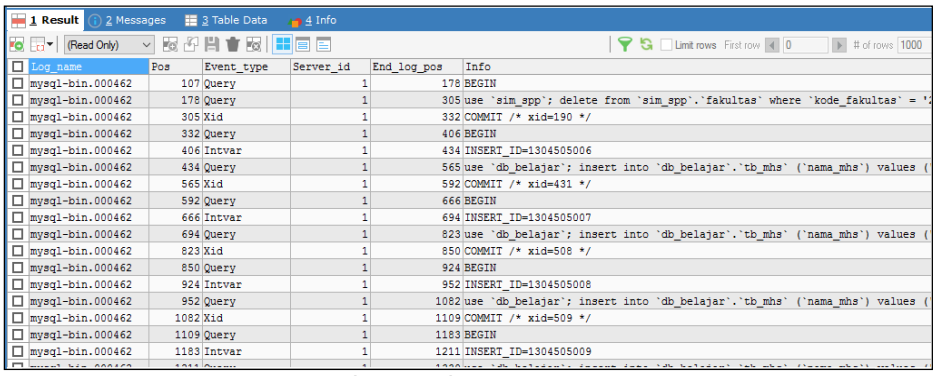
File Binary Log memiliki master index yang menampung berbagai macam *event* dari *query* yang berjalan.

Log_name	File_size
mysql-bin.000459	2446
mysql-bin.000460	382
mysql-bin.000461	126
mysql-bin.000462	17929
mysql-bin.000463	2563
mysql-bin.000464	107

Gambar 6. Master *Binary Log*

Gambar 6 merupakan master *binary log* yang digunakan sebagai acuan proses perekaman data melalui *binary log.* Kolom `Log_name` adalah nama master yang ada. Kolom `Pos` merupakan posisi terakhir dari isi *file* master tersebut. BinlogCDC menyimpan nama beserta posisi terakhir dari master setiap saat terjadi perubahan data. Proses ini berfungsi untuk menghindari adanya informasi perubahan data yang terlewatkan untuk direkam.

Posisi binlog memiliki nilai yang dimanfaatkan BinlogCDC untuk menjadi acuan proses pembacaan isi *binary log* itu sendiri.



1 Result | 2 Messages | 3 Table Data | 4 Info

(Read Only) | Limit rows | First row 0 | # of rows 1000

Log_name	Pos	Event_type	Server_id	End_log_pos	Info
mysql-bin.000462	107	Query	1	178	BEGIN
mysql-bin.000462	178	Query	1	305	use `sim_spp`; delete from `sim_spp`.`fakultas` where `kode_fakultas` = '2
mysql-bin.000462	305	Xid	1	332	COMMIT /* xid=190 */
mysql-bin.000462	332	Query	1	406	BEGIN
mysql-bin.000462	406	Intvar	1	434	INSERT_ID=1304505006
mysql-bin.000462	434	Query	1	565	use `db_belajar`; insert into `db_belajar`.`tb_mhs` (`nama_mhs`) values (
mysql-bin.000462	565	Xid	1	592	COMMIT /* xid=431 */
mysql-bin.000462	592	Query	1	666	BEGIN
mysql-bin.000462	666	Intvar	1	694	INSERT_ID=1304505007
mysql-bin.000462	694	Query	1	823	use `db_belajar`; insert into `db_belajar`.`tb_mhs` (`nama_mhs`) values (
mysql-bin.000462	823	Xid	1	850	COMMIT /* xid=508 */
mysql-bin.000462	850	Query	1	924	BEGIN
mysql-bin.000462	924	Intvar	1	952	INSERT_ID=1304505008
mysql-bin.000462	952	Query	1	1082	use `db_belajar`; insert into `db_belajar`.`tb_mhs` (`nama_mhs`) values (
mysql-bin.000462	1082	Xid	1	1109	COMMIT /* xid=509 */
mysql-bin.000462	1109	Query	1	1183	BEGIN
mysql-bin.000462	1183	Intvar	1	1211	INSERT_ID=1304505009

Gambar 7. *Binary Log Events*

Enam *field* dalam *binary log events* yaitu `Log_name`, `Pos`, `Events`, `Server_id`, `End_log_pos`, `Info`. *Field* Info merupakan informasi yang berisikan *query* dari aktivitas manipulasi data yang terjadi dalam *database.* Informasi inilah yang dimanfaatkan dalam membangun sebuah sistem *change data capture* dengan pendekatan *binary log* yang ditunjukkan pada Gambar 7.

Penggunaan pendekatan *binary log* membuat sistem bekerja tanpa harus menyentuh *database* dari sumber data secara langsung karena *query* yang ada dalam *binary log* menjadi satu-satunya acuan proses perekaman. Sumber data tetap aman tanpa harus takut adanya pihak yang mengubah struktur sumber data dengan hak ataupun tanpa hak sekalipun.

3. Kajian Pustaka

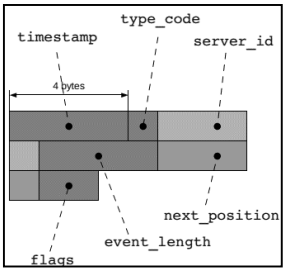
Pengumpulan teori-teori yang didapatkan dari buku atau internet maupun jurnal yang menunjang pembuatan aplikasi.

3.1. *State of The Art*

Change Data Capture (CDC) merupakan aplikasi yang membaca segala aktivitas yang berhubungan dengan manipulasi data dalam *database.* Metode yang digunakan dalam membangun CDC yaitu; berdasarkan *Database Trigger, Timestamp* pada setiap baris data, membaca *Log File*, dan berdasarkan melakukan partisi secara berjangka [3]. Pendekatan *Binary Log* cocok digunakan untuk membangun sinkronisasi data secara *real time* karena segala aktivitas *update* data yang terjadi pada *database* dalam MySQL tercatat secara *real-time* oleh *Binary Log*. Konfigurasi pada *Binary Log* juga mudah dilakukan serta memungkinkan untuk diolah pada level aplikasi sebagai acuan untuk mengetahui perubahan data secara *real-time.* Pemrosesan data yang *real time* dapat membantu dalam mempercepat pembaruan data pada sebuah instansi ataupun perusahaan yang menggunakan lebih dari satu *database* dengan *host* berbeda yang saling terintegrasi [1]. Metode CDC dengan memanfaatkan *log scanning* dapat menangkap semua berubah data yang berasal dari sumber data dalam durasi yang mendekati waktu sebenarnya tanpa mengubah struktur sistem sumber (ODS) [5].

3.2. Binary Log

Binary Log adalah *log* atau catatan yang berisi semua perintah DML atau *insert, update* dan *delete* pada mysql dan menjabarkan secara rinci tentang apa saja *query* yang digunakan saat melakukan manipulasi data pada *database* [4]. Struktur *binary log* terlihat pada Gambar 8.



Gambar 8. Struktur *Binary Log Events* [5]

Gambar 8 menunjukkan struktur dari Binary Log Event. Binary Log Event adalah informasi dari sebuah Event yang ada pada Binary Log. Adapun bagian dari *binary log* Event adalah sebagai berikut:

timestamp : catatan waktu eksekusi *query*
type_code : kode tipe dari *binary log event*
server_id : informasi server ID pada sebuah *host*
event_length : ukuran panjang dari *event* binlog
next_position : informasi posisi *binary log event*

Semua perintah atau sintaks disimpan dalam bentuk event yang menjelaskan perubahan atau manipulasi data pada database [4].

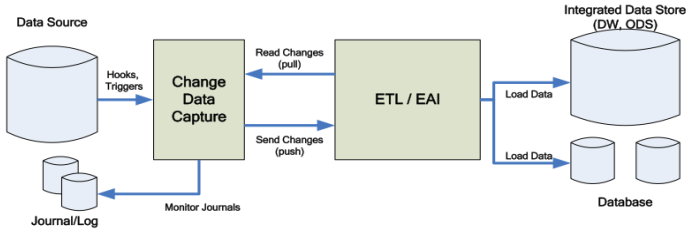
Tiga kegunaan mendasar dari *Binary Log* ini, yaitu :

1. Proses Replikasi, *Binary Log* digunakan oleh *Server Master* untuk menyimpan tiap perintah yang dikirim ke *Server Slave*. *Server Master* mengirim *event log* ke dalam *Binary Log* yang ada di *Server Slave*, sehingga dengan *Binary Log* pada *Server Slave*, *Server Slave* juga mengalami apa yang dialami oleh *Server Master* [4].
2. Salah satu proses *recovery* / *restore* menggunakan *Binary Log*. *Event* pada *Binary Log* disimpan setelah data *backup* di-*restore*. *Event* tersebut membuat *Database restore* menjadi *up to date* di proses *backup* [4].
3. Proses *log scanning* atau catatan yang ada pada *Binary Log* digunakan untuk menjadi acuan dalam membangun CDC [4].

Binary Log juga berisi seberapa jauh tiap perintah melakukan manipulasi pada data, sehingga dapat diketahui aktivitas perubahan yang terjadi pada suatu data atau *database* [6].

3.3. *Change Data Capture*

Change Data Capture (CDC) adalah teknik untuk merekam aktivitas yang terjadi pada *on-line transaction processing* (OLTP) sebagai sumber data untuk dilakukannya proses integrasi dengan tujuan datanya. CDC menjadikan integrasi data menjadi lebih efisien dan mengurangi *latency* antara waktu terjadinya perubahan data dengan data yang tersedia untuk diolah oleh pengguna [7]. Contoh penerapan CDC untuk memecahkan masalah bisnis dijabarkan pada Gambar 9.



Gambar 9. Infrastruktur *Change Data Capture* [6]

CDC pada *Data Warehouse* memerlukan media sebagai perantara agar dapat menangkap segala perubahan yang terjadi pada sumber data. Tiga pendekatan yang memungkinkan untuk digunakan dalam membangun CDC, yaitu; Berdasarkan *Trigger*, *Log Scanning,* dan *Database Replication* [6]. Teknik SCD digunakan untuk mencatat perubahan lambat yang terjadi pada tabel dimensi agar *history* data yang tersimpan dalam tabel dimensi tidak hilang [8].

4. Hasil dan Pembahasan

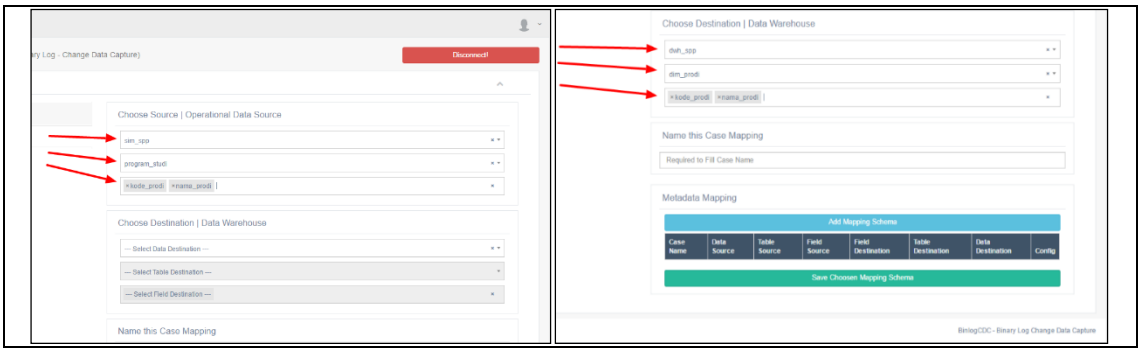
Hasil dan Pembahasan menjelaskan tentang tahapan-tahapan *Change Data Capture* dengan Pendekatan *Binary Log* yang terdiri dari proses metadata *mapping*, pembacaan *binary log* dan *loading data* yang diterapkan pada penelitian ini.

4.1. Pengujian Sistem

Langkah awal dalam melakukan pengujian yaitu melakukan pemetaan metadata pada aplikasi BinlogCDC. Pemetaan metadata menjadi acuan *engine* dalam melakukan proses *loading* ke dalam tujuan pengiriman data.

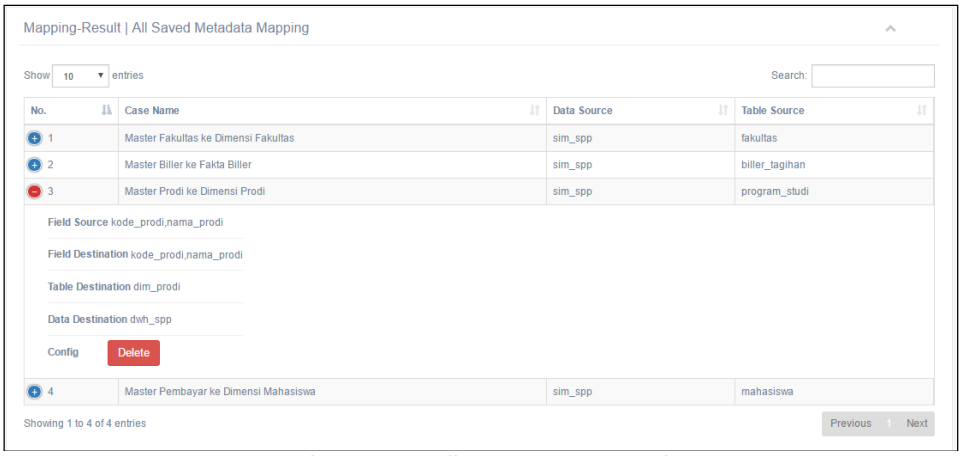
4.1.1. Pemetaan Metadata

Pemetaan metadata yaitu melakukan konektivitas terhadap tujuan data, dengan memasukkan *hostname, username, password,* dan *port* dari koneksi *database* yang menjadi tujuan. Metadata *mapping* hanya dapat dilakukan setelah pengguna terkoneksi ke *host* tujuan. Metadata *mapping* dimulai dengan memilih sumber data yaitu nama *database* dari OLTP, dilanjutkan dengan memilih tabel dan *field* yang ingin diambil datanya untuk keperluan proses *loading* yang ditunjukan pada Gambar 10.



Gambar 10. Pemetaan Sumber Data

Tahap berikutnya yaitu memilih tujuan data. Proses pemilihan tujuan data dimulai dengan memilih nama *database,* tabel, dan *field* yang menjadi tujuan pengiriman data.



Gambar 11. Hasil Pemetaan Metadata

Hasil dari pemetaan ditampilkan pada halaman *Mapping Result* seperti yang ditunjukkan Gambar 11. Segala manipulasi data yang dilakukan pada tabel `program_studi` pada *database* `sim_spp` direkam dan dilakukan proses *loading* ke dalam tabel `dim_prodi` pada *database* `dwh_spp`. Proses pendorongan data dilakukan secara *real time,* yang mana proses ini merupakan tujuan utama dari dibuatnya *change data capture* dengan pendekatan *binary log.*

4.1.2. *Loading* Data

Tahap pengujian berikutnya yaitu melakukan suatu manipulasi pada salah satu data di tabel `program_studi` yang memiliki tujuan data yaitu tabel dimensi `dim_prodi`. Manipulasi data yang dilakukan yaitu proses *insert*, *update*, dan *delete* data yang dilakukan pada tabel yang sudah dilakukan pemetaan antara sumber data dengan tujuan data pada tahapan pengujian sebelumnya.

kode_prodi	nama_prodi	kode_fakultas
3	Arkeologi	1
4	Teknik Arsitektur	3
5	Teknik Mesin	3
6	Teknik Sipil	3
7	Akuntansi	2
9	Ekonomi Pembangunan	2
10	Manajemen	2
11	Ilmu Hukum	5
13	Matematika	10
14	Sastra Jepang	1
15	Teknologi Informasi	3
(Auto)	*(NULL)*	*(NULL)*

Gambar 12. *Insert* data Program Studi

Gambar 12 merupakan tahap melakukan *insert* data. Data yang dimasukkan adalah nama program studi yaitu "Teknologi Informasi". *Primary Key* dari tabel ini yaitu `kode_prodi` dengan `id` *auto increment* terakhir adalah "14", sehingga saat data baru ditambahkan pastinya memiliki id yaitu "15".

dwh_spp.dim_prodi: 14 rows total (approximately)

id	kode_prodi	nama_prodi	start_time	end_time	flag
1	1	Sastra Jawa Kuno	2017-04-20 14:08:38	0000-00-00 00:00:00	1
2	2	Sastra Inggris	2017-04-20 14:08:50	0000-00-00 00:00:00	1
3	3	Arkeologi	2017-04-20 14:09:02	0000-00-00 00:00:00	1
4	4	Teknik Arsitektur	2017-04-20 14:09:14	0000-00-00 00:00:00	1
5	5	Teknik Mesin	2017-04-20 14:09:50	0000-00-00 00:00:00	1
6	6	Teknik Sipil	2017-04-20 14:10:02	0000-00-00 00:00:00	1
7	7	Akuntansi	2017-04-20 14:10:15	0000-00-00 00:00:00	1
13	9	Ekonomi Pembangunan	2017-04-20 14:15:38	0000-00-00 00:00:00	1
14	10	Manajemen	2017-04-20 14:16:15	0000-00-00 00:00:00	1
15	11	Ilmu Hukum	2017-04-21 16:02:54	0000-00-00 00:00:00	1
16	12	Farmasi	2017-04-21 16:48:38	2017-04-21 16:48:52	0
17	13	Matematika	2017-04-21 16:48:52	0000-00-00 00:00:00	1
18	14	Sastra Jepang	2017-04-27 19:10:18	0000-00-00 00:00:00	1
19	15	Teknologi Informasi	2017-04-27 19:13:22	0000-00-00 00:00:00	1

Gambar 13. Hasil *Capture Insert* Data pada Tujuan Program Studi

Gambar 13 merupakan data hasil *capture* yang telah sampai ke dalam tujuan data. Data dengan `kode_prodi` "15" dan `nama_prodi` "Teknologi Informasi" telah masuk dengan `start_time` "2017-04-27 19:13:22" yang merupakan waktu dilakukan penambahan data pada sumber data. Kolom `end_time` bernilai kosong karena data masih ada dan aktif pada sumber data. Kolom `flag` bernilai "1" untuk data yang masih aktif.

kode_prodi	nama_prodi	kode_fakultas
1	Sastra Jawa Kuno	1
2	Sastra Inggris	1
3	Arkeologi	1
4	Teknik Arsitektur	3
5	Teknik Mesin	3
6	Teknik Sipil	3
7	Akuntansi	2
9	Ekonomi Pembangunan	2
10	Manajemen	2
11	Ilmu Hukum	5
13	Matematika	10
14	Sastra Jepang	1
(Auto)	*(NULL)*	*(NULL)*

(a)

kode_prodi	nama_prodi	kode_fakultas
3	Arkeologi	1
4	Teknik Arsitektur	3
5	Teknik Mesin	3
6	Teknik Sipil	3
7	Akuntansi	2
9	Ekonomi Pembangunan	2
10	Manajemen	2
11	Ilmu Hukum	5
13	Hitung-hitungan	10
14	Sastra Jepang	1
15	Teknologi Informasi	3
(Auto)	*(NULL)*	*(NULL)*

(b)

Gambar 14. (a) Data Awal (b) Hasil *Update* data Program Studi

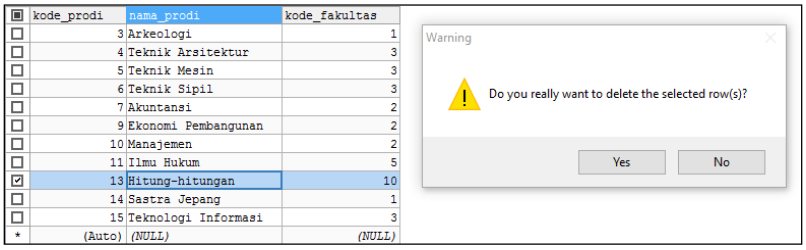
Gambar 14 merupakan pengujian tahap kedua yaitu *update* data yang dilakukan pada data dengan `kode_prodi` "13" (a). Data yang diubah memiliki `nama_prodi` awal yaitu "Matematika" dan diubah menjadi "Hitung-hitungan" (b).


dwh_spp.dim_prodi: 15 rows total (approximately)

id	kode_prodi	nama_prodi	start_time	end_time	flag
1	1	Sastra Jawa Kuno	2017-04-20 14:08:38	0000-00-00 00:00:00	1
2	2	Sastra Inggris	2017-04-20 14:08:50	0000-00-00 00:00:00	1
3	3	Arkeologi	2017-04-20 14:09:02	0000-00-00 00:00:00	1
4	4	Teknik Arsitektur	2017-04-20 14:09:14	0000-00-00 00:00:00	1
5	5	Teknik Mesin	2017-04-20 14:09:50	0000-00-00 00:00:00	1
6	6	Teknik Sipil	2017-04-20 14:10:02	0000-00-00 00:00:00	1
7	7	Akuntansi	2017-04-20 14:10:15	0000-00-00 00:00:00	1
13	9	Ekonomi Pembangunan	2017-04-20 14:15:38	0000-00-00 00:00:00	1
14	10	Manajemen	2017-04-20 14:16:15	0000-00-00 00:00:00	1
15	11	Ilmu Hukum	2017-04-21 16:02:54	0000-00-00 00:00:00	1
16	12	Farmasi	2017-04-21 16:48:38	2017-04-21 16:48:52	0
17	13	Matematika	2017-04-21 16:48:52	2017-04-27 19:15:20	0
18	14	Sastra Jepang	2017-04-27 19:10:18	0000-00-00 00:00:00	1
19	15	Teknologi Informasi	2017-04-27 19:13:22	0000-00-00 00:00:00	1
20	13	Hitung-hitungan	2017-04-27 19:15:20	0000-00-00 00:00:00	1

Gambar 15. Hasil *Capture Update* Data pada Tujuan Program Studi

Data yang dimanipulasi telah direkam dan hasil *capture* data terlihat pada Gambar 15. Data lama pada tujuan tidak diubah ataupun dihapus, hanya terjadi pengisian waktu terjadinya manipulasi pada kolom `end_time` dan mengganti nilai pada kolom `flag` dari "1" menjadi "0". Data baru hasil *capture* memiliki `id` sumber data yang sama. Aturan dasar dari konsep *data warehouse* adalah selalu menyimpan riwayat perubahan dan manipulasi yang dilakukan pada sumber data. Data "Hitung-hitungan" pada kolom `nama_prodi` telah sampai pada tujuan data dengan `flag` bernilai "1" yang menandakan bahwa data dengan `nama_prodi` "Matematika" telah diubah menjadi "Hitung-hitungan".



Gambar 16. Melakukan *Delete* data Program Studi

Tahap terakhir merupakan proses menghapus data pada tabel `program_studi`. Proses *delete* data yang dilakukan yaitu pada data dengan `kode_prodi` "13". Proses penghapusan data terlihat pada Gambar 16.


dwh_spp.dim_prodi: 15 rows total (approximately)

id	kode_prodi	nama_prodi	start_time	end_time	flag
1	1	Sastra Jawa Kuno	2017-04-20 14:08:38	0000-00-00 00:00:00	1
2	2	Sastra Inggris	2017-04-20 14:08:50	0000-00-00 00:00:00	1
3	3	Arkeologi	2017-04-20 14:09:02	0000-00-00 00:00:00	1
4	4	Teknik Arsitektur	2017-04-20 14:09:14	0000-00-00 00:00:00	1
5	5	Teknik Mesin	2017-04-20 14:09:50	0000-00-00 00:00:00	1
6	6	Teknik Sipil	2017-04-20 14:10:02	0000-00-00 00:00:00	1
7	7	Akuntansi	2017-04-20 14:10:15	0000-00-00 00:00:00	1
13	9	Ekonomi Pembangunan	2017-04-20 14:15:38	0000-00-00 00:00:00	1
14	10	Manajemen	2017-04-20 14:16:15	0000-00-00 00:00:00	1
15	11	Ilmu Hukum	2017-04-21 16:02:54	0000-00-00 00:00:00	1
16	12	Farmasi	2017-04-21 16:48:38	2017-04-21 16:48:52	0
17	13	Matematika	2017-04-21 16:48:52	2017-04-27 19:15:20	0
18	14	Sastra Jepang	2017-04-27 19:10:18	0000-00-00 00:00:00	1
19	15	Teknologi Informasi	2017-04-27 19:13:22	0000-00-00 00:00:00	1
20	13	Hitung-hitungan	2017-04-27 19:15:20	2017-04-27 19:17:04	0

Gambar 17. Hasil *Capture Delete* Data pada Tujuan Program Studi

Gambar 17 menunjukan bahwa hasil *capture* dari proses menghapus data yang dilakukan pada sumber data telah sampai hasilnya ke dalam tujuan data. *Data warehousing* memiliki konsep bahwa data yang dihapus pada sumber data, tidak hilang pada tujuan datanya. Data pada tujuan hanya diberikan nilai waktu pada kolom `end_time` dan pada kolom `flag` nilainya diubah menjadi "0" yang membuktikan data telah berakhir masa aktifnya karena sudah tidak ada pada sumber data.

4.2. Analisa Hasil Pengujian

Berdasarkan hasil uji coba dapat diketahui bahwa sistem yang dibuat mampu mengatasi permasalahan-permasalahan yang dikemukakan sebelumnya, yaitu :

1. Implementasi sistem memiliki sifat dinamis yang tinggi untuk diimplementasikan pada mesin basis data MySQL dan sangat efektif untuk dipasang pada sistem *data warehouse* yang memerlukan kebaruan data terkini.
2. Sistem dapat melakukan proses *capture* pada data dalam sumber data yang mengalami manipulasi *insert, update,* dan *delete* yang kemudian langsung diolah untuk didorong ke dalam tujuan data.
3. Sistem sama sekali tidak menyentuh *database* dari sumber data dalam bentuk proses apapun seperti *select* atau *view* dan tidak harus mengubah struktur tabel data sumber seperti halnya menambahkan kolom *flag* sebagai penanda, karena sistem menggunakan pendekatan *binary log* dan murni hanya *binary log* sebagai acuan dalam membangun suatu *engine change data capture* berbasis *binary log.*
4. Sistem berhasil merekam segala aktivitas manipulasi data tanpa terlewatkan sedikitpun meskipun terjadi banyak transaksi dan *file* master *binary log* telah mengalami pergantian ketika *engine* dalam keadaan dimatikan. Keberhasilan ini dikarenakan setiap kali proses pembacaan informasi dalam *binary log* dilakukan, sistem selalu menyimpan nama *file* master beserta posisi terakhirnya, sehingga tidak memungkinkan adanya informasi yang terlewatkan ataupun terduplikasi.

4.3. Perbandingan BinlogCDC dengan sistem CDC yang telah ada sebelumnya

Perbandingan sistem BinlogCDC dengan beberapa sistem CDC yang telah ada sebelumnya ditampilkan dalam bentuk tabel yaitu pada Tabel 1.

Tabel 1. Perbandingan Jenis-Jenis CDC

Cdc Approach	Process Capture Speed	Dynamic Data Source	Host To Host	Accuration
Base on Trigger	Fast	No	No	Good
Replication Method	Mid	No	Yes	Good
Comparing Source and Target by Using Crc	Slow	Yes	Yes	Mid
Full Comparing Source and Target	Slow	Yes	Yes	Bad
Binary Log Approach	**Fast**	**Yes**	**Yes**	**Good**

Perbandingan pada Tabel 1 berlaku jika sistem sudah berjalan dengan sempurna dan telah diaplikasikan dengan melakukan konfigurasi *binary log* pada DBMS yang digunakan. CDC dengan memanfaatkan *Trigger* merupakan pendekatan CDC yang memiliki proses *capture* tercepat, yaitu satu detik hingga lima belas detik untuk sekali proses. Pendekatan *binary log* dapat melakukan proses *capture* tiga detik hingga sepuluh detik untuk sekali proses. Tiga detik pertama merupakan waktu yang terpakai untuk melakukan koneksi jika server tujuan data memiliki *host* yang berbeda. Pendekatan *binary log* memiliki kinerja yang lebih baik dibandingkan metode ataupun pendekatan yang telah ada sebelumnya dalam berbagai aspek.

4.4. Kelebihan dan Kekurangan

Pembuatan suatu sistem tentunya memiliki kelebihan maupun kekurangan sesuai dengan metode dan proses-proses yang digunakan. Tujuan dari dibangunnya sebuah model pengembangan adalah untuk menangani kekurangan-kekurangan pada sistem yang ada sebelumnya. Aplikasi BinlogCDC ini memiliki kelebihan-kelebihan diantaranya :

1. Kecepatan proses *capture* data hingga proses *loading* data ke tujuan menggunakan BinlogCDC dapat dikatakan maksimal dan mendekati waktu yang sebenarnya.
2. Sumber data yang dinamis dapat memudahkan pengimplementasian sistem terhadap berbagai *database* tanpa harus mengubah ataupun menambah kolom bantuan pada tabel bantu yang menjadi sumber data.
3. Tujuan data dapat berada pada *host* yang berbeda,
4. Pemrosesan data yang *real time* dapat membantu dalam mempercepat perbaruan data dalam sebuah instansi ataupun perusahaan.

5. Penanganan masalah ketika *engine* mati akibat *server* yang *down* berhasil dilakukan, sehingga tidak lagi terjadi hilangnya informasi akibat manipulasi data.
6. Penggunaaan pendekatan *binary log* membuat sistem bekerja tanpa harus menyentuh *database* dari sumber data secara langsung karena *query* yang ada dalam *binary log* adalah satu-satunya acuan. Sumber data tetap aman tanpa harus takut adanya pihak-pihak yang mengubah struktur sumber data tanpa hak.

Aplikasi yang dikembangkan pastilah memiliki kekurangan, begitu pula halnya dengan sistem BinlogCDC. Kekurangan yang ada pada aplikasi BinlogCDC antara lain:

1. Konfigurasi *binary log* saat awal pemasangan sistem harus dilakukan dengan benar terlebih dahulu. Konfigurasi juga harus dilakukan pada level DBMS karena belum ditemukan cara menjalankan *query binary log* melalui level aplikasi.
2. BinlogCDC hanya terfokus pada proses *capture* terhadap data yang mengalami perubahan atau dimanipulasi sehingga belum adanya proses ETL lainnya seperti *data cleansing*.
3. Aplikasi harus dipasang pada *host* yang sama dengan sumber data karena belum ditemukan cara pengolahan *binary log* dari *server* memiliki *host* berbeda.

5. Kesimpulan

Hasil pengujian dari pemanfaatan *binary log* dalam *change data capture* menunjukkan aplikasi sudah berfungsi dan memberikan hasil yang sesuai dengan konsep dasar *change data capture* dengan *delay* waktu yang mendekati waktu sebenarnya. Proses *real time* memanfaatkan *time delay* sehingga *engine* dapat dikonfigurasikan sesuai dengan pengaturan durasi *time delay*. *Engine* BinlogCDC memakan waktu minimal tiga detik dan maksimal sepuluh detik untuk sekali melakukan proses *capture* hingga *loading* data. BinlogCDC menjadikan informasi yang tersedia dalam data warehouse sebagai tujuan data, selalu diperbarui setiap saat terjadinya manipulasi atau perubahan pada OLTP sebagai sumber data sehingga informasi terkini yang tersimpan menjadi valid. Komunikasi antar sistem dapat dilakukan secara dinamis yaitu sumber dengan tujuan data yang berbeda *host* dapat dilakukan dengan struktur sumber data yang dinamis, sehingga tidak harus merombak sistem pada level *coding*. Pemanfaatan *binary log* untuk CDC dapat mengatasi masalah terkait pengelolaan data yang diakibatkan oleh matinya *engine* ataupun transaksi yang bergerak terlalu cepat dan menyebabkan banyak informasi yang terlewatkan.

Daftar Pustaka

[1] I. G. S. Aryandana, I. M. Sukarsa, and P. W. Buana, “Pembentukan Data Mart Menggunakan Metode Generalization”, *Lontar Komputer.*, vol. 7, no. 3, pp. 814–825, 2016.
[2] W. M. M. Ali. Abdelmgeid A, “Monitoring Business Transactions for a Real-time Data Warehouses”, *International Journal of Computer Applications*, vol. 146, no. 8, pp. 8–11, 2016.
[3] Sukarsa. I Made, et al, “Change Data Capture on OLTP Staging Area for Nearly Real Time Data Warehouse base on Database Trigger”, *International Journal of Computer Applications*, vol. 52, no. 11, pp. 32–37, 2012.
[4] G. H. Surya, I. M. Sukarsa, I. G. Made, and A. Sasmita, “Two-Ways Database Synchronization in Homogenous Database Management System”, vol. 65, no. 1, 2014.
[5] J. G. Shi, Y. Bin Bao, F. L. Leng, and G. Yu, “Study on log-based change data capture and handling mechanism in real-time data warehouse”, *Proc. - Int. Conf. Comput. Sci. Softw. Eng. CSSE 2008*, vol. 4, pp. 478–481, 2008.
[6] M. Kindahl and L. Thalmann, *An API for Reading the MySQL Binary Log*. Oracle.
[7] Atunnity, “Efficient and Real Time Data Integration with Change Data Capture”, 2009, no. Cdc, pp. 1–20.
[8] W. Wisswani, “Penerapan Hybrid Slowly Change Dimension untuk Nearly Realtime Datawarehouse”, *Lontar Komputer,* vol. 4, no. 1, pp. 215–223, 2013.