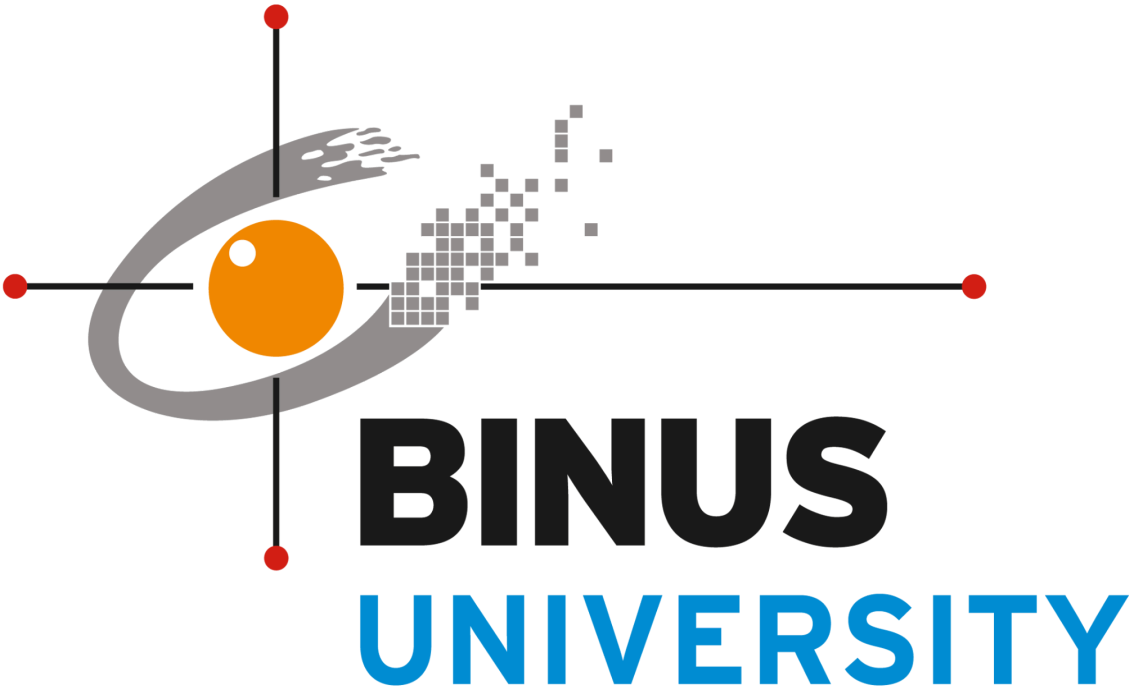
Database Administration (LC-01)

Replikasi dan Migrasi database dari MySQL ke Oracle pada PT. Metro Batavia



Disusun oleh:
Karel Alexander (2301869992)
Michael Albert Purnama (2301877716)
Andre Panji (2301909346)
David Yu (2301932456)
Andini Artika Dewi (2301935672)

I. Introduction

Perkembangan di dunia bisnis sekarang sudah berkembang dengan begitu cepat. Dengan banyaknya perusahaan yang terjun, membuat persaingan terasa semakin kompetitif. Untuk mengatasi persaingan yang ada, banyak perusahaan yang berlomba-lomba untuk berinovasi dengan memanfaatkan perkembangan teknologi yang ada sehingga proses bisnis dapat berjalan lebih efektif dan efisien. Sehingga, dengan banyaknya perusahaan yang sudah memanfaatkan teknologi yang ada, membuat jaminan akan keamanan dan keandalan teknologi tersebut

menjadi faktor yang sangat penting. Namun, seperti yang kita tahu, tidak ada satupun penyedia teknologi yang dapat menjamin seratus persen semuanya aman, seperti sistem yang berjalan tidak akan mengalami gangguan. Hal ini membuat perusahaan-perusahaan menghadapi resiko seperti integrasi data yang menggunakan database yang berbeda, karena dapat terjadinya *downtime* ketika migrasi data terjadi, yang membuat kinerja dari server menjadi tinggi membuat server bisa mengalami *crash*. Gangguan seperti ini tentu bisa merugikan perusahaan tersebut.

Di dalam laporan ini, kami ingin membahas tentang Replikasi dalam suatu Database. Dengan mengambil contoh studi kasus dari PT. Metro Batavia yang di mana perusahaan ini bergerak di jasa transportasi udara (*airline*). Dalam bisnisnya, PT. Metro Batavia belum menerapkan semua sistem yang terintegrasi ke dalam satu database yang sama, dimana sistem *hangar* pada perusahaan ini masih menggunakan Database MySQL. Dan sekarang, dikarenakan kebutuhan data dan informasi yang begitu banyak, perusahaan ini menginginkan melakukan migrasi database dari MySQL menjadi Oracle agar dapat menjalankan sistemnya dengan baik.

Data dan informasi yang begitu banyak dihasilkan dalam proses bisnisnya, membuat PT. Metro Batavia memerlukan adanya sistem *hangar* yang berjalan secara *realtime* sehingga data dan informasi yang masuk dapat selalu disalurkan *on time*. Kemudian juga, PT. Metro Batavia menginginkan adanya suatu sistem yang bisa melakukan *continuous data availability* sehingga sistem tersebut bebas dari gangguan yang bisa membuat perusahaan itu kehilangan data dan membuat proses bisnis mereka jadi terhambat.

Karena itu, PT. Metro Batavia menginginkan adanya perancangan terhadap datanya dengan menggunakan replikasi data pada sistem *hangar* agar bisa menjaga kelangsungan dari proses bisnis mereka, serta menjaga keutuhan informasi mereka.

Disini, kami akan melakukan analisis terhadap migrasi dan replikasi yang dilakukan oleh PT. Metro Batavia, sesuai dengan topik 9, yaitu *Replication*.

II. Theory Basis

Replikasi merupakan suatu proses menyalin data dari database central ke satu atau lebih database. Replikasi dalam database mempunyai beberapa komponen, yakni roles, data, agents, dan internal components. Roles dalam replikasi dibagi menjadi beberapa bagian, yakni Publisher, Distributor, dan Subscriber. Publisher adalah server/database instance yang menjadi sumber untuk artikel yang dipublish. Distributor adalah perantara dalam tindakan publishing dan subscribing, dan dalam beberapa jenis replikasi, distributor menyimpan data yang akan direplikasi dari publisher. Distributor adalah database yang dapat hidup di

server publishing, server sendiri, atau pun di subscriber. Subscriber adalah server yang menjadi tujuan artikel yang dipublish. Dalam beberapa model replikasi, subscriber juga bisa menjadi publisher. Subscriber dapat mempublish ulang artikel ketika mereka harus dikirim ke subscriber lain dalam Updating Subscriber model. Subscriber juga dapat mempublish ulang artikel dalam model peer-to-peer.

Komponen data dalam replikasi dibagi menjadi tiga, yaitu Article, Publication, dan Subscription. Article merupakan kumpulan data terkecil yang dapat dikonfigurasi untuk replikasi. Article dapat terdiri dari table, view, atau stored procedure, dan juga dapat memiliki batasan tambahan pada baris dan kolom yang disertakan dalam setiap article. Lalu, Publication merupakan pengelompokkan article yang dipublish bersama-sama. Menggunakan publication memungkinkan replikasi article dikelompokkan secara logis untuk dikelola bersama dibandingkan harus mengelola setiap article satu per satu. Selanjutnya, Subscription adalah permintaan untuk menerima data dari satu atau lebih publikasi. Lalu, replication agents, adalah program yang dapat dieksekusi yang melakukan banyak pekerjaan replikasi. Mereka biasanya dieksekusi melalui SQL Server Agent jobs, tapi dapat juga dijalankan secara manual. SQL Server Agent jobs berikut ini dibuat dengan replikasi, Snapshot agent, dijalankan oleh SQL Agent jobs yang mengambil dan menerapkan snapshot untuk tiga jenis replikasi, yakni transaksi, penggabungan, dan replikasi snapshot. Lalu ada Log Reader agent, dijalankan untuk membaca log transaksi pada publisher dan mencatat transaksi untuk setiap artikel yang diterbitkan ke dalam distribution database. Distribution agent, dijalankan untuk membaca transaksi yang ditulis ke distribution database dan menerapkannya ke subscribing database untuk replikasi transaksi. Merge agent, dijalankan untuk memindahkan perubahan di publisher ke subscriber, memindahkan perubahan dari subscriber ke publisher, dan memulai proses konflik jika perlu. Yang terakhir, Queue Reader Agent, digunakan untuk membaca pesan yang disimpan dalam antrian SQL Server, atau antrian Microsoft Message. Kemudian, menerapkan pesan-pesan itu ke publisher.

Selanjutnya, ada Replication Maintenance Jobs, dimana replikasi menggunakan tambahan jobs dari SQL Server Agent untuk melakukan beberapa pemeliharaan, yakni, Agent History Cleanup, yang mana pekerjaan ini untuk menghapus riwayat replication agent history yang disimpan dalam distribution database. Distribution cleanup, untuk menghapus transaksi dari distribution database setelah tidak lagi diperlukan. Expired Subscription Cleanup, bertugas untuk menentukan kapan snapshot telah kedaluwarsa dan menghapusnya. Reinitialize Failed Subscriptions, pekerjaan ini mencari subscription yang gagal dan menandainya untuk inisialisasi ulang. Replication Agent Monitor, untuk memantau eksekusi SQL Agents, dan menulis ke event log Windows saat job step gagal. Replication Agents Checkup, untuk memantau eksekusi replication agents yang sedang berjalan, tetapi belum mencatat riwayat apapun dalam interval yang ditentukan.

Lalu ada beberapa jenis dari Replikasi, antara lain: Snapshot, Transactional, dan Merge. Snapshot replication sesuai namanya mengambil snapshot dari publikasi dan membuatnya tersedia untuk subscribers. Ketika snapshot diterapkan pada subscribing database, artikel di subscribers (seperti table, views, dan stored procedure) di-drop dan dibuat ulang. Dalam replikasi snapshot ini tidak ada aliran data yang berkelanjutan dari publisher ke subscriber. Replikasi jenis ini paling cocok untuk data yang cukup statis, saat salinan data yang kedaluwarsa di antara interval replikasi dapat diterima, atau saat ukuran artikel kecil. Lalu selanjutnya Transactional Replication, mereplikasi perubahan pada artikel saat terjadi. Untuk menyiapkan transactional replication, snapshot diambil dan diterapkan ke subscriber satu kali untuk membuat kumpulan data yang sama. Setelah itu, Log Reader agent membaca semua transaksi yang terjadi terhadap artikel yang diterbitkan, dan mencatatnya dalam distribution database. Transactional replication ini memungkinkan sinkronisasi data yang lebih cepat dengan latensi yang lebih sedikit. Bergantung pada cara pengaturannya, sinkronisasi data ini dapat terjadi hampir secara real time. Yang terakhir adalah Merge Replication, biasanya digunakan setiap kali ada koneksi jaringan yang lambat atau terputus-putus antara publisher dan subscriber. Ini memungkinkan situs untuk bekerja secara mandiri, dan untuk menyinkronkan perubahan pada data saat mereka online kembali. Diperlukan snapshot untuk menginisialisasi replikasi, setelah itu perubahan selanjutnya dilacak dengan triggers. Salah satu efek dari merge replication ini adalah kemungkinan konflik jika saat perubahan offline disinkronkan. Merge replication secara otomatis menyelesaikan masalah ini di Merge agent menggunakan model pemecah konflik yang dipilih saat publikasi dibuat.

III. Methods

3.1 Riwayat Perusahaan

PT. Metro Batavia telah memulai bisnis di indonesia lebih dari 20 tahun yang dimulai dari usaha *travel agent* dan terus tumbuh menjadi usaha *charter* untuk angkatan udara. Pada tahun 2002, Batavia Air memperoleh sertifikasi sebagai operator penerbangan dan pada Januari 2006 mereka menjadi perusahaan penerbangan nasional pertama yang mengoperasikan pesawat Airbus A-319. Dan sekaran, Batavia Air mengoperasikan lebih dari 150 rute penerbangan setiap harinya (sebelum pandemi) baik dalam negeri maupun mancanegara.

3.2 Analisis Sistem Hanggar Berjalan

3.2.1 Alur Sistem Berjalan

Di dalam sistem hanggar, ada tiga bagian penting yang saling terkait yaitu, *Land Maintenance (LM), Base Maintenance (BM), dan PPC & MMC Login.* Biasanya LM dan BM bertugas untuk melakukan reparasi pesawat, pengecekan kondisi pesawat, reparasi pesawat, serta perawatan berkala untuk setiap pesawat. Sedangkan PPC & MMC akan melakukan pembuatan *Taskcard* agar prosedur pada sistem hanggar dapat dimulai.

Untuk program hanggar, digunakan 3 server untuk memenuhi kebutuhan data logistik, fungsi login, dan data hanggar. Dan setiap server yang ada, memiliki database masing-masing. Database yang digunakan adalah MySQL. Disinilah ditemukan masalah, dimana MySQL tidak menyediakan fasilitas dblink, sehingga ketika database terdapat di 2 server yang berbeda harus dibangun 2 koneksi terlebih dahulu, selain itu juga MySQL juga tidak mendukung fasilitas interjoin dan juga leftjoin.

3.2.2 Database Sistem Hanggar

Dalam proses migrasi dan replikasi yang akan dilakukan, database yang digunakan adalah database sistem Hanggar, dimana database ini berisikan data-data yang berhubungan dengan kondisi pesawat, task yang dikerjakan, *workorder, reminder,* dan persediaan suku cadang untuk reparasi pesawat.

3.3 Analisis Proses Migrasi dan Replikasi

Proses migrasi database adalah proses untuk mengkonversi data dari satu database ke database yang lain baik dalam *environment* yang sama maupun di dalam *environment* yang berbeda. *Environment* yang dimaksud adalah *hardware* dan *software* yang digunakan bersamaan dengan database. Di dalam proses ini, tidak hanya ada saja yang dipindahkan, tetapi juga objek-objek yang ada di dalam database tersebut termasuk *table, view, trigger, stored procedure, function,*dan objek-objek lainnya.

Sedangkan proses replikasi database adalah proses menggandakan semua perubahan yang terjadi pada suatu database yang disebut *master server* atau

master ke server lain yang disebut *slave server* atau juga *slave.* Dengan ini, suatu perusahaan dapat membuat replika dari database yang sedang mereka pakai sebagai *backup* apabila terjadi kegagalan pada sistem, baik yang disengaja maupun yang tidak agar perusahaan tersebut masih tetap memiliki salinan dari server utama untuk menjalankan *reporting* dan analisis kerja tanpa mengganggu jalannya proses bisnis.

3.4 Analisis Masalah

Dan masalah yang kini dihadapi oleh PT. Metro Batavia berkaitan dengan proses migrasi dan replikasi khususnya untuk sistem hanggar, antara lain:

1. PT. Metro Batavia menggunakan MySQL untuk database pada sistem hanggar mereka. Beberapa masalah yang ada ketika menggunakan database ini, antara lain:
    - Ukuran penampungan data pada MySQL sudah tidak cukup untuk menampung banyaknya data yang harus disimpan. Sehingga dibutuhkan database yang lebih besar.
    - Proses pengolahan data menjadi lambat karena database sudah terlalu padat sehingga proses bisnis pun menjadi lambat.
    - Tidak mendukung fitur DB-Link. Sistem hanggar berhubungan dengan database sistem lain, sehingga diperlukan koneksi antar dua database tersebut. Tanpa fitur DB-Link, koneksi yang harus dibuat akan semakin rumit.
    - Dari pihak PT. Metro Batavia memilih untuk melakukan migrasi dari MySQL menuju Oracle, dimana antar kedua database ini memiliki perbedaan yang mendasar seperti proses pembacaan metadata, tipe data, dan syntax SQL yang berbeda antar satu sama lain, sehingga jika memutuskan untuk melakukan migrasi, maka diperlukan *tool* yang kuat, hemat biaya, serta dapat melakukan migrasi pada berbagai platform.

2. Database pada sistem hanggar biasanya diakses setiap hari (24 x 8) yang dimana memiliki beberapa permasalahan, antara lain:
    - Pengaksesan pada database yang berlebihan (*overload transaction)* menyebabkan pengolahan data akan menjadi lambat, tentu ini akan mempengaruhi proses bisnis yang ada.
    - SIstem hanggar belum mempunyai database *backup* sehingga sistem tidak akan dapat berjalan ketika adanya gangguan yang menyebabkan

kegagalan pada database utama. Tentu jika dibiarkan, maka akan menyebabkan kerugian yang cukup besar untuk PT. Metro Batavia.

3.5 Analis Migrasi Data menggunakan Oracle GoldenGate

Dengan menggunakan Oracle GoldenGate v11.1.1, author disini mencoba untuk melakukan migrasi database dari MySQL menuju Oracle dengan berbagai langkah yang banyak yang menggunakan platform source Centos 5.5 dan platform target menggunakan RedHat Linux 4.7

3.5.1 Konfigurasi pada source database (MySQL)

Hal yang dilakukan pertama adalah, dilakukan konfigurasi pada TCP/IP dan DNS pada /etc/hosts dengan tujuan agar *secondary system* dapat dikenal oleh *hostname*-nya. Setelah itu, garis dipastikan bahwa port 7809 belum digunakan dengan command **netstat-na.** Port ini adalah port default yang akan digunakan untuk manager dari Oracle GoldenGate. Disini manager bertindak sebagai admin dan mengatur segala aktivitas dari GoldenGate pada sistem tersebut. Kemudian akan dibuat user sistem yang akan diberi hak dalam pengaksesan dan penggunaan Oracle GoldenGate.

3.5.2 Konfigurasi pada target database (Oracle)

Pada tahap sebelumnya, telah dilakukan *create* pada user sistem untuk mengakses database, selanjutnya akan dilakukan lebih lanjut pada Oracle dengan cara dilakukan konfigurasi pada TCP/IP dan DNS pada /etc/hosts, yang kemudian harus dipastikan bahwa port 7809 belum digunakan dengan command **netsat-na**.

3.5.3 Konfigurasi Extract pada source database (MySQL)

Disini, kita akan melakukan start pada MySQL sehingga Oracle bisa mengakses databasenya, kemudian tambahkan primary extract untuk primary system. Lalu add extrail untuk primary extract, terakhir edit parameter extract

3.5.4 Konfigurasi Replication pada target database (Oracle)

Lakukan alter database dalam mode archievelog, lalu add supplemental log data diikuti open database jika database belum dalam keadaan open. Kemudian add replicat untuk secondary system, dilanjutkan dengan edit parameter pada replicat

3.5.5 Masalah Binary Log

Pada tahap-tahap sebelumnya, sudah dilakukan penyesuaian environment dengan Oracle GoldenGate, melakukan inisialisasi extract dan replicat, diikuti dengan parameter-parameternya. Kemudian yang harus dilakukan adalah START EXTRACT ext_1. Namun disini ada masalah, dimana ketika dijalankan, maka akan didapatkan extract tidak berjalan dan berstatus ABENDED.

Error menyatakan bahwa Oracle GoldenGate menemukan bahwa binary log pada mysql-bin.index tidak dalam keadaan enabled.

3.5.6 Hasil Analisis

Setelah dilakukan analisa, bisa diambil kesimpulan bahwa Oracle GoldenGate v11.0.0 belum bisa digunakan untuk dilakukan migrasi database dari MySQL menuju Oracle dikarenakan adanya masalah binary-log.

Binary log pada MySQL mengandung topik yang menggambarkan adanya perubahan database seperti operasi create table terhadap data pada tabel. Binary log juga mengandung topik untuk statement-statement yang berpotensi memberikan perubahan seperti DELETE. Sehingga apa yang harus kita lakukan? agar bisa dilakukan replikasi dari MySQL ke Oracle, Oracle perlu membaca binary log dari MySQL, tetapi pada akhirnya, migrasi belum berhasil dilakukan menggunakan Oracle GoldenGate karena error yang ada, sedangkan kenyataannya error tersebut tidak sesuai dengan keadaan yang ada seperti yang dijelaskan sebelumnya, yang dimana Oracle membaca bahwa binary log pada file /etc/my.cnf belum dilakukan enabled, tetapi kenyataanya sudah dilakukan enabled dan ada pada directory /var/lib/my,sql.

Dan fakta menariknya, ternyata belum ada pengguna di forum Oracle (http://forums.oracle.com) yang berhasil melakukan migrasi dari MySQL ke Oracle, sehingga disini kita menemukan ada masalah antara MySQL dan juga Oracle GoldenGate, yang bisa kita jabarkan sebagai berikut:

1. MySQL memang memiliki masalah dengan binary-log, di tengah keadaan masih dalam tahap development.
2. Oracle GoldenGate versi 11.0.0 baru diluncurkan, sehingga juga ada kemungkinan bahwa Oracle GoldenGate masih dalam pengembangan, dan seperti yang kita tahu software yang masih baru akan ada kemungkinan munculnya bug yang belum terselesaikan

3.6 Solusi untuk melakukan Migrasi dan Replikasi

Berdasarkan analisa permasalahan yang sudah dijabarkan sebelumnya, maka kita bisa mengambil kesimpulan dan solusi dalam mengatasi masalah migrasi dan replikasi yang ada, antara lain:

1. Migrasi dapat dilakukan menggunakan tool yang disebut Oracle *SQL Developer*. Keunggulan yang ada pada tool ini adalah merupakan produk yang berasal dari Oracle juga, gratis, dan mudah dalam penggunaannya. Namun, perusahaan tetap tidak bisa mengelak dari downtime ketika melakukan migrasi.
2. Sehingga, disini untuk permasalahan replikasi, disarankan dilakukan pengimplementasian replikasi menggunakan Oracle GoldenGate, dimana software ini menawarkan replikasi *bi-directional active-active*. Dengan konfigurasi ini, maka perusahaan mendapatkan beberapa *benefit*, yaitu:
    a. Konfigurasi ini memungknkan proses replikasi berlangsung secara dua arah antara dua database aktif. sehingga dapat meringankan beban yang ada dibandingkan jika hanya menggunakan satu database aktif untuk banyak sistem (*transaction load distribution)*.
    b. Salah satu database yang aktif bisa dijadikan sebagai database *backup* dimana bila terjadi kegagalan pada database satunya.
    c. Secara *overall*, konfigurasi ini memastikan ketersediaan data maksimum untuk suatu sistem, meringankan beban untuk tiap database, menyediakan *backup* untuk mengatasi kegagalan yang disengaja maupun yang tak disengaja.

Walau belum berhasil dilakukan pengimplementasian migrasi dari MySQL menuju Oracle, tetapi bisa dipastikan bahwa replikasi antar database Oracle bisa berjalan dengan baik sesuai solusi yang sudah dijabarkan di atas.

IV. Implementation

Implementasi replikasi pada PT. Metro Batavia disarankan menggunakan Oracle GoldenDate. Oracle GoldenDate adalah software untuk replikasi dan data-integrasi. Golden Gate pada awalnya yang dikembangkan oleh Software GoldenGate, akan tetapi pada tahun 2009 dipasarkan oleh Oracle Corporation, sehingga namanya berubah menjadi Oracle Golden Gate
Mengapa kita menggunakan Oracle GoldenDate pada implementasi replikasi di PT. Metro Batavia? karena jika dilihat dari

- Real Time Data Warehouse, biasanya status data warehouse yang telah dibangun pada H-1, atau data hari kemarin, dengan Oracle GoldenGate (OGG) bisa membuat sinkronisasi data dari OLTP/database staging menjadi lebih cepat.

- Migrasi data yang juga melibatkan beda database (Heterogeneous database) contohnya seperti migrasi data dari database Oracle ke DB2/MySQL/Oracle/Teradata maupun sebaliknya
- Pada proses pembangunan data warehouse nanti akan menghadapi data yang besar seperti 1 table OLTP berisi 900 juta records, sehingga untuk mengcapture 1 tabel tersebut ke table staging akan memerlukan waktu yang cukup lama, tetapi beda halnya jika langsung baca ke tabel OLTP tersebut maka akan mempengaruhi proses production, dengan OGG bisa untuk mengambil data perubahannya saja, sehingga untuk kasus 1 table tadi, pertama dilakukan initial load untuk tabel tersebut untuk mengcopy datanya, selanjutnya untuk membaca perubahannya OGG akan melakukan sinkronisasi secara otomatis dan sistem ini sangat cocok dengan PT. Metro Batavia.

V. Conclusion and Suggestions

Perancangan migrasi dan replikasi data yang dilakukan pada sistem hanggar bertujuan untuk meningkatkan performa dari database itu sendiri dikarenakan banyaknya jumlah transaksi yang ada serta dapat membangun konfigurasi aktif yang berguna ketika terjadi outage pada database tersebut.

Sehingga dari sini, dapat ditarik beberapa kesimpulan yaitu:

1. Masih ada beberapa masalah yang terjadi pada saat migrasi, yaitu pada binary log di MySQL dan pada Oracle ketika adanya pengaksesan status dari binar log yang padahal sudah dalam keadaan *enabled* tetapi terbaca dalam keadaan *disabled*.
2. Replikasi aktif-aktif (*bi-directional replication)* dilakukan untuk membangun suatu konfigurasi antar dua database agar kedua database dapat saling tersinkronisasi dan menghasilkan dua database yang memiliki set data yang identif. Replikasi ini menjamin adanya ketersediaan data pada sistem hanggar serta dapat mengurangi beban kerja yang ada pada *primary database*. Kemudian set data hasil dari replikasi (*secondary database)* dapat digunakan untuk menggantikan *primary database* apabila terjadi outage pada database tersebut.

Dengan adanya replikasi dan migrasi database dari MySQL ke Oracle, memudahkan PT. Metro Batavia dalam mengakses data sistem hanggar yang berjalan secara realtime dan mereka juga menggunakan replikasi data pada sistem hanggar tersebut supaya bisa menjaga kelangsungan proses bisnis mereka.

Proses replikasi database adalah proses menggandakan semua perubahan yang terjadi pada suatu database yang disebut master server ke server lain yang disebut slave server. Sedangkan proses migrasi database adalah proses untuk

mengkonversi data dari satu database ke database yang lain baik dalam environment yang sama maupun di dalam environment yang berbeda. Environment yang dimaksud adalah hardware dan software yang digunakan bersamaan dengan database.

Migrasi dapat dilakukan menggunakan tool yang disebut Oracle *SQL Developer*. Dan untuk replikasi, disarankan dilakukan pengimplementasian replikasi menggunakan Oracle GoldenGate, dimana software ini menawarkan replikasi *bi-directional active-active*. Dengan konfigurasi ini, maka perusahaan mendapatkan beberapa *benefit*

Dalam proses migrasi dan replikasi yang akan dilakukan, database yang digunakan adalah database sistem Hanggar, dimana database ini berisikan data-data yang berhubungan dengan kondisi pesawat, task yang dikerjakan, workorder, reminder, dan persediaan suku cadang untuk reparasi pesawat.

Sehingga, ada beberapa saran yang dapat diberikan untuk pengembangan lebih lanjut, yaitu:

1. Perusahaan disarankan untuk mendokumentasikan tabel-tabel pada setiap database yang ada serta alur bisnisnya agar dapat memudahkan para *developer* untuk merancang konfigurasi migrasi dan replikasi di masa yang akan datang. Hal ini didasarkan pada kesulitan saat mendapatkan struktur database pada sistem hanggar yang akan dilakukan migrasi dan replikasi data.
2. Perusahaan disarankan untuk melakukan konfigurasi penginstallan database Oracle dan Oracle GoldenGate mengikuti *best practice* untuk menghindari kesalahan-kesalahan yang mungkin terjadi.

Referensi:

- Ratna, A & Raymon. (2011). Analisis Migrasi Database dan Replikasi Bidirectional pada PT. Metro Batavia. (Skripsi S1, Universitas Bina Nusantara, 2011) Diakses dari http://library.binus.ac.id/Collections/ethesis_detail/2011-1-00251-IF
- Adam Jorgensen, Bradley Ball, Steven Wort, Ross LoForte, Brian Knight. (2014). Professional Microsoft SQL Server 2014 Administration. 1st E. Wrox Press. USA. ISBN: 9781118859131.