ANALISIS LOGGING DAN AUDITING AKSES DATABASE SQL MELALUI KONEKSI REMOTE AKSES MENGGUNAKAN ANYDESK

Ivan Joshua, Dian W. Chandra
Teknik Informatika, Universitas Kristen Satya Wacana
Jalan Diponegoro 52-60, Salatiga 50711, Indonesia
672019040@student.uksw.edu

ABSTRAK

Semakin banyak organisasi atau perusahaan mengadopsi praktik kerja jarak jauh dan mengandalkan koneksi *remote* untuk mengakses basis data SQL yang mengandung informasi penting mereka. Ini memunculkan kebutuhan untuk mengamankan, memantau, dan mengaudit aktivitas akses *database* yang dilakukan melalui koneksi *remote*. Keamanan, integritas, dan kepatuhan data menjadi prioritas utama dalam menghadapi tantangan ini untuk menjaga kesuksesan bisnis di era digital yang terus berkembang. Permasalahannya dalam pertanyaan penelitian adalah bagaimana cara memastikan bahwa data yang diinputkan ke *database* SQL melalui koneksi *remote* menggunakan *anydesk* sudah benar dan tidak mengalami redudansi data atau manipulasi yang tidak sah?. Penelitian ini menggunakan metode NDLC (*Network Development Life Cycle*) yang merupakan teknik analisis terstruktur yang digunakan untuk merencanakan dan mengelola proses pengembangan sistem. Hasil penelitian mendapatkan 2 bagian log yang tercatat yaitu bagian *login* dan bagian *database* atau *transaction log*.

Kata kunci : *Remote;* Basis data*; Anydesk; Log*

1. PENDAHULUAN

Dalam lingkungan bisnis yang semakin terhubung dan didorong oleh mobilitas, semakin banyak organisasi mengadopsi praktik kerja jarak jauh [1] dan mengandalkan koneksi *remote* untuk mengakses basis data SQL yang mengandung informasi penting mereka. Sebagai dampak dari pergeseran ini, koneksi *remote* melalui perangkat seperti *Anydesk* telah menjadi norma, memungkinkan tim dan individu untuk mengelola, memanipulasi, dan mengakses data bisnis yang sangat berharga dari hampir setiap lokasi di dunia [2]. Namun, bersamaan dengan kebebasan dan fleksibilitas yang ditawarkan oleh koneksi jarak jauh ini, ada tantangan serius yang muncul terkait dengan keamanan, pengawasan, dan kontrol terhadap aktivitas yang terjadi dalam basis data SQL [3].

Ketika data bisnis yang kritis mengalir melalui koneksi *remote*, perluasan keberhasilan bisnis dan menjaga integritas serta kerahasiaan data menjadi prioritas utama [4]. Oleh karena itu, Analisis *Logging* dan *Auditing* Akses menjadi hal yang sangat penting, karena mereka memungkinkan organisasi untuk mendeteksi ancaman keamanan, mengidentifikasi potensi masalah atau kebijakan yang dilanggar, serta memastikan bahwa aktivitas akses *database* yang dilakukan melalui koneksi *remote* tercatat dan dapat dianalisis [5].

Keamanan dan integritas data menjadi semakin krusial dalam lingkungan bisnis saat ini. *Logging* memungkinkan catatan perubahan data, sementara *auditing* memantau aktivitas pengguna. Ini membantu organisasi mendeteksi ancaman dan memitigasi risiko [6]. *Logging* dan *auditing* bukan hanya tentang keamanan data, tetapi juga tentang akuntabilitas. Memahami bagaimana data diproses dan siapa yang bertanggung jawab sangat penting. Oleh karena itu, konsep ini tidak boleh diabaikan dalam pengelolaan *database* SQL [7]. Penting bagi perusahaan untuk mengembangkan kemampuan dalam melakukan analisis dan penyusunan laporan secara efektif, efisien, dan terintegrasi dari sistem informasi [8]. Banyak perusahaan menginginkan proses analisis diselesaikan dalam waktu sesingkat mungkin [9]. Sistem Informasi telah ada dan digunakan dalam praktik sejak awal era revolusi digital. Meskipun informasi telah digunakan dalam proses bisnis, isu keamanan informasi awalnya terfokus pada otentikasi dan otorisasi. Meskipun kedua konsep tersebut penting untuk melindungi informasi, mereka kurang efektif dalam mendukung penyelidikan. Untuk melacak jejak akses ke *database* dan sistem, log *file* telah diusulkan sebagai solusi.

Meskipun log *file* pertama-tama dirancang untuk tujuan pemulihan transaksi, penggunaan mereka telah berkembang untuk mendukung penyelidikan. Untuk menyelidiki transaksi yang telah terjadi, penggunaan *database auditing* dapat menjadi opsi yang lebih relevan [10]. Melakukan audit perubahan data pada database sangat penting untuk mengidentifikasi tindakan yang mencurigakan, memastikan kualitas data yang terjaga, dan meningkatkan efisiensi kinerja sistem [11]. *Logging* adalah fondasi untuk audit dan investigasi keamanan. Dalam database SQL, informasi seperti siapa yang mengakses data dan kapan data dimodifikasi sangat penting. Inilah mengapa pemahaman mendalam tentang *logging* dan *auditing* sangat dibutuhkan [12]. Dengan demikian, latar belakangnya menyoroti bahwa penggunaan teknologi seperti *Anydesk* untuk mengelola basis data SQL secara *remote* memunculkan tantangan yang memerlukan pendekatan proaktif untuk mengamankan, memantau, dan mengaudit aktivitas akses. Ini bukan hanya tentang memungkinkan akses yang efisien, tetapi juga tentang memastikan

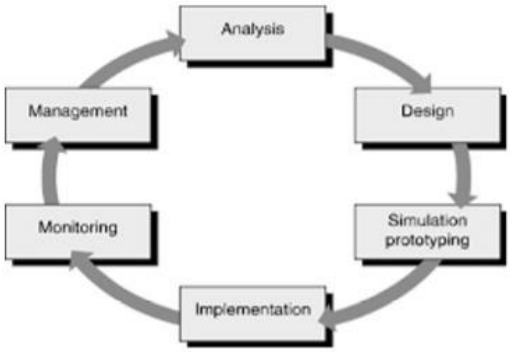
keamanan, integritas, dan kepatuhan data, yang merupakan unsur kunci dalam keberhasilan dan keberlanjutan bisnis di era digital yang terus berkembang.

2. TINJAUAN PUSTAKA

Penelitian sebelumnya telah menginvestigasi aspek *database auditing*, dengan beberapa di antaranya mengembangkan teori-teori untuk mendukung proses *auditing*. Penelitian dengan judul *Database Auditing Design on Historical Data* mengusulkan beberapa metode untuk melakukan manajemen historikal data *auditing* pada database, seperti audit berbasis baris, audit berbasis kolom, audit dengan tabel log dan audit berbasis semi-struktur. Pada dasarnya, auditing merupakan tindakan pemantauan dan pencatatan aktivitas pengguna tertentu dalam database. Hasil dari proses auditing ini dikenal sebagai audit trail, yang berisi catatan tentang peristiwa-peristiwa yang terjadi dalam database. Setiap sistem manajemen basis data (DBMS) memiliki batasan dalam kapasitasnya untuk merekam dan menyimpan catatan aktivitas, yang dapat bervariasi sesuai dengan sistem tersebut. [13] Seorang peneliti bernama Meg Coffin Murray menjelaskan bahwa database *auditing* memiliki kemampuan untuk mengidentifikasi pelaku yang mengakses *database*, jenis kegiatan yang dilakukan, serta data yang mengalami perubahan.

Melakukan audit terhadap aktivitas dan akses *database* dapat membantu dalam menemukan potensi masalah keamanan dalam sistem basis data, dan memungkinkan penyelesaiannya dengan cepat. Sebagai fungsi utama, audit memainkan peran kunci dalam memastikan kepatuhan terhadap peraturan dan aturan karena audit melibatkan pemeriksaan dokumen terkait tindakan, praktik, serta perilaku bisnis atau individu[14]. Kunci keberhasilan dalam proses *auditing* adalah kemampuan untuk melacak perubahan data, jenis operasi modifikasi yang dilakukan, dan waktu terjadinya operasi tersebut melalui data historis. Data historis dapat direpresentasikan dalam sebuah *database* relasional dengan berbagai teknik, seperti penggunaan tabel terpisah untuk mencatat riwayat, log transaksi, atau data multidimensional. Untuk menjaga data historis dalam konteks *auditing*, ada beberapa teknik yang dapat diimplementasikan, seperti *auditing* berbasis baris (*Row-based Auditing*) atau *auditing* berbasis kolom (*Column-based Auditing*) [8]. Tiga teknik *auditing* pertama dapat diterapkan dengan *database* relasional, sedangkan audit berbasis semi struktur diterapkan dengan menggunakan *extension* dari mesin database relasional dan teknologi XML (*extensions markup language*) seperti IBM DB2 9.5, Oracle 10g, dan MS SQL Server.

3. METODE PENELITIAN



Gambar 1. Metode NDLC

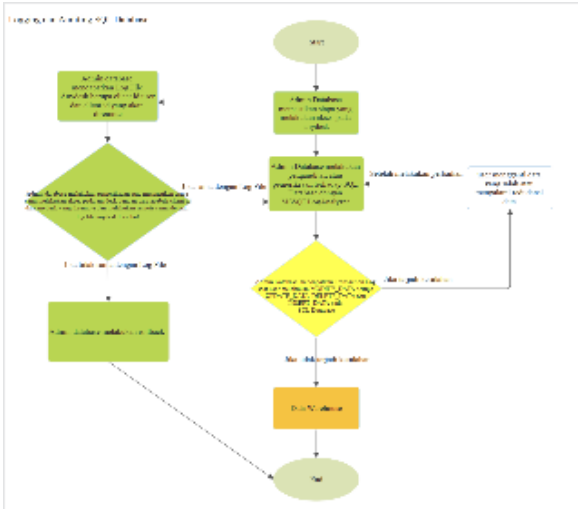
Penelitian ini menggunakan metode NDLC (*Network Development Life Cycle*) yang merupakan teknik analisis terstruktur yang digunakan untuk merencanakan dan mengelola proses pengembangan sistem. NDLC memiliki enam tahap yaitu *Analysis, Design, Simulation Prototyping, Implementation, Monitoring, dan Management*.

Pada tahap awal ini akan dilakukan proses analysis yang nanti akan dilakukan analisa alur auditing meliputi 3 kondisi data yaitu insert, update, dan delete, cara yang gunakan adalah memeriksa setiap data yang didapat dari data-data sebelumnya, maka perlu dilakukan analisa data tersebut untuk masuk ke tahap berikutnya

Pada tahap kedua ini akan dilakukan proses *design* dari data-data yang didapatkan sebelumnya, tahap *Design* ini akan membuat gambar alur kerja *logging* dan *auditing*, diharapkan dengan gambar ini akan memberikan gambaran seutuhnya dari kebutuhan yang ada. Design bisa berupa cara kerja logging seperti jejak audit dari *log on* dan *log off*, serta mencatat semua upaya *login* yang gagal, dan cara kerja melalui aktivitas DML (*Data Manipulation Languange*) perubahan yang terjadi, baik nilai lama maupun nilai baru, dapat terekam. Biasanya hasil dari *design* berupa:

Hasil *Screenshot log file* (untuk melihat pencatatan *login*) dan MSSQLLOGANALYZER (untuk melihat waktu, nama objek, operasi dan apa yang dilakukan user (INSERT Command, UPDATE Command, dan DELETE Command)).

Rollback akan dilakukan apabila dari audit didapatkan akses dari pihak yang tidak diketahui, maka perubahan yang dibuat oleh pelaku akan dikembalikan seperti semula. Kalau itu pembacaan informasi maka pemilik data/user diberikan peringatan bahwa datanya telah diakses. Tetapi kalau perubahan dari pihak yang diijinkan maka dilihat apakah perubahan yang dilakukan benar atau tidak (dilakukan audit)



Gambar 2. Alur Kejar *Logging* Dan *Auditing*

Pada tahap ketiga ini akan dilakukan proses *simulation prototyping* yang artinya beberapa *database* administrator akan membuat dalam bentuk simulasi dengan bantuan *Tools* MSSQLLOGANALYZER ini digunakan untuk melihat waktu, nama objek, operasi dan apa yang dilakukan user (INSERT Command, UPDATE Command, dan DELETE Command.

Pada tahap keempat ini akan dilakukan proses implementation. Dalam implementasi database administrator akan menerapkan semua yang telah direncanakan dan di design sebelumnya. Ada beberapa masalah-masalah yang sering muncul pada tahapan ini, diantaranya:

a. Keterbatasan penyimpanan Log *auditing database* SQL,
b. Terlalu banyak atau terlalu sedikit *logging* dan *auditing* dapat menjadi masalah. Terlalu banyak *logging* dapat membanjiri sistem dengan data yang tidak perlu, sementara terlalu sedikit *logging* mungkin gagal mendeteksi aktivitas yang mencurigakan atau tidak sah.

Pada Tahap kelima ini akan dilakukan proses *monitoring*. Tahap ini paling penting dikarenakan proses *logging* dan *auditimg* dapat berjalan sesuai dengan keinginan dan tujuan awal dari user pada tahap awal *analysis*, maka perlu dilakukan kegiatan *monitoring*.

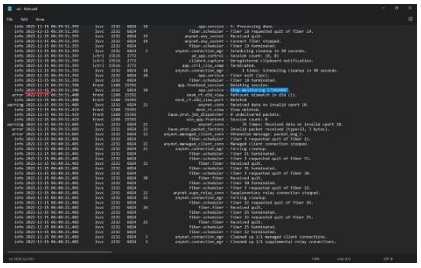
Monitoring bisa berupa melakukan pengamatan pada ;

1. Infrastruktur *hardware* : dengan mengamati kondisi *reliability* / kehandalan sistem yang telah dibangun dan (*reliability* = *performance* + *availability* + *effective*),
2. Memperhatikan alurnya *auditing* di *database* (*insert*, *update*, dan *delete*)
3. Pendekatan yang dilakukan adalah pendekatan *logging* dan *Auditing*, dengan pendekatan ini *database administrator* dapat memastikan bahwa seluruh data-data telah tersimpan, tersedia dan terjaga keamanannya dengan baik.

Pada Tahap terakhir ini akan dilakukan proses *management*, salah satu yang menjadi perhatian khusus adalah masalah *Policy*, kebijakan perlu dibuat untuk membuat / mengatur agar sistem yang telah dibangun dan berjalan dengan baik dapat berlangsung lama dan unsur *Reliability* terjaga. *Policy* akan sangat tergantung dengan kebijakan *level management* dan strategi bisnis perusahaan tersebut. *Auditor* IT sebisa mungkin harus dapat mendukung atau *alignment* dengan strategi bisnis perusahaan.

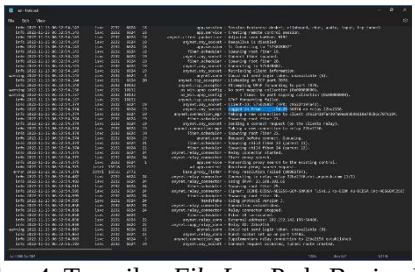
4. HASIL DAN PEMBAHASAN

Jadi hasilnya, saya mendapatkan 2 bagian log yang tercatat yaitu bagian *login* dan bagian *database* atau *transaction log*. Untuk bagian *login* dapat dilihat di log *file anydesk* (*file ad*) terdapat *client* id pengguna dan *client* id yang akan di*remote*, serta untuk bagian *database* dapat dilihat dengan menggunakan MSSQLLogAnalyzer (dapat tercatat saat melakukan MODIFY_DATA berupa UPDATE_DATA, DELETE_DATA dan INSERT_DATA untuk RedoSQL dan UndoSQL setiap transaksi). Serta, menjelaskan proses *rollback* yang dilakukan apabila user yang *login* tidak sah, karena pada penelitian ini tidak ada skenario *login* yang tidak sah.



Gambar 3. Tampilan *File Log* Pada Bagian *Login*

Pada gambar 3 terdapat pesan **"*stop monitoring* 575028887**" artinya Pesan "*stop monitoring* 575028887" dalam *log AnyDesk* merupakan catatan tentang penghentian pemantauan atau *monitoring* terhadap sesuatu yang teridentifikasi dengan kode atau ID "575028887".



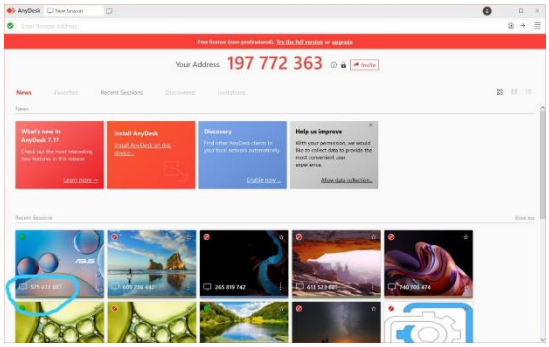
Gambar 4. Tampilan *File Log* Pada Bagian *Login*

Pada gambar 4 terdapat pesan **"*logged in from* **.**.94.26:56074 *on relay* 22be2156"** artinya informasi tentang aktivitas *login* atau masuk ke sesi jarak jauh dari alamat IP tertentu ke *relay* yang disediakan oleh layanan *AnyDesk*.

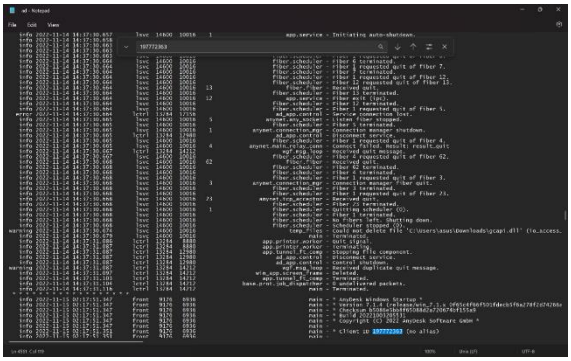
Untuk pembahasan lebih rinci :

- "*logged in from* 36.95.94.26:56074"
Ini mengindikasikan bahwa ada aktivitas *login* atau masuk ke aplikasi *AnyDesk* dari alamat IP spesifik yaitu **"**.**.94.26"** pada *port* **"56074".** IP *address* ini adalah alamat dari mana koneksi tersebut berasal.
- "*on relay* 22be2156"
Menunjukkan bahwa koneksi atau sesi yang terbentuk menggunakan *relay* dengan ID, yaitu **"22be2156".** *Relay* ini adalah bagian dari infrastruktur yang digunakan oleh *AnyDesk* untuk menghubungkan klien yang terhubung secara langsung.

Jadi, pesan tersebut memberikan informasi tentang asal koneksi (alamat IP) serta informasi *relay* yang digunakan untuk menghubungkan sesi jarak jauh melalui *AnyDesk*. Ini adalah bagian dari pencatatan aktivitas yang berguna untuk melacak masuknya pengguna ke sesi jarak jauh menggunakan *AnyDesk*.



Gambar 5. Tampilan *Anydesk* Yang Menunjukkan *Client* ID Yang Melakukan *Remote* Dan Yang Di*remote* (yang diberi lingkaran)



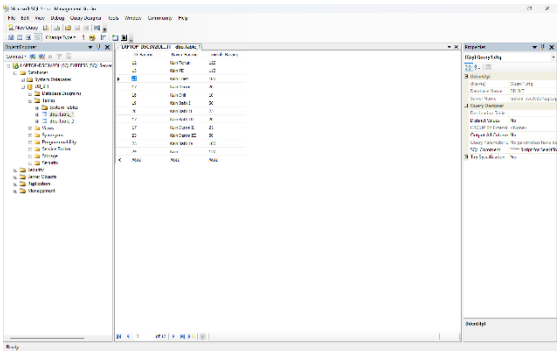
Gambar 6. Tampilan *File* Log Pada Bagian *Login*

Pada gambar 3 dan 5 merupakan file log pada bagian *login* sehingga mendapatkan log berupa kata *Stop Monitoring* **575028887** adalah id dari *anydesk* yang di*remote* dan *Client ID* **19777363** adalah id dari *anydesk* laptop saya (yang melakukan remote).

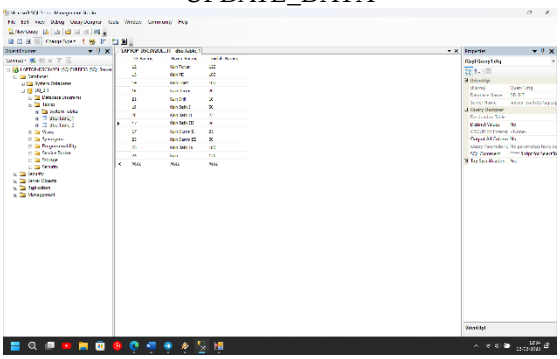
Dan juga terdapat pesan **"*Client* ID 19777363 (no alias)"** menunjukkan informasi tentang klien atau pengguna yang terhubung ke layanan *AnyDesk* dengan **ID klien 19777363** dan tanpa alias (nama pengenal tambahan).
Untuk pembahasan lebih rinci :
- "*Client* ID"
Merupakan identifikasi unik yang diberikan kepada pengguna atau klien yang terhubung ke layanan *AnyDesk*. Dalam hal ini, **"19777363"** adalah ID yang diberikan kepada klien tersebut.
- "(no alias)"
Menandakan bahwa klien tersebut tidak memiliki alias atau nama tambahan yang terkait dengannya dalam sistem. Biasanya, pengguna dapat memberikan nama alias atau label untuk mengidentifikasi klien mereka secara lebih mudah dalam daftar koneksi atau riwayat log.

Pesan ini mencatat bahwa klien dengan **ID 19777363** terhubung ke layanan *AnyDesk*, tetapi tidak memiliki alias yang terkait dengannya. Informasi ini mungkin dicatat untuk melacak aktivitas atau koneksi yang dilakukan oleh klien tersebut.

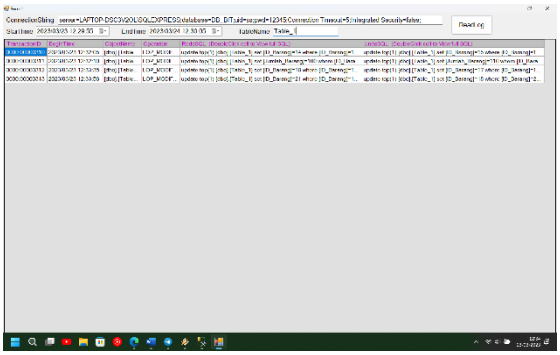
Tampilan Log Bagian *Database* (*Transaction Log*) Saat melakukan operasi *UPDATE*_DATA



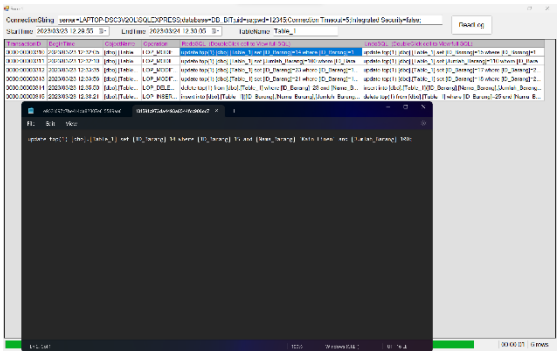
Gambar 7. Tampilan Microsoft SQL *Server Management Studio* Sebelum Melakukan *UPDATE*_DATA



Gambar 8. Tampilan Microsoft SQL *Server Management Studio* sesudah melakukan *UPDATE*_DATA



Gambar 9. Tampilan MSSQLLOGANALYZER Memperlihatkan Telah Melakukan *UPDATE*_DATA



Gambar 10. Tampilan Perintah SQL (*Querry*nya) Untuk Melakukan Operasi *Update* Data

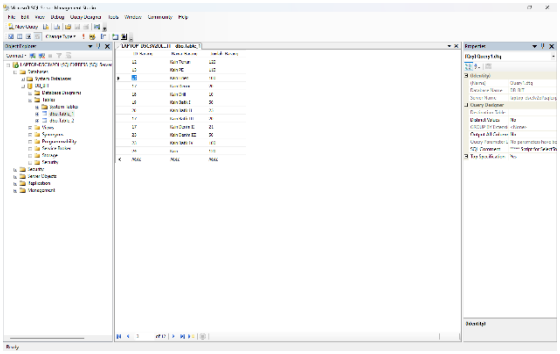
Pada gambar 10 merupakan MSSQLLOGANALYZER memperlihatkan telah melakukan *UPDATE*_DATA pada baris pertama dengan *querry* ***update top*(1) [dbo].[*Table* 1] *set* [ID_Barang]=14 *where* [ID_Barang]=15 *and* [Nama_Barang]='Kain Linen' *and* [Jumlah_Barang]=100;** dilakukan pada tanggal 23 Maret 2023 jam 12:32.

Contoh perintah SQL 1 untuk melakukan operasi *update* data adalah sebagai berikut:

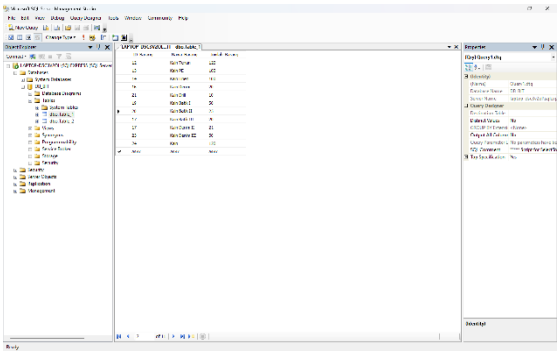
***update top*(1) [dbo].[*Table* 1] set [ID_Barang]=14 *where* [ID_Barang]=15 and [Nama_Barang]='Kain Linen' and [Jumlah_Barang]=100;**

Dalam kode perintah 1,***UPDATE TOP*(1) [dbo].[*Table* 1]:** Ini adalah perintah ***UPDATE*** yang digunakan untuk mengubah satu baris pertama dari tabel **"[dbo].[*Table* 1]".** Penggunaan ***TOP*(1)** menentukan bahwa hanya satu baris pertama yang memenuhi kondisi yang akan diubah. ***SET* [ID_Barang] = 14:** Ini adalah bagian dari perintah yang akan mengubah nilai kolom **[ID_Barang]** menjadi **14** pada baris yang memenuhi kondisi tertentu.***WHERE* [ID_Barang] = 15 *AND* [Nama_Barang] = 'Kain Linen' *AND* [Jumlah_Barang] = 100:** Ini adalah klausa ***WHERE*** yang mengatur kondisi yang harus dipenuhi oleh baris yang akan diubah. Hanya baris yang memenuhi semua kondisi ini yang akan mengalami perubahan. Dalam kasus ini, hanya baris yang memiliki **[ID_Barang] sama dengan 15, [Nama_Barang] sama dengan 'Kain Linen', dan [Jumlah_Barang] sama dengan 100** yang akan diubah.

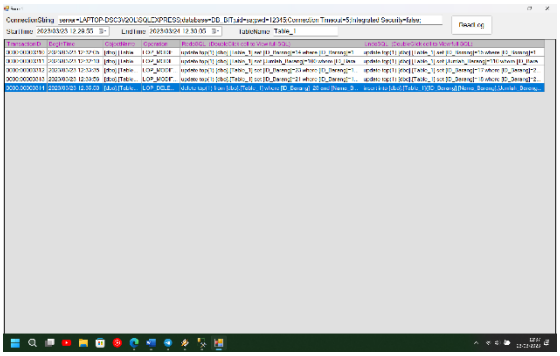
Saat melakukan operasi *DELETE*_DATA



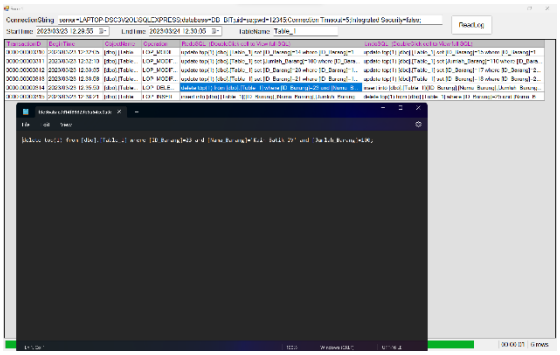
Gambar 11. Tampilan Microsoft SQL *Server Management Studio* Sebelum Melakukan *DELETE*_DATA (yang mau dihapus adalah baris nama_barang=Kain Batik IV)



Gambar 12. Tampilan Microsoft SQL *Server Management Studio* Sesudah Melakukan *DELETE*_DATA (yang mau dihapus adalah baris nama_barang=Kain Batik IV)



Gambar 13. Tampilan MSSQLLOGANALYZER Memperlihatkan Telah Melakukan *DELETE*_DATA



Gambar 14. Tampilan Perintah SQL (*Querry*nya) Untuk Melakukan Operasi *Delete* Data.

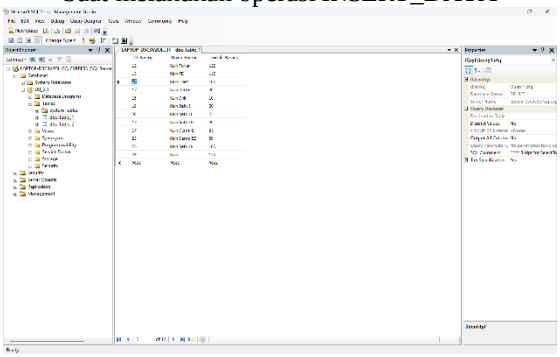
Pada gambar 14 merupakan MSSQLLOGANALYZER memperlihatkan telah melakukan *DELETE*_DATA pada baris kelima dengan *querry delete top*(1) *from* [dbo].[Table_1] *where* [ID_Barang]=23 and [Nama_Barang]='Kain Batik IV'and [Jumlah_Barang]=100; dilakukan pada tanggal 23 Maret 2023 jam 12:35.

Contoh perintah SQL 2 untuk melakukan operasi *delete data* adalah sebagai berikut:

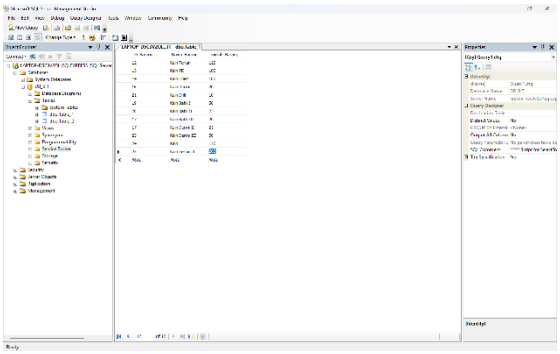
delete top(1) *from* [dbo].[*Table*_1] *where* [ID_Barang]=23 *and* [Nama_Barang]='Kain Batik IV'and [Jumlah_Barang]=100;

Dalam kode perintah 2,*DELETE TOP*(1) *FROM* [dbo].[*Table*_1]: Ini adalah perintah *DELETE* yang digunakan untuk menghapus satu baris pertama dari tabel "[dbo].[*Table*_1]". Penggunaan *TOP*(1) menentukan bahwa hanya satu baris pertama yang memenuhi kondisi yang akan dihapus. *WHERE* [ID_Barang] = 23 *AND* [Nama_Barang] = 'Kain Batik IV' *AND* [Jumlah_Barang] = 100: Ini adalah klausa *WHERE* yang mengatur kondisi yang harus dipenuhi oleh baris yang akan dihapus. Hanya baris yang memenuhi semua kondisi ini yang akan dihapus. Dalam kasus ini, hanya baris dengan [ID_Barang] sama dengan 23, [Nama_Barang] sama dengan 'Kain Batik IV', dan [Jumlah_Barang] sama dengan 100 yang akan dihapus.

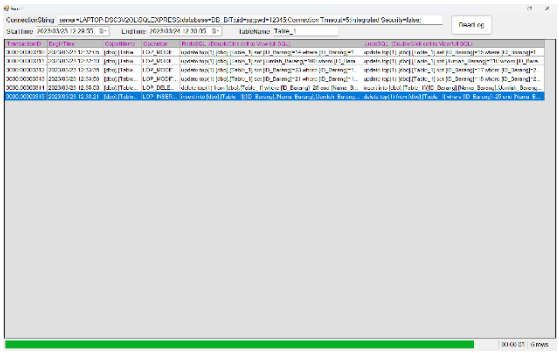
Saat melakukan operasi *INSERT*_DATA



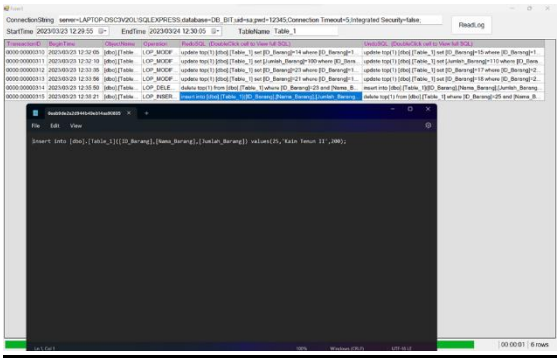
Gambar 15. Tampilan Microsoft SQL *Server Management Studio* Sebelum Melakukan *INSERT*_DATA



Gambar 16. Tampilan Microsoft SQL *Server Management Studi*o Sesudah Melakukan *INSERT*_DATA (yang ditambahkan baris baru adalah baris nama_barang=Kain Tenun II)



Gambar 17. Tampilan MSQLLOGANALYZER Memperlihatkan Telah Melakukan *INSERT*_DATA



Gambar 18. Tampilan Perintah SQL (*Querry*nya) Untuk Melakukan Operasi *Insert* Data

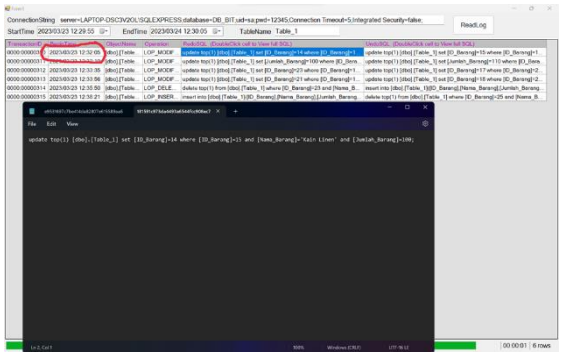
Pada gambar 18 merupakan MSSQLLOGANALYZER memperlihatkan telah melakukan *INSERT*_DATA pada baris keenam dengan *querry* ***Insert into*** **[dbo].[*Table*_1]([ID_Barang], [Nama_Barang], [Jumlah Barang])** ***values*** **(25,'Kain Tenun II',200);** dilakukan pada tanggal 23 Maret 2023 jam 12:38.

Contoh perintah SQL 3 untuk melakukan operasi *insert data* adalah sebagai berikut:

Insert into **[dbo].[*Table*_1]([ID_Barang], [Nama_Barang], [Jumlah Barang])** ***values*** **(25,'Kain Tenun II',200);**

Dalam kode perintah 3, ***INSERT INTO*** **[dbo].[*Table*_1]:** Ini adalah perintah *INSERT* yang digunakan ketika ingin memasukkan data ke dalam

tabel **"[dbo].[*Table*_1]". ([ID_Barang], [Nama_Barang], [Jumlah Barang]):** Ini adalah daftar kolom yang Anda ingin masukkan data ke dalamnya. Anda menyebutkan kolom-kolom ini dalam urutan yang sesuai dengan nilai-nilai yang akan Anda masukkan. ***VALUES* (25, 'Kain Tenun II', 200):** Ini adalah nilai-nilai yang akan dimasukkan ke dalam kolom-kolom yang Anda sebutkan sebelumnya. Dalam kasus ini, anda menyisipkan nilai **25 ke dalam kolom [ID_Barang], 'Kain Tenun II' ke dalam kolom [Nama_Barang], dan 200 ke dalam kolom [Jumlah Barang].**



Gambar 19. Menunjukkan Waktu Saat *User* Melakukan *Querry* Pada *Database*

Dalam analisis hasil kapan seorang *user* mulai *login* di dalam suatu komputer = *User* akan memulai *login* saat mulai melakukan akses pada anydesk dan dalam rentang waktu tersebut terdapat berapa *querry* yang dilakukan = 12 baris yang terdiri dari 6 *Redo*SQL dan *Undo*SQL

5. KESIMPULAN

Berdasarkan hasil penelitian dan pembahasan yang telah dilakukan tentang Analisis *Logging* dan *Auditing* Akses *Database* SQL melalui Koneksi *Remote* Akses menggunakan *Anydesk* maka dapat disimpulkan bahwa dengan menggunakan *logging* dan *Auditing* mendapatkan 2 bagian log yang tercatat yaitu bagian *login* dan bagian *database* atau *transaction log* .Untuk bagian *login* dapat dilihat di *log file anydesk* (*file ad*) terdapat *client* id pengguna dan *client* id yang akan di*remote*, serta untuk bagian *database* dapat dilihat dengan menggunakan MSSQLLogAnalyzer (dapat tercatat saat melakukan *MODIFY*_DATA berupa *UPDATE*_DATA, *DELETE*_DATA dan *INSERT*_DATA untuk *Redo*SQL dan *Undo*SQL setiap transaksi). Dengan alat atau *tools* ini dapat memudahkan dan menjawab tantangan yang memerlukan pendekatan proaktif untuk mengamankan, memantau, dan mengaudit aktivitas akses. Ini bukan hanya tentang memungkinkan akses yang efisien, tetapi juga tentang memastikan keamanan, integritas, dan kepatuhan data, yang merupakan unsur kunci dalam keberhasilan dan keberlanjutan bisnis di era digital yang terus berkembang.

DAFTAR PUSTAKA

[1] A. A. D. Lestari, "Peran Teknologi dalam Perubahan Bisnis di Era Globalisasi," Universitas Cendekia Mitra Indonesia, vol. 01, no. 21, pp. 113–120. 2022,.

[2] R. D Fahlepi, "Implementasi Sistem Akses Komputer Jarak Jauh Dengan Penerapan Anydesk," Universitas Bina Sarana Informatika, vol.19, pp. 90–110. 2022.

[3] Bertino, E. Database Auditing and Security: Challenges and Opportunities.IEEE Security & Privacy,vol. 7, no. 1, pp. 53–68. 2004

[4] Xiong, L., & Liu, L. Secure Logging Mechanisms for Databases. ACM Computing Surveys (CSUR), vol. 13, no. 1, pp. 104–116. 2008.

[5] Guimaraes, M. New challenges in teaching databasesecurity. In Proceedings of the 3rd Annual Conference on information Security Curriculum Development Kennesaw,Georgia, vol. 7, no. 5, pp. 90–99 September, 2006.5

[6] Sandhu, Ravi S. "Security in Database Systems."ACM Computing Surveys (CSUR) 26, no. 4 (1994): 355-387.

[7] Masud, Mohammad M., Latifur Khan, dan Bhavani Thuraisingham. "A Survey of Database Security Techniques." ACM Computing Surveys (CSUR) 42, no. 2 (2010): 1-64.

[8] W. Wisswani, "Penerapan Hybrid Slowly Change Dimension Untuk Nearly Realtime Data Warehouse," Lontar Komput., vol. 4, no. 1, 2013

[9] S. A, M. Sukarsa, and W. B, "Pembentukan Data Mart Menggunakan Metode Generalization," Lontar Komput., vol. 7, no. 3, 2016.

[10] N. Waraporn, "Database Auditing Design on Historical Data," Proc. Second Int. Symp. Netw. Netw. Secur. vol. 2, no. 3, pp. 249–260,, 2010.

[11] W. Lu and G. Miklau, "Auditing a Database Under Retention Restrictions,"IEEE Int. Conf.Data Eng., vol. 20, no. 1, pp. 117–127. 2009.

[12] Shenoi, Sujeet, dan Clare J. Hooper. "Auditing in Database Systems: State of the Art." ACM Computing Surveys (CSUR) 33, no. 3 (2001): 273-328.

[13] M. C. Muray, "Database Security: What Students Need to Know," J. Inf.Technol. Educ.Innov. Pract., vol. 7, no. 7, pp. 100-110 Oktober 2010.

[14] L. Yang, "Teaching Database Security and Auditing," Proc. 40th ACM Tech. Sympo-sium Comput. Sci. Educ. vol. 1, no. 2, p. 11,, 2009.

[15]