

Pengembangan Program Alternatif untuk Proses Konsolidasi Multiple Database Menggunakan Pandas dan MongoDB

Joshe Forgaritenzo, ArgoWibowo*, Katon Wijana

Fakultas Teknologi Informasi, Sistem Informasi, Universitas Kristen Duta Wacana, Yogyakarta, Indonesia

¹joshe.forgaritenzo@si.ukdw.ac.id, ²*argo@staff.ukdw.ac.id, ³katony@staff.ukdw.ac.id

Email Penulis Korespondensi: argo@staff.ukdw.ac.id*

Submitted: 04/11/2024; Accepted: 26/07/2025; Published: 08/09/2025

Abstrak— E-Commerce merupakan salah satu bidang bisnis yang sangat besar di Indonesia yang menghasilkan transaksi daring yang besar jumlahnya, sehingga ribuan hingga ratusan ribu data harus dikelola setiap harinya oleh pihak perusahaan melalui proses konsolidasi. Konsolidasi merupakan sebuah proses penggabungan data antar dua database. Perusahaan yang diteliti untuk mengelola proses konsolidasi menggunakan program pihak ketiga bernama Pentaho, namun program ini sering mengalami maintenance sehingga mengganggu proses bisnis yang berjalan. Penelitian ini dilakukan untuk membuat sebuah program alternatif yang dapat digunakan ketika Pentaho mengalami kendala. Program yang dikembangkan memanfaatkan proses loading yang merupakan metode dalam dunia pengelolaan database, dimana data yang dimiliki kemudian dimasukkan ke dalam database tujuan. Pengembangan program ini akan memanfaatkan library python pandas dan database relational serta NoSQL untuk melakukan proses loading dan proses konsolidasi. Penelitian ini akan mencoba menganalisis dan membuat program berjalan dengan lebih efisien dan memberikan pengembangan agar proses loading dan konsolidasi secara keseluruhan dapat menjadi lebih baik. Rata-rata proses loading program hasil pengembangan yang menggunakan database relational menunjukkan peningkatan sekitar 8% atau 20 - 30 detik lebih cepat untuk data berjumlah sekitar 500.000. Pengembangan proses loading menggunakan database NoSQL menunjukkan adanya peningkatan sekitar 6,5 - 9,6% untuk jumlah data yang berkisar dari 20 - 500.000 data. Proses ini juga menunjukkan peningkatan sekitar 17,5% dari program yang digunakan perusahaan sebelumnya untuk proses 500.00 data.

Kata Kunci: Loading Data; Konsolidasi; Pandas Library; Database; E-Commerce; Alternatif

Abstract— E-Commerce is one of the largest business sectors in Indonesia, generating a significant number of online transactions, thus requiring the management of thousands to hundreds of thousands of data points daily by companies through a consolidation process. Consolidation is a process that merge two or more databases. The company under study uses a third-party program called Pentaho for database management, particularly for the consolidation process. However, frequent maintenance issues with Pentaho can and will disrupt business operations. This research aims to develop an alternative program to be used when the third-party software encounters problems. The developed program utilizes the Loading process for the consolidation, a common approach in database management where specific data is loaded into the target database. The development of this program will leverage the Python library Pandas and relational and NoSQL databases to do its main method, the loading process. This study seeks to analyze and ensure the alternative program operates with greater efficiency and identifies enhancements to improve the overall consolidation process. On average, the loading process using the program that uses relational database is showing an increase of about 8% or about 20-30 seconds faster for data of about 500.000. This development for loading process that uses a NoSQL database shown a improvement of about 6.5 - 9.6% for the amount of data that ranged from 20 - 500.000 data. It also showed an improvement of about 17.5% from the program that the company used before for processing 500.000 data.

Keywords: Data Loading; Consolidation; Pandas Library; Database; E-Commerce; Alternative

1. PENDAHULUAN

Melalui konferensi pers yang dilakukan oleh Bank Indonesia disampaikan bahwa jumlah transaksi e-commerce di Indonesia pada tahun 2023 mencapai angka 3,71 miliar [1]. Volume transaksi yang besar ini secara tidak langsung mempengaruhi jumlah dan variasi data yang dikelola oleh aplikasi perusahaan ini. Database untuk aplikasi ini dikelola oleh tim software development perusahaan dan secara berkala dikonsolidasi menuju database pusat yang digunakan untuk berbagai keperluan baik dari sisi manajerial maupun sisi teknis aplikasi ini sendiri. Data-data pada database ini meliputi antara lain data stok, produk, promo, dan data-data lain yang diperlukan untuk aplikasi.

Perusahaan yang diteliti memiliki toko retail dan grosir yang tersebar di beberapa kota besar dan perusahaan ini memiliki aplikasi yang memudahkan user dalam membeli barang dari tokonya untuk memenuhi kebutuhan sehari-hari. Aplikasi yang digunakan sebagai objek penelitian adalah aplikasi yang digunakan untuk membeli barang pada toko khusus grosir perusahaan. Aplikasi pada toko cabang grosir memudahkan user untuk berbelanja dengan menggunakan internet dan perangkat genggam serta dengan memberikan jasa pengiriman langsung ke lokasi user. Selain sebagai aplikasi penyedia barang, aplikasi juga digunakan sebagai media promosi dan penawaran promo yang terkadang hanya tersedia melalui aplikasi.

Salah satu permasalahan yang terjadi adalah ketika proses konsolidasi database yang berukuran besar dan bersirkulasi cepat ini mengalami kendala akibat useran tools eksternal yaitu Pentaho yang sering mengalami proses maintenance dan timeout. Proses yang perlu dilakukan secara berkala juga menambah permasalahan yang muncul

dengan kendala tools yang ada. Penyimpanan data pada berbagai database yang berbeda juga dapat menimbulkan proses pengambilan data menjadi tugas yang rumit karena untuk mengumpulkan data diperlukan akses untuk sumber data yang berbeda serta rentan akan kesalahan dan dapat memakan waktu yang lama [2] selain itu useran database yang banyak dapat menguras sumber daya yang lebih banyak baik dari sisi keuangan hingga tenaga [3]. Beberapa permasalahan tersebut menjadi kendala yang menghambat proses analisis serta pengelolaan aplikasi. Permasalahan seperti ini harus bisa diatasi bahkan dihindari karena aplikasi digunakan oleh user yang banyak dan dengan intensitas yang cukup tinggi. Apabila tidak diatasi maka permasalahan akan berakibat pada menurunnya kualitas aplikasi yang dapat memberikan kerugian materiil yang cukup besar [4] dan dapat mengakibatkan keterlambatan untuk proses bisnis lainnya.

Beberapa penelitian dengan kemiripan karakteristik masalah telah dilakukan oleh Aisyah et. al. [5] dengan memanfaatkan bahasa pemrograman PHP dalam proses loading database dan akhirnya digunakan untuk pembuatan *Data Warehouse*. Program yang dikembangkan membantu perusahaan dalam menyelesaikan permasalahan jumlah database perusahaan besar yang berada pada aplikasi perusahaan. Penelitian yang dilakukan oleh Andriansyah [6] dan Anggrainy [7] sama-sama melakukan proses pembuatan *Data Warehouse* pada industri menggunakan tools bernama *Pentaho Data Integration*. Jurnal penelitian yang dilakukan oleh Putra & Adhitya Putra [8] membahas tentang pengembangan program mereka dengan bahasa *Python* sebagai proses dalam pembangunan *Data Warehouse* sebuah perusahaan penjualan. Penelitian lainnya juga telah dilakukan oleh Pratama [9] dengan melakukan analisis pola data penjualan menggunakan TALEND untuk membuat *Data Warehouse*.

Perbedaan yang dilakukan pada pengembangan program pada penelitian ini dibandingkan dengan penelitian yang telah ditinjau sebelumnya adalah pengembangan pada penelitian ini akan berfokus pada pembuatan program atau tools khusus untuk proses loading dan konsolidasi database dan tidak berfokus untuk pembuatan *Data Warehouse*. Pengembangan program ini akan dikembangkan menggunakan bahasa pemrograman *Python* agar dapat menggunakan berbagai *library open source*-nya terutama *Pandas Library* untuk mencoba meniru proses yang dilakukan menggunakan *Pentaho Data Integration*. Program yang akan dikembangkan juga akan dapat diakses melalui *API Endpoint* yang akan dibuat nantinya agar mempermudah proses pengembangan selanjutnya. Program yang akan dikembangkan pada penelitian ini juga akan dapat berjalan secara otomatis sesuai kebutuhan aplikasi menggunakan *Windows Task Scheduler*. Program yang dikembangkan akan memiliki fitur untuk melakukan proses loading menuju database NoSQL untuk kebutuhan konsolidasi. Fitur ini akan diuji pada akhir penelitian untuk melihat apakah pengembangan database aplikasi pada penelitian selanjutnya dapat mengimplementasikan database NoSQL atau tidak.

Berdasarkan permasalahan yang telah diidentifikasi maka tim peneliti memerlukan adanya alternatif untuk proses konsolidasi database. Salah satu solusi yang dapat dicoba adalah dengan membuat program khusus yang dapat melakukan proses loading sehingga konsolidasi database ini dapat berjalan konsisten dan efisien. Program yang dikembangkan diharapkan dapat melakukan proses dengan margin kesalahan yang lebih rendah dan dapat dikonfigurasi untuk berjalan secara otomatis, serta menampilkan tampilan untuk proses monitoring dan dikelola pada database relational maupun non-relational. Program yang dikembangkan akan menggunakan library *Pandas*. Tujuan pengembangan program ini dapat melakukan proses konsolidasi dengan lebih efisien dan dapat membantu pengelolaan data dengan lebih baik sehingga dapat meningkatkan proses analisis oleh pihak manajerial serta meningkatkan kualitas aplikasi dan layanannya.

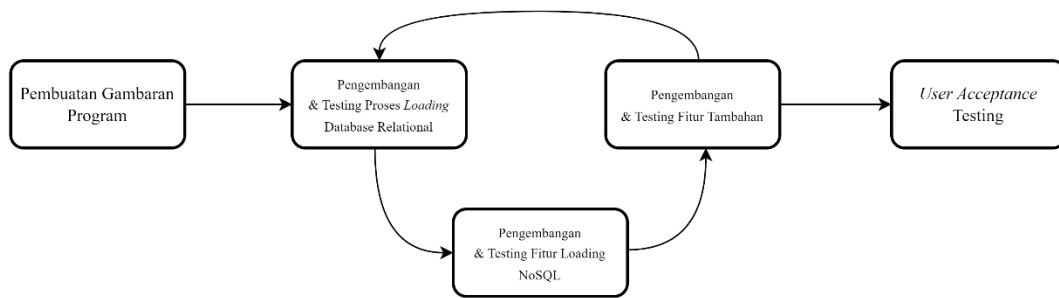
2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Tahapan pengembangan program yang dilakukan akan mengikuti proses Loading serta menerapkan metode *Feature Driven Development* (FDD) sebagai landasan pekerjaan. Pengembangan *Feature Driven Development* akan berfokus pada objek fitur yang diperlukan oleh kebutuhan bisnis user [10]. Tahapan utama dalam FDD secara garis besar adalah seperti berikut [11]:

- Pengembangan gambaran besar program
- Buat daftar fitur yang akan dikembangkan
- Perencanaan sesuai daftar fitur
- Pembuatan desain sesuai daftar fitur, dimana proses ini dilakukan pembuatan alur kerja sistem, rancangan database, dan lain-lain
- Pengembangan sesuai daftar fitur

Tahapan yang dilalui digambarkan pada Gambar 1 mengikuti kelima proses ini sebagai acuan yang akan memiliki alur seperti berikut:



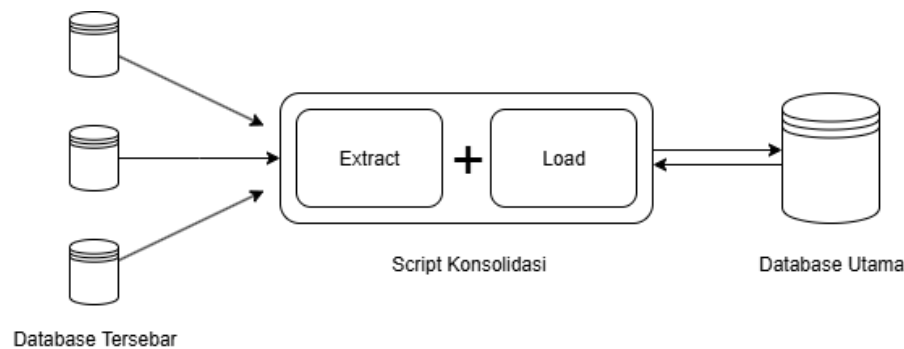
Gambar 1. Tahapan Penelitian

- Pengembangan Gambaran Program**
Pada tahapan ini akan dilakukan pertemuan dengan tim pengembang aplikasi terutama Backend Developer dan Supervisor Divisi untuk menjelaskan kebutuhan apa yang diperlukan untuk sistem yang akan dikembangkan. Pertemuan ini akan menghasilkan sebuah rancangan program akan dibuat untuk mengakomodasi permasalahan proses konsolidasi dan loading database dari tools yang sering mengalami permasalahan.
- Pengembangan & Testing Proses Loading Database Relational**
Tahapan pertama dalam proses pengembangan adalah proses loading. Loading dalam dunia data warehousing merupakan proses dimana data yang telah di kelola dimasukkan menuju data warehouse yang telah dibuat sesuai model yang dibutuhkan [12], dengan kata lain proses loading akan mengambil data yang telah diekstrak dan ditransformasikan menuju database tujuan. Database yang akan digunakan untuk tujuan adalah database MySQL
- Pengembangan & Testing Proses Loading Database NoSQL**
Tahapan selanjutnya adalah proses loading menuju database NoSQL yaitu MongoDB. Database MongoDB adalah salah satu database NoSQL yang menyimpan data dalam bentuk dokumen tanpa skema yang baku [13]. Database ini digunakan karena memiliki kelebihan pada kecepatan komputasinya, kemudahan useran dan keserbagunaannya serta useran BSON yang mana lebih efisien dan lebih mendukung jenis data dibanding JSON [14][15][16]. Proses ini akan mengimplementasikan proses yang sama dengan proses sebelumnya namun mengubah proses nya agar dapat menyesuaikan pada database NoSQL
- Pengembangan & Testing Fitur Tambahan**
Tahapan selanjutnya adalah pembuatan fitur tambahan seperti halaman monitoring, API endpoint, dan otomasi agar terbentuk program yang komprehensif dan memiliki fungsionalitas yang lebih baik
- Testing**
Tahapan terakhir adalah testing yaitu *Performance Testing* dan *User Acceptance Testing*. Testing yang pertama adalah proses tes performa untuk melihat performa program dari sisi kecepatan, stabilitas, dan skalabilitas nya pada proses program itu sendiri dimana pada proses ini akan disimulasikan input user sebestarnya dan digunakan untuk menemukan kemungkinan permasalahan pada sisi resource server [17][18]. API endpoint juga akan menjadi objek yang akan dilakukan tes performa dan hasil tes akan dibandingkan untuk kedua proses database Relational dan NoSQL. Proses testing terakhir adalah testing bersama user (UAT) untuk melihat apakah program telah memenuhi kebutuhan yang telah ditentukan oleh user dan meminta validasi dari pihak user yang terkait. Hasil tes UAT akan menghasilkan keluaran berupa dokumen yang mana dapat menjadi tempat user untuk dapat memberi masukan untuk pengembangan program selanjutnya [19][20]

2.2 Rancangan Proses Program

Program akan melewati proses utama berupa ekstraksi dan loading proses ini menghasilkan sebuah database yang dapat digunakan oleh pihak terkait. Tahapan dari proses ini dapat dilihat pada Gambar 2 dengan penjelasan sebagai berikut:

- Proses Ekstraksi**
Ekstraksi adalah proses awal yang perlu dilakukan. Pada proses ini akan dilakukan ekstraksi data dari database Oracle milik aplikasi e-commerce. Data akan diambil sesuai kebutuhan melalui query yang telah disiapkan dan untuk pengembangan akan digunakan data stok, produk, dan produk_view. Proses selanjutnya adalah transformasi, dimana data yang diekstrak dari database Oracle akan di ubah agar dapat menyesuaikan dengan database tujuan pada kantor pusat. Transformasi untuk proses ini akan memanfaatkan library Pandas dan Python List untuk melakukan menyesuaikan dengan database tujuan.
- Proses Loading**
Tahapan terakhir untuk proses ini adalah proses Loading. Proses loading akan memasukkan data yang telah dimanipulasi agar dimasukkan ke database tujuan. Database tujuan untuk penelitian ini akan menggunakan database lokal MySQL dan MongoDB.

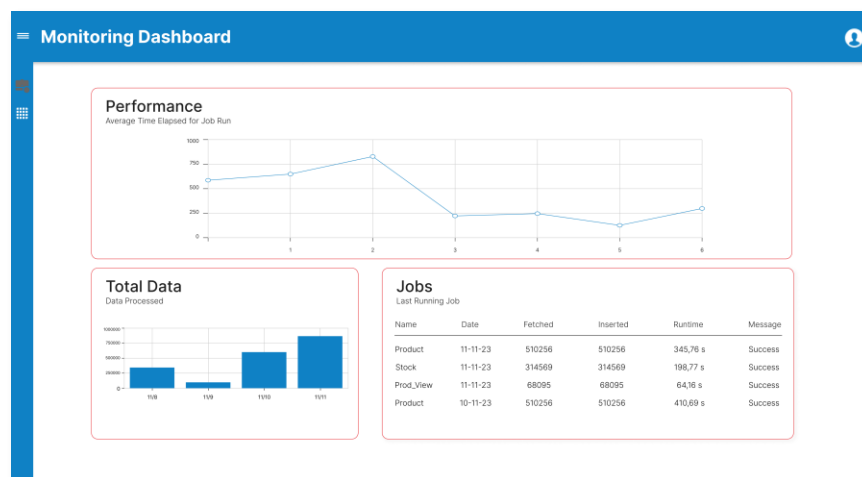


Gambar 2 Rancangan Arsitektur Program

2.3 Rancangan Fitur – Fitur Program

a. Rancangan Halaman Monitoring

Monitoring untuk program akan dibuat menggunakan halaman sederhana yang akan menampilkan performa waktu, data terproses, dan data-data proses yang berjalan. Rancangan halaman terlihat pada Gambar 3.



Gambar 3 Rancangan Halaman Monitoring

b. API Endpoint

API yang akan dikembangkan menggunakan data yang dikumpulkan akan memiliki endpoint GET seperti berikut:

1. `/consolidate/{job_name}` : Proses Konsolidasi & loading untuk data sesuai dengan job_name
2. `/log/{job_name}` : Proses get log data untuk proses terakhir sesuai job_name

c. Rancangan Scenario Testing

Scenario Testing adalah sebuah panduan yang dibuat untuk melihat hasil tes yang akan dilakukan. Pada scenario testing untuk program ini testing akan dilakukan sesuai fitur yang dibuat. Tabel 1 menunjukkan rancangan scenario testing yang akan digunakan:

Tabel 1 Rancangan Scenario untuk Testing

Test Scenario	Result	Notes
Akses ke Dashboard		
Proses Konsolidasi Manual		
Proses Konsolidasi Otomatis		

Proses Konsolidasi via API tabel
product, stocks, product_view

Proses pengambilan data log
konsolidasi product, stock,
product_view melalui API

Proses Konsolidasi data products,
stocks, product view melalui dashboard

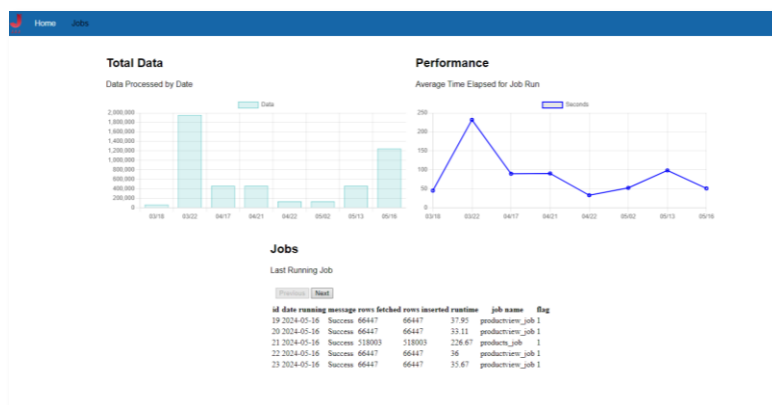
3. HASIL DAN PEMBAHASAN

3.1 Implementasi dan Pengujian

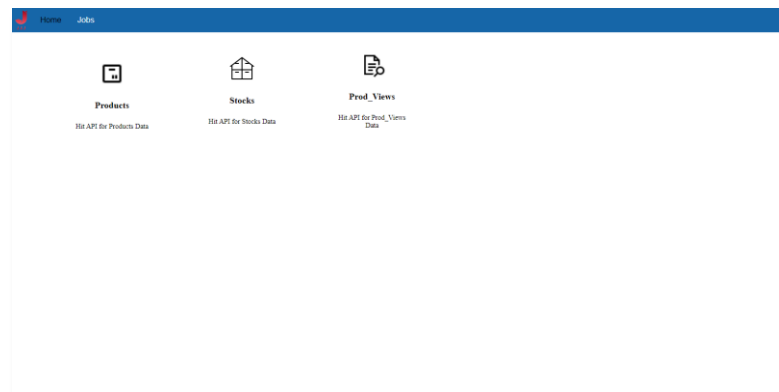
Pada bagian ini akan dijelaskan implementasi fitur – fitur dan implementasi proses konsolidasi yang dilakukan untuk program yang dikembangkan pada penelitian ini. Selain itu pengujian User Acceptanse Testing atau UAT juga dilakukan pada program untuk menemukan validasi dan masukan dari user yang dapat diperbaiki, yang kemudian dapat didokumentasikan. Pengujian performa juga dilakukan untuk menganalisis program terutama pada proses loading pada dua jenis database yaitu MongoDB atau NoSQL dan database Relational MySQL.

3.1.1 Implementasi Fitur – Fitur

Fitur – fitur tambahan di luar proses utama yang dikembangkan pada program ini berfungsi sebagai fitur tambahan untuk memudahkan proses konsolidasi ke depannya maupun membantu proses pengembangan untuk penelitian selanjutnya. Fitur yang pertama dimiliki program ini adalah fitur monitoring berupa laman *web* untuk memantau proses yang sudah berjalan. Selain itu dibuat fitur eksekusi melalui laman web dengan halaman jobs yang berfungsi sebagai halaman untuk menjalankan proses eksekusi program melalui laman eksternal. Gambar 4 menunjukkan implementasi fitur halaman monitoring dan eksekusi program.



Gambar 4 Halaman Monitoring



Gambar 5 Halaman Jobs/Eksekusi

Halaman home dan jobs atau eksekusi ini memiliki fungsi yang sederhana dimana setiap halaman melakukan tujuan sesuai dengan namanya, dimana home atau monitoring dibuat untuk melakukan proses pemantauan untuk data runtime perhari untuk semua jobs yang telah berjalan, jumlah data yang diproses melalui proses konsolidasi program hingga detail data jobs yang telah berjalan. Halaman jobs atau eksekusi sendiri dibuat agar user dapat menjalankan proses eksekusi konsolidasi melalui laman eksternal dengan memanfaatkan API Endpoint nya masing-masing baik *products* maupun *stock*. Melalui kedua fitur dan halaman ini diharapkan user dan pengembang dapat mengakses proses konsolidasi dengan lebih mudah dan optimal.

Fitur selanjutnya yang dikembangkan untuk program ini dan yang telah digunakan pada fitur monitoring dan eksekusi adalah API Endpoints. Fitur ini dibuat untuk memudahkan eksekusi dan monitoring program. API Endpoint akan dikembangkan menggunakan *library* FastAPI yang akan membantu pembuatan API agar dengan lebih mudah dan efisien. Gambar 6 menunjukkan implementasi API Endpoint untuk program yang dikembangkan.



Gambar 6 API Endpoint Program

API Endpoint yang dibuat sesuai dengan rancangan yang dijelaskan sebelumnya yang mana endpoint */consolidate* yang disediakan akan melakukan proses konsolidasi dan endpoint */log* akan melakukan pengambilan data dengan metode GET. Kedua API Endpoint yang dibuat memiliki query khusus yang dapat dimasukkan agar endpoint dapat melakukan proses sesuai dengan jobs nya, query ini adalah *products*, *productviews*, dan *stock* untuk data sesuai namanya. Sebagai contoh useran endpoint */log/products* maka endpoint akan mengembalikan data log proses *products* yang berjalan, dan endpoint */consolidate/products* akan menjalankan proses konsolidasi untuk tabel data produk. Dengan demikian API endpoint ini diharapkan mampu membantu proses secara keseluruhan agar lebih fleksibel baik pada useran maupun untuk pengembangan selanjutnya.

3.1.2 Implementasi *Pandas* untuk Proses Loading Konsolidasi

Proses loading untuk proses konsolidasi akan memiliki dua tahapan yang akan dilalui, yaitu tahap ekstraksi dan tahap loading itu sendiri. Tahapan ekstraksi akan melakukan proses ekstraksi sesuai database sumbernya yang mana pada penelitian ini database sumber adalah database *Oracle*. Proses loading dilakukan akan dilakukan pada dua jenis database yaitu database MySQL dan NoSQL yaitu MongoDB. Program yang dikembangkan akan berjalan pada *backend* program dan dapat dilihat proses nya secara langsung melalui terminal *windows* yang dapat dijalankan secara langsung. Proses konsolidasi dapat dilakukan secara otomatis dengan mengubah program menjadi file *.bat* dan kemudian dijalankan menggunakan *Windows Task Scheduler*. *Pandas*

library pada proses ini digunakan untuk melakukan ekstraksi dan pengolahan data nya agar proses dapat berjalan dengan lebih baik dan efisien. Berikut implementasi proses ekstraksi dan loading pada program dapat dilihat pada Gambar 7 dan Gambar 8.

```
Dataframe with 1447 rows is extracted
Time elapsed for the extraction : 31.99 s
Total Data extracted : 66,447 rows
```

Gambar 7 Proses Ekstraksi

```
productview is now emptied
Time elapsed for the loading : 6.33 s
Total data after inserted : 66,447 rows
Total time for code run: 60.29 s
```

Gambar 8 Proses Loading

Gambar 7 dan Gambar 8 menunjukkan tampilan terminal yang menunjukkan jalannya proses ekstraksi dan loading, dimana proses loading juga memanfaatkan metode dari *pandas* untuk proses loading pada database MySQL. Sedangkan pada database NoSQL memanfaatkan metode milik library *pymongo*.

3.1.3 Pengujian

Pengujian UAT (*User Acceptance Testing*) dan *Performance Testing* (Pengujian Performa) dilakukan untuk program hasil penelitian yang telah dibuat. Hasil dari pengujian User Acceptance Testing yang dilakukan bersama pihak user memberikan hasil pada Tabel 2.

Tabel 2 Hasil Scenario untuk Testing

Test Scenario	Result	Notes
Akses ke Dashboard	Success	Informasi grafis dapat ditambahkan dan dikembangkan, serta informasi grafis dapat dibuat realtime.
Proses Konsolidasi Manual	Success	
Proses Konsolidasi Otomatis	Success	Proses otomatis dapat dikembangkan untuk OS Linux
Proses Konsolidasi via API tabel product, stocks, product_view	Success	
Proses pengambilan data log konsolidasi product, stock, product_view melalui API	Success	
Proses Konsolidasi data products, stocks, product view melalui dashboard	Success	

Hasil pengujian UAT program ini menunjukkan bahwa program yang dikembangkan telah sesuai dengan kebutuhan fitur dari user yang akan menggunakan program ini. Dokumen hasil UAT juga menunjukkan bahwa pihak user memberikan masukan untuk dapat meningkatkan halaman monitoring agar dapat memberikan grafik

yang lebih baik dan lebih lengkap serta realtime serta membuat program dapat dijalankan otomatis untuk *OS Linux* untuk pengembangan dan penelitian selanjutnya.

Pengujian selanjutnya yang dilakukan terhadap program konsolidasi ini adalah *Performance Testing* atau dapat disebut juga pengujian performa yang berfokus pada pengujian waktu saat *run* program yang dilakukan. Pengujian ini dilakukan dengan cara program dijalankan sepuluh kali untuk setiap job yang ada dan kemudian diambil nilai rata-rata data waktu *runtime* program. Data waktu *runtime* program kemudian dibandingkan dengan satu sama lain. Perbandingan pertama yang dilakukan dan dianalisis adalah pada proses loading pada dua jenis database yang berbeda yaitu MySQL dan MongoDB dan perbandingan kedua yang dilakukan adalah antar kedua jenis database dengan proses kontrol pada perusahaan objek penelitian yang pada saat penelitian menggunakan tools *Pentaho Data Integration*. Tabel 3 dan Tabel 4 menunjukkan hasil pengujian performa yang dilakukan.

Tabel 3 Hasil Uji coba pertama antara dua Database

Percobaan	Jobs		
	Product	Stocks	Product_views
<i>Relational</i>	255,70	150,87	35,18
<i>Non-Relational</i>	210,71	132,69	30,86
<i>Peningkatan</i>	9,6%	6,4%	6,5%

Tabel 4 Tabel Hasil uji coba kedua Perbandingan tiga proses

Percobaan	Jobs Product
<i>Kontrol (c) (Pentaho)</i>	300
<i>Program (p) (MySQL)</i>	255,70
<i>Saran (s) (MongoDB)</i>	210,71
<i>Peningkatan Control (c) – P (p)</i>	8,0%
<i>Peningkatan Control (c) - Saran (s)</i>	17,5%

3.2 Pembahasan

Pengembangan program yang dibuat pada penelitian ini berfokus pada pembuatan program untuk melakukan proses Loading. Dibandingkan dengan penelitian sebelumnya program dibuat khusus dengan kebutuhan dimana user telah memiliki bentuk data yang diinginkan dan tidak berfokus program untuk pembuatan *Data Warehouse* sehingga proses loading yang menjadi fokus utama untuk program. Pengembangan program khusus ini melakukan proses ekstraksi sesuai dengan query yang diperlukan oleh pihak user yang dikemudian waktu digunakan untuk kebutuhan pihak user dan berbeda dengan penelitian yang dilakukan oleh Putra dan Adhitya P. [8] dimana peneliti membuat program khusus untuk proses ETL dan digunakan untuk *Data Warehouse*. Penelitian ini juga berbeda dengan penelitian oleh Aisyah [5] yang membuat program khusus menggunakan bahasa pemrograman PHP. Program yang dibuat pada penelitian ini berfokus khusus untuk melakukan proses loading sehingga waktu *runtime* program menjadi hal esensial yang perlu diperhatikan dan diteliti.

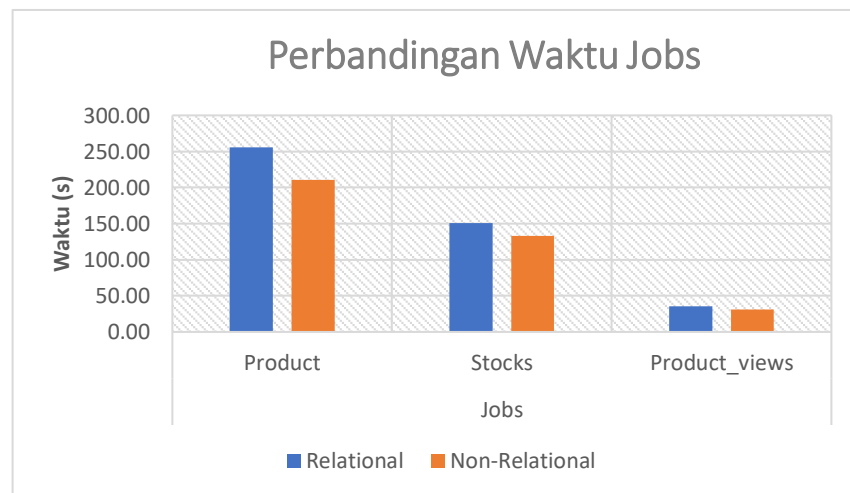
Program yang dibuat memanfaatkan library *pandas* dalam pengkodean, dimana proses ekstraksi dan loading menggunakan metode yang disediakan *pandas*. Proses ekstraksi memanfaatkan metode '*read_sql()*' dari *pandas* untuk proses ekstraksi akan dilakukan menggunakan query yang telah disediakan kemudian menyimpannya dalam bentuk *dataframe* milik *pandas* yang nantinya akan digunakan untuk proses loading. Data yang telah

diekstrak kemudian dimasukkan ke database tujuan menggunakan metode *'to_sql()'*. Semua proses dijalankan setiap *chunk* agar mengefisienkan *resource device* yang digunakan. *Chunk* adalah cara untuk memotong data agar lebih kecil dan hemat *resource*.

Proses ekstraksi secara garis besar mengawali proses dengan membuat koneksi menuju database sumber yang kemudian akan melakukan proses ekstraksi setiap 25.000 data menggunakan perulangan dan menyimpan data dalam sebuah *dataframe pandas*. Loading yang dilakukan selanjutnya mengambil *dataframe* ini untuk proses loading setiap 25.000 data yang dilakukan melalui sebuah perulangan yang dilakukan hingga data selesai di masukkan ke database tujuan. Semua proses yang berjalan akan otomatis dihentikan dan dilakukan *rollback* ketika pada satu proses terdeteksi mengalami kesalahan. Setelah *rollback* dilakukan dan benar terjadi kesalahan, maka keseluruhan proses akan diulangi sesuai jumlah waktu yang ditentukan oleh user. Kedua mekanisme ini dikembangkan untuk memastikan integritas data tetap terjaga dan proses dapat dilanjutkan maupun dihentikan secara aman tanpa kehilangan data yang secara tidak langsung dapat menyebabkan inkonsistensi data.

Perbedaan untuk proses loading menuju database MongoDB dari database MySQL yang terletak pada library yang digunakan. Proses pada MongoDB menggunakan library *pymongo* untuk proses loading dengan metode *insert_many()*. Koneksi database langsung dibuat dengan menggunakan *pymongo.MongoClient*. Sebaliknya pada proses MySQL untuk proses pembuatan koneksi menuju database digunakan *SQLAlchemy* dalam proses pembuatan koneksi. Proses pada kedua database ini mengikuti rancangan program yang dibuat pada Gambar 2, dimana seluruh proses akan melalui proses ekstraksi memanfaatkan library milik *pandas* untuk penyimpanan data sementara dalam bentuk *dataframe*, kemudian melakukan metode selanjutnya melalui proses loading. Implementasi *pandas* pada proses loading ini menunjukkan bahwa pengembangan program memanfaatkan metode – metode *pandas* dapat digunakan untuk pembuatan program khusus untuk proses konsolidasi database.

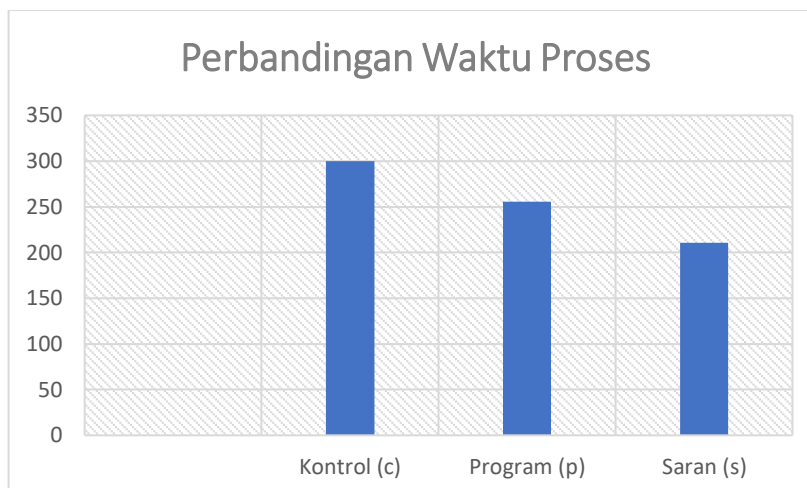
Pengujian performa yang dilakukan pada program menunjukkan bahwa ada peningkatan untuk proses yang dilakukan pada MySQL dan MongoDB. Hasil pengujian dapat dilihat dalam diagram batang Gambar 9.



Gambar 9 Perbedaan pada dua jenis database

Berdasarkan hasil pengujian yang dapat terlihat pada Gambar 9 dapat disimpulkan bahwa proses konsolidasi memanfaatkan proses loading dengan menggunakan database non-relational atau NoSQL cenderung menjalankan proses dengan lebih cepat dan efisien. Secara spesifik proses loading dengan database NoSQL menunjukkan peningkatan dengan rentang 3 – 20 detik lebih cepat atau peningkatan sekitar 6 – 10% dibandingkan dengan database relational MySQL yang dapat dilihat pada Tabel 3.

Pengujian selanjutnya dilakukan untuk mengetahui apakah pemakaian program yang dikembangkan menghasilkan proses yang lebih efisien atau tidak. Dengan menggunakan data proses sebelumnya pada Tabel 3 dan Tabel 4, kemudian mengambil rata-rata proses sekarang yang dilakukan di perusahaan menggunakan tools *pentaho* sebagai kontrol. yaitu 300 detik untuk 500.000 data maka didapatkan bagan yang dapat dilihat pada Gambar 10.



Gambar 10 Perbandingan Kontrol

Gambar 10 menunjukkan bahwa adanya peningkatan yang positif, dimana program yang dikembangkan menunjukkan peningkatan sekitar 45 detik atau sekitar 8% dari program kontrol atau *Pentaho* untuk job product, yang memiliki 500.000 data kemudian program yang menggunakan MongoDB memberikan peningkatan hingga 17.5% atau lebih spesifiknya sekitar dipercepat kurang lebih 60 detik hingga 90 detik dalam prosesnya.

Hasil pengujian ini juga menunjukkan semakin besar ukuran data yang dikonsolidasi menuju database non-relational maka waktu proses nya akan lebih cepat dibanding database relational. Proses peningkatan ini sesuai dengan pernyataan dari jurnal sebelumnya [13][14][15] yang menunjukkan bahwa proses pada database MongoDB atau database NoSQL memberikan proses komputasi yang lebih cepat rata-rata lebih dari 75% dengan komputasi *multi – core*, walaupun secara persentase penelitian ini belum meningkat sebanyak 3 penelitian sebelumnya karena disebutkan bahwa semakin banyak data maka semakin terlihat pula peningkatan kecepatan prosesnya. Peningkatan berdasarkan hasil yang didapatkan dapat memberikan pengaruh yang baik, dimana dengan waktu proses yang lebih cepat terutama pada aplikasi *e-commerce* yang setiap hari memproses ribuan data. Proses yang lebih cepat dapat memberikan kemampuan untuk proses operasional yang lebih responsif, efektif, dan adaptif, serta menurunkan useran sumber daya yang pada akhirnya dapat mengurangi biaya operasional. Dengan demikian berdasarkan dari hasil pengujian dan pengembangan ini dapat dilihat bahwa useran database MongoDB atau *NoSQL* secara garis besar dapat menjadi pilihan yang lebih baik untuk perusahaan yang memiliki jumlah data yang besar dan perlu diolah setiap harinya. Alternatif pengolahan data ini dapat memberikan keuntungan kompetitif .

4. KESIMPULAN

Hasil dari pengembangan dan penelitian ini mampu memberikan alternatif untuk proses loading dan konsolidasi database saat tools *Pentaho* mengalami *down* atau *maintenance*. Pengembangan ini juga menunjukkan bahwa library *Pandas* yang sering digunakan untuk pengolahan data juga dapat diimplementasikan untuk melakukan proses backend sebuah program dengan metode-metode yang tersedia. Program yang dihasilkan juga menunjukkan bahwa program mampu melakukan proses dengan efektivitas yang baik dan dapat memberikan proses bisnis sebuah alternatif apabila terjadi hal-hal di luar dugaan dalam proses konsolidasi. Kinerja pada database NoSQL juga menunjukkan proses yang lebih efektif dan lebih cepat dibanding proses yang dilakukan pada database relational dengan peningkatan yang cukup signifikan. Rata-rata proses loading program hasil pengembangan yang menggunakan database relational menunjukkan peningkatan sekitar 8% atau 20 - 30 detik lebih cepat untuk data berjumlah sekitar 500.000. Pengembangan proses loading menggunakan database NoSQL menunjukkan adanya peningkatan sekitar 6,5 - 9,6% untuk jumlah data yang berkisar dari 20 - 500.000 data. Proses ini juga menunjukkan peningkatan sekitar 17.5% dari program yang digunakan perusahaan sebelumnya untuk proses 500.00 data. Proses pengembangan dari program ini menunjukkan bahwa proses konsolidasi database dapat dibuat tidak hanya menggunakan database relational tetapi juga dapat menggunakan database non-relational. Konsolidasi menuju database non-relational dapat menjadi salah satu fitur yang dapat dikembangkan ke depannya agar proses konsolidasi dapat dilakukan dengan lebih efektif. Pengembangan pada penelitian ini masih terbatas karena pengembangan dilakukan pada *device* lokal dan diuji menggunakan *environment local*. Tahapan selanjutnya adalah pengembangan program dapat dilanjutkan dengan implementasi pada server *backend* dari perusahaan yang menjadi objek penelitian. Program ini juga dapat ditambahkan library bahkan teknologi dari *python* yang lain untuk mencapai tingkat efisiensi yang lebih baik lagi.

REFERENCES

- [1] M. Elena, "Top! Transaksi E-commerce Capai Rp453,75 Triliun Sepanjang 2023," *Bisnis.com*, 2023. <https://finansial.bisnis.com/read/20240118/90/1733241/top-transaksi-e-commerce-capai-rp45375-triliun-sepanjang-2023>.
- [2] V. B. Ramu, "Optimizing Database Performance: Strategies for Efficient Query Execution and Resource Utilization," *Int. J. Comput. Trends Technol.*, vol. 71, no. 7, pp. 15–21, 2023, doi: 10.14445/22312803/ijctt-v71i7p103.
- [3] E. SEUN, G. BABAJIDE, F. TAYE, A. ADERONKE, and B. OLABODE, "Impact of Information Systems on Operational Efficiency: A Comprehensive Analysis," *Indian J. Comput. Sci. Eng.*, vol. 14, no. 4, pp. 661–673, 2023, doi: 10.21817/indjce/2023/v14i4/231404013.
- [4] H. Krasner, "Cost of Poor Software Quality in the U.S.: A 2022 Report," *CISQ*. Bloomsbury Publishing Plc, p. 35, 2022, doi: 10.5040/9781501365287.638.
- [5] Y. Aisyah, S. Anwar, and - Samidi, "Pembuatan Data Warehouse secara Berjenjang dari Data Transaksi dengan ETL Script PHP," *Techno.Com*, vol. 22, no. 3, pp. 609–621, 2023, doi: 10.33633/tc.v22i3.8084.
- [6] D. Andriansyah, "Implementasi Extract-Transform-Load (ETL) Data Warehouse Laporan Harian Pool," *J. Tek. Inform.*, vol. 8, no. 2, pp. 45–49, 2022, doi: 10.51998/jti.v8i2.486.
- [7] T. D. Anggrainy and A. R. Sari, "Implementation of Extract, Transform, Load on Data Warehouse and Business Intelligence Using Pentaho and Tableau to Analyze Sales Performance of Olist Store," *Int. Res. J. Adv. Eng. Sci.*, vol. 7, no. 2, pp. 368–374, 2022.
- [8] I. M. S. Putra and D. K. T. Adhitya Putra, "Rancang Bangun Engine ETL Data Warehouse dengan Menggunakan Bahasa Python," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 3, no. 2, pp. 113–123, 2019, doi: 10.29207/resti.v3i2.872.
- [9] I. P. A. E. Pratama and I. Komang Wahyu Hadi Permana, "Analisa Pola Penjualan Berbasis Data Warehouse Menggunakan Metode ETL dan OLAP (Studi Kasus: Restoran Khayangan Kuliner)," *Tematik*, vol. 10, no. 1, pp. 124–130, 2023, doi: 10.38204/tematik.v10i1.1317.
- [10] A. R. Chrismanto, A. Wibowo, L. Chrisantyo, and M. N. A. Rini, "Implementasi Feature Driven Development untuk Mempermudah Kualitas Fitur dan Adaptasi pada Pengembangan Portal Dutatani Web dan Mobile," *JEPIN (Jurnal Edukasi ...)*, vol. 8, no. 1, pp. 62–73, 2022, [Online]. Available: <https://jurnal.untan.ac.id/index.php/jepin/article/view/50715%0Ahttps://jurnal.untan.ac.id/index.php/jepin/article/viewFile/50715/75676592891>.
- [11] I. T. Kusnadi, J. M. Huddin, A. Supiandi, and R. Oktapiani, "Implementasi Feature Driven Development Pada Sistem Informasi Absensi Dan Penggajian (Sisenji) Berbasis Web," *J. Responsif Ris. Sains dan Inform.*, vol. 6, no. 2, pp. 195–204, 2024, doi: 10.51977/jti.v6i2.1632.
- [12] A. Dhaouadi, K. Bousselmi, M. M. Gammoudi, S. Monnet, and S. Hammoudi, "Data Warehousing Process Modeling from Classical Approaches to New Trends: Main Features and Comparisons," *Data*, vol. 7, no. 8, p. 113, 2022, doi: 10.3390/data7080113.
- [13] N. Zaniewicz and A. Salamończyk, "Comparison of MongoDB, Neo4j and ArangoDB databases using the developed data generator for NoSQL databases," *Stud. Informatica. Syst. Inf. Technol.*, vol. 26, no. 1, pp. 61–72, 2022, doi: 10.34739/si.2022.26.04.
- [14] H. A. Mumtahana, "Optimization of Transaction Database Design with MySQL and MongoDB," *Sinkron*, vol. 7, no. 3, pp. 883–890, 2022, doi: 10.33395/sinkron.v7i3.11528.
- [15] C. M. Sowandi, I. Leonita, S. Hartanto, P. Arisaputra, and D. David, "Exploring the Effectiveness, Features, and Compatibility of MongoDB and MySQL: A Comprehensive Comparison of NoSQL and Relational Databases," *J. MIND J. | ISSN*, vol. 8, no. 2, pp. 217–229, 2023, [Online]. Available: <https://doi.org/10.26760/mindjournal.v8i2.217-229>.
- [16] R. Andreoli, T. Cucinotta, and D. Pedreschi, "RT-MongoDB: A NoSQL Database with Differentiated Performance," *Proceedings of the 11th International Conference on Cloud Computing and Services Science*. SCITEPRESS - Science and Technology Publications, pp. 77–86, 2021, doi: 10.5220/0010452400770086.
- [17] F. A. Mufarroha, A. F. Haq, A. Maghfiroh, D. R. Anamisa, A. A. Supianto, and A. Jauhari, "Quality Assurance of Academic Websites using Performance Testing Tools," *Tech. Rom. J. Appl. Sci. Technol.*, vol. 16, pp. 226–233, 2023, doi: 10.47577/technium.v16i.9985.
- [18] D. Rasch, "Pressman, R. S.: Software Engineering. Grundkurs für Praktiker. McGraw-Hill Software, Engineering. Hrsg.: Dipl.-Inform. John-Harry Wieken. McGraw-Hill Book Company GmbH, Hamburg 1989. 302 S., DM 63, 55, ISBN 3-89028-163-X," *Biometrical J.*, vol. 33, no. 3, p. 378, 1991, doi: 10.1002/bimj.4710330333.
- [19] D. H. A. Gani, I. K. A. Kadir, A. A. Rahman, and A. M. Yunus, "Electronic Document Management System in Electronic Government Environment," *Proc. 9th Int. Conf. Mark. Retail. (INCOMaR 2023), March 1-2, 2023, Kota Kinabalu, Sabah, Malaysia*, vol. 133, p. Dayangku Horiah Awang-597, 2024, doi: 10.15405/epsbs.2024.05.48.
- [20] Y. N. Dalimonthe, A. D. Kalifia, and S. Diwandari, "Pemanfaatan API (Application Programming Interface) Untuk Pengembangan Sistem Pelayanan Dinas Kependudukan dan Pencatatan Sipil," *TEKINKOM*, vol. 6, no. 2, pp. 760–772, 2023, doi: 10.37600/tekinkom.v6i2.1053.