RANCANG BANGUN APLIKASI *EXTRACT, TRANSFORM* DAN *LOAD* UNTUK *DATA WAREHOUSE* BERBASIS *WEB*

I Kadek Sastrawan[1], I Made Arsa Suyadnya[2], Made Sudarma[3]
[123]Jurusan Teknik Elektro Fakultas Teknik Universitas Udayana
Email: kdk.sastrawan@gmail.com[1], mdearsa@yahoo.com[2], sudarma@ee.unud.ac.id[3]

Abstrak

Pesatnya perkembangan teknologi informasi memudahkan terjadi pertukaran data sehingga data yang tersimpan pada sistem operasional menjadi besar dan mengakibatkan menurunnya kinerja server. Data warehouse merupakan solusi dari masalah dalam menampung semua data summary dari sistem informasi operasional. Data warehouse sangat didukung oleh aplikasi dalam proses extract, transform, dan load data yang sering disingkat menjadi ETL. Oleh karena itu dalam penelitian ini diteliti mengenai perancangan aplikasi ETL berbasis web yang dapat mencatat log mulai dari proses extract, transform, sampai pada proses load sehingga dapat mempermudah operator untuk mengetahui data yang gagal diproses. DBMS yang digunakan antara lain MySQL, SQLServer dan ORACLE.

Kata kunci: *data warehouse, extract, transform, load*

1. PENDAHULUAN

Pesatnya perkembangan teknologi informasi sangat mempengaruhi kebutuhan informasi bagi *user*. Oleh karena itu muncul inovasi dalam pembuatan sistem informasi penunjang kegiatan operasional. Seiring banyaknya transaksi yang terjadi pada sistem maka akan menimbulkan permasalahan pada proses *query* sehingga untuk mendapatkan informasi dari *server* menjadi lambat. Berdasarkan permasalahan tersebut maka muncul teknologi *data warehouse*.

Data warehouse ini didukung oleh aplikasi *ETL(Extraction, Transformation, Loading)* yang digunakan untuk menggambarkan suatu proses pengolahan data mulai dari mengumpulkan, menyaring, mengolah dan menggabungkan data yang relevan dari berbagai sumber. *Extract* adalah proses dimana data diambil atau diekstrak dari berbagai sistem operasional. *Transform* adalah proses dimana data mentah (*raw data*) hasil ekstraksi disaring dan diubah. *Load* adalah proses pemuatan data yang didapatkan dari hasil *transform* ke dalam *data warehouse*.

Permasalahan yang sering timbul pada aplikasi ETL adalah fleksibilitas aplikasi terhadap jenis DBMS (*Data Base Management System*) yang digunakan oleh sistem terkait, dimana hanya dapat menghubungkan antara satu jenis DBMS sumber ke satu jenis DBMS tujuan. Hal tersebut menyebabkan setiap sistem akan memerlukan pengembangan aplikasi ETL khusus.

Pengembangan aplikasi berbasis *web* dapat memudahkan pengguna (*user*) mengakses informasi hanya dengan menggunakan *browser*. DBMS yang digunakan meliputi *Oracle, MySQL, MsSQLServer.* diharapkan mampu menghasilkan Aplikasi ETL yang dinamis. Dinamis mempunyai arti proses dalam *ETL* dalam *data warehouse* tersebut bisa disesuaikan oleh *user* yang menggunakan aplikasi ini tanpa mengubah kode program dari aplikasi utamanya. Mengingat fungsi ETL yang begitu penting dan aplikasi berbasis *web* dirasa perlu untuk akses yang lebih luas, maka pada penelitian ini dilakukan pembuatan Aplikasi *Extract, Transform* dan *Load* untuk Data *Warehouse* Berbasis *Web* serta perancangan *database* sebagai penampung data konfigurasi untuk membantu proses ETL.

2. KAJIAN PUSTAKA

2.1 Tinjauan Mutakhir

Penelitian terkait aplikasi *ETL* telah beberapa kali dilakukan sebelumnya.

Armadiyah (2010) melakukan penelitian mengenai analisis faktor-faktor yang mempengaruhi proses *ETL* pada *data warehouse*. Menurut Armadiyah, *ETL (Extract Transform Loading)* pada proses

develop *data warehouse* merupakan suatu proses yang memakan waktu paling lama. Kesuksesan proses *ETL* sangat dipengaruhi oleh kualitas data yang ada pada *database OLTP*. Penelitian ini bertujuan untuk mencari *noise* yang mungkin timbul pada proses ETL pada pengembangan *data warehouse*. Dari analisa yang dilakukan ditemukan bahwa *noise* banyak disebabkan karena adanya data yang bernilai *null* Sehingga sebelum proses *ETL* dilakukan perlu adanya proses menghilangkan noise yang ada pada *database sumber* atau *database OLTP* [1].

Penelitian lainnya dilakukan oleh Mahendra (2011), yang berpendapat bahwa *ETL* merupakan pondasi utama dari *data warehouse*, maka dalam penelitian ini akan diteliti mengenai perancangan dan implementasi aplikasi *ETL*. Aplikasi *ETL* dibuat mampu melakukan proses *cleaning* data yang berfungsi untuk menjamin kualitas data yang akan ditransfer ke dalam *data warehouse*. Untuk mengantisipasi terjadinya kesalahan pada saat proses *ETL* berlangsung maka diperlukan pembentukan data *log* dan *metadata* untuk membantu dalam pemetaan dari sistem sumber menuju sistem tujuan. Untuk menjaga kualitas data agar memperoleh data yang valid maka diperlukan fitur *data cleaning*. Konfigurasi yang dilakukan pada proses *ETL* sangat menentukan proses *ETL* yang akan terjadi dalam sistem. Pengembangan aplikasi *ETL* tidak hanya sebatas aplikasi *desktop* namun dapat berbasis *web* untuk mempermudah pendistribusian [2].

Penelitian lainnya juga dilakukan oleh Febriani (2014), Aplikasi *OLAP* yang dibangun dalam penelitiannya diharapkan dapat membantu mengatasi penumpukan data tanaman hortikultura agar dapat diolah dan dianalisis sehingga membantu pengguna dalam memperoleh informasi ringkasan tanaman hortikultura dengan lebih cepat, yang sangat ditekankan adalah dalam proses pembuatan fungsi *ETL (Extract Transform Load)* untuk mengolah data dan membedakan hasil untuk data yang bernilai 0 dan data yang tidak tersedia [3].

Dari beberapa penelitian yang pernah dilakukan tersebut, maka akan dibuat suatu aplikasi *ETL* yang dinamis, berbasis web dan multi *DBMS*. Aplikasi *ETL* yang dinamis berarti *user* dapat merubah pengaturan pemetaan tabel dan *field* antara *database* sumber dengan tujuan tanpa harus mengubah kode aplikasi, sehingga *user* merasa lebih praktis dalam penggunaannya. Berbasis *web*, dengan teknologi ini user dapat mengakses aplikasi dari mana saja dengan syarat adanya koneksi internet. Aplikasi ini dapat memproses 3 jenis DBMS yaitu MySQL, Oracle dan MsSQLServer, dipilihnya 3 jenis DBMS ini dikarenakan paling banyak digunakan pada sistem tepatnya sistem operasional. Aplikasi ETL yang akan dibangun dibatasi pada yaitu MySQL, Oracle dan MsSQLServer.

2.2 Tinjauan Pustaka

2.2.1 Konsep Data Warehouse

Menurut Inmon (2005), *Data Warehouse* adalah koleksi data yang mempunyai sifat berorientasi subjek, terintegrasi, *time-variant*, dan bersifat tetap dari koleksi data dalam mendukung proses pengambilan keputusan manajemen. Inmon menegaskan bahwa *Data Warehouse* sebaiknya dibangun jika desain arsitektur *Data Warehouse* sudah dibuat (*top-down approach*. *Data Warehouse* merupakan bagian dari *business intelegent* sehingga segala informasi berasal dari satu *Data Warehouse* [4].

2.2.2 Karakteristik *Data Warehouse*

Menurut Turban (2005) karakteristik *data warehouse* dapat dibagi menjadi empat jenis yaitu berorientasi subyek, terintegrasi, rentang waktu dan non volatile. *Data Warehouse* berorientasi subyek artinya *Data Warehouse* didesain untuk menganalisa data berdasarkan subyek-subyek tertentu dalam organisasi, bukan pada proses atau fungsi aplikasi tertentu. *Data warehouse* dapat menyimpan data-data yang berasal dari sumber-sumber yang terpisah kedalam suatu format yang konsisten dan saling terintegrasi satu dengan lainnya. Seluruh data pada *data warehouse* dapat dikatakan akurat atau valid pada rentang waktu tertentu. Melihat interval waktu yang digunakan dalam mengukur keakuratan suatu *data warehouse.* Karakteristik keempat dari data warehouse adalah *non-volatile*, maksudnya data pada *data warehouse* tidak di-*update* secara *real time* tetapi di *refresh* dari sistem operasional secara reguler [5].

2.2.3 Arsitektur *Data Warehouse*

Menurut Kimball (2004), Arsitektur *data warehouse* secara umum terdiri dari beberapa komponen penting diantaranya adalah *Operational Source, Staging Area, Data Warehouse, Data Mart* dan *Metadata.*

Komponen *operational sources* merupakan komponen yang berfungsi sebagai sumber dari data yang diolah ke dalam *data warehouse*. Komponen *staging area* merupakan komponen yang digunakan sebagai tempat proses *ETL*. Komponen *data marts* ini merupakan sebuah laporan yang dihasilkan dari proses *query* yang dilakukan pada *data warehouse* [6].

2.2.4 ETL dalam *Data Warehouse*

Menurut Kimball (2004), ETL (*extraction, transformation, loading*) merupakan aplikasi yang terpisah dari *data warehouse* dan berfungsi sebagai pondasi dari *data warehouse* itu sendiri. ETL terdiri dari tiga proses utama. Ketiga proses ini dilakukan secara berurutan [6].

3. METODOLOGI PENELITIAN

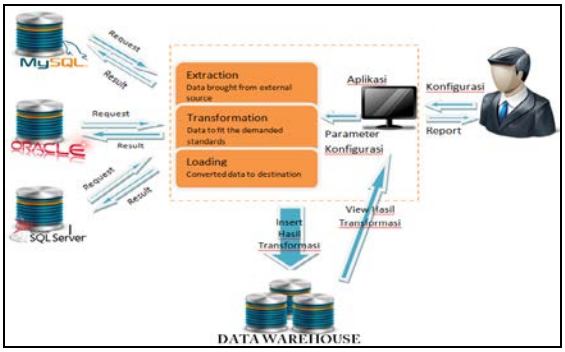
3.1 Tahapan Penelitian

Tahapan Penelitian yang dilakukan adalah sebagai berikut :

1. Pendefinisian masalah dari sistem.
2. Pengumpulan data dan studi literatur terkait pembuatan Rancang Bangun Aplikasi *Extract, Transform* dan *Load Data Warehouse* Berbasis *Web* .
3. Mempelajari proses sistem yang akan dibuat sehingga dapat dilakukan pemodelan sistem terkait aplikasi yang dibangun.
4. Perancangan *database* yang digunakan untuk menampung data yang diperlukan dalam proses ETL.
5. Pengembangan aplikasi perancangan perangkat lunak yang digunakan sebagai *user interface*. Aplikasi yang dikembangkan menggunakan bahasa pemrograman PHP, HTML, CSS dan *Javascript*.DBMS yang digunakan yaitu MySQL, Oracle dan MsSQLServer.
6. Penambahan contoh data yang digunakan pada proses *extract, proses transform* sampai proses *load* .
7. Pengujian sistem ETL dan analisis pengujian sistem ETL.
8. Pengambilan kesimpulan.

3.2 Gambaran Umum Sistem

Gambaran umum sistem dari aplikasi ETL ini dapat digambarkan seperti pada Gambar 1.



Gambar 1 Gambaran Umum Sistem

1. Oracle, MySQL dan SQL Server
   Oracle, MySQL, dan SQL Server merupakan DBMS dari *operational database source* (ODS) dari data yang diekstrak ke dalam *database* tujuan.
2. *Extraction*
   Proses saat ekstraksi, aplikasi mengambil paramater koneksi yang telah dikonfigurasikan oleh *admin* di awal jika berhasil melakukan koneksi ke sistem sumber, maka *admin* akan memilih tabel mana yang diekstrasi.
3. *Transformation*
   Proses ini admin konfigurasi yang dilakukan seperti menentukan jenis transformasi.
4. *Loading*
   Proses pada saat memasukkan data hasil transformasi ke dalam *database* tujuan.
5. *Data Warehouse*
   Data Warehouse ini merupakan tujuan dari *output* yang dihasilkan dalam proses transformasi.

3.3 Himpunan Entitas

Himpunan entitas yang terdapat pada aplikasi *extract, transform* dan *load data warehouse* berbasis *web* ini adalah sebagai berikut:

1. Tb_user(id_user, username, passwd, email)
2. Tb_log(id_log, id_user, action, datetime, ip)
3. Tb_session_restore(id_user, id_profile, datetime, offset, total, status)
4. Tb_profile(id_profile, src_dbms, src_dsn, src_db, src_user, src_passwd, src_port, des_dbms, des_dsn, des_db, des_user, des_passwd, des_port)
5. Tb_tabel_mapping(tm_id, tm_profile, tm_src_tabel, tm_des_tabel)
6. Tb_field_mapping(fm_id, fm_tm_id, fm_src_field, fm_src_type, fm_src_size, fm_des_field,

fm_des_type, fm_des_size, fm_des_format)
7. Tb_log_extract(ext_id, ext_session, ext_map_field, ext_offset, ext_error, ext_status)
8. Tb_log_transform(trf_id, trf_session, trf_map_field, trf_offset, trf_error, trf_status)
9. Tb_log_load(load_id, load_session, load_map_field, load_offset, load_error, load_status)

3.4 Metode Pengujian

Pada penelitian ini aplikasi akan diuji dengan metode *black box* yang akan berfokus pada persyaratan fungsional perangkat lunak. Pengujian *black box* pada aplikasi berusaha menemukan :

1. Fungsi yang tidak berjalan dengan benar atau hilang
2. Kesalahan pada *interface*
3. Kesalahan pada struktur data
4. Kesalahan kinerja sistem
5. Inisialisasi dan kesalahan terminasi.
6. Kesalahan performansi sistem
7. Kesalahan inisialisasi dan tujuan akhir

Dengan mengaplikasikan teknik *black box*, maka akan ditarik serangkaian *test case* dengan cara mengurangi, dengan nilai lebih dari satu, jumlah *test case* tambahan yang harus didesain untuk mencapai pengujian yang dapat dipertanggungjawabkan.

4. HASIL DAN PEMBAHASAN

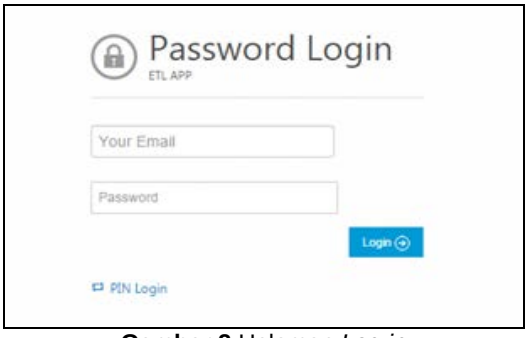
4.1 Hasil

Aplikasi *extract, transform,* dan *load* untuk *data warehouse* berbasis *web* ini merupakan aplikasi *ETL* yang berfungsi sebagai penunjang utama pada *data warehouse*, aplikasi akan mempermudah user untuk melakukan pemetaan antara *database* sumber ke *database* tujuan. Setiap proses *extract, transform* dan *load* yang dilakukan akan disimpan oleh sistem dalam sebuah *log* untuk mempermudah user untuk melakukan *trace* data yang dipetakan.

4.2 Pembahasan Sistem

4.2.1 Tampilan Awal Aplikasi

Tampilan utama dari aplikasi ini adalah halaman *login*. Jika sudah terdaftar pada aplikasi ini, dapat melakukan *login* dengan mengisi form *login* seperti email dan *password* yang didaftarkan. Jika belum terdaftar dapat melakukan pendaftaran melalui admin. Halaman *login* digambarkan pada Gambar 2.



Gambar 2 Halaman *Log in*

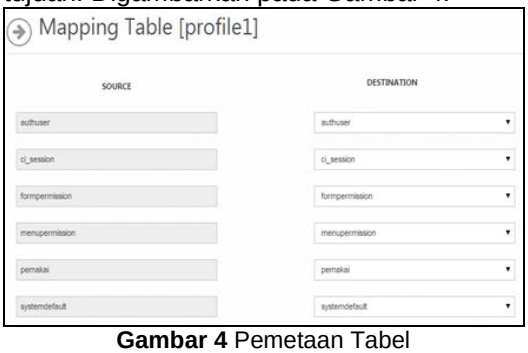
4.2.2 Pembuatan Profil Koneksi

Dalam form profil koneksi ini ada beberapa *field* data yang harus dilengkapi antara lain nama profil, pada bagian sumber terdapat *host* sumber, *driver database* sumber, nama *database* sumber, *user database* sumber, *password database* sumber dan *port database* sumber, dan pada bagian tujuan terdapat *host* tujuan, *driver database* tujuan, nama *database* tujuan, *user database* tujuan, *password database* tujuan dan *port database* tujuan. Halaman profil koneksi digambarkan pada Gambar 3.

Gambar 3 Profil Koneksi

4.2.3 Pemetaan Tabel

Tahapan selanjutnya adalah pemetaan tabel. Setelah profil koneksi berhasil dibuat maka dilanjutkan dengan pemetaan tabel dari tabel asal ke tabel tujuan. Digambarkan pada Gambar 4.

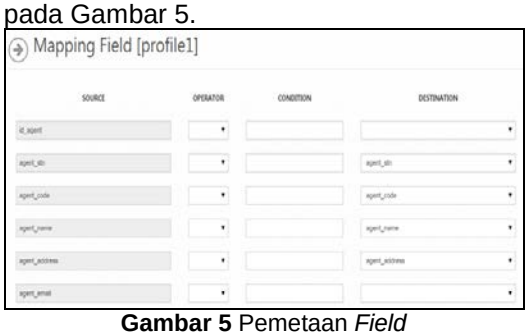


Gambar 4 Pemetaan Tabel

4.2.4 Pemetaan *Field*

Pemetaan *field* dilakukan dengan menentukan *field* yang akan dipetakan dari tabel sumber ke tabel tujuan dan dapat

disesuaikan dengan beberapa kondisi data yang diinginkan oleh user. Digambarkan pada Gambar 5.



Gambar 5 Pemetaan *Field*

4.2.5 Tahap Extract

Proses *Extract* dilakukan dengan data yang berasal dari tabel sumber yang sesuai dengan kondisi yang dinginkan *user* pada pemetaan *field*. Digambarkan pada Gambar 6.

Gambar 6 Proses *Extract*

Gambar 7 menampilkan kode program proses *extract*.

```
function process_extract()
{
$restore_id = $this->etl_model->
save_data("etl_session_restore",$datarestore);
$datarestore["restore_id"] = $restore_id;
$this->session-> set_userdata("restore",$datarestore);
$result_mpt = $this->etl_model->
get_mapped_table_by_id_mpt($mpt_id);
$pk = "" ;
foreach($result_mpt as $rmpt){
$data["src_table"] = $rmpt->mpt_src_table;
$data['src_field'] = $this->etl_model->
show_field_table_src($data["src_table"]);
foreach($data['src_field'] as $sf){
if($sf->primary_key == 1){ $pk = $sf-> name;}
                                    }
                            }
$data["result"] = $this->etl_model->
process_extract($pro_id,$mpt_id,$offset);
$data["result_field"] = $this->etl_model->
get_mapped_field_list_active($mpt_id);
foreach($data["result"] as $row){
    $datalog = array(
            "ext_mpt_id" => $mpt_id,
            "ext_session" => $restore_id,
            "ext_offset" => $row->$pk,
    );
$this->etl_model-> save_log_extract($datalog);
}
```

Gambar 7 Kode Program Proses *Extract*

4.2.6 Tahap Transform

Pada tahap *transform* akan dilakukan penyesuaian format data hasil *extract* dari DBMS sumber dengan format data pada DBMS tujuan. Digambarkan pada Gambar 8.

Gambar 8 Proses *Transform*

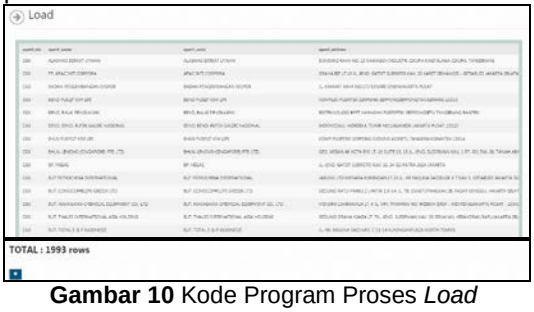
Gambar 9 menampilkan kode program proses *transform*.

```
function process_transform()
{
$result_mpt = $this->etl_model->
get_mapped_table_by_id_mpt($mpt_id);
$pk = "" ;
foreach($result_mpt as $rmpt){
$data["src_table"] = $rmpt->mpt_src_table;
$data['src_field'] = $this->etl_model->
show_field_table_src($data["src_table"]);
  foreach($data['src_field'] as $sf){
    if($sf->primary_key == 1){ $pk = $sf-> name;}
  }
$data["des_table"]= $rmpt->mpt_des_table;
$data['des_field'] = $this->etl_model->
show_field_table_des($data["des_table"]);
   $i = 0;
$restore=$this->session-> userdata("restore");
$data["result"]=$this->etl_model->
process_extract($pro_id,$mpt_id,$restore["restore_off
set"]);
$data["result_field"] = $this-> etl_model->
get_mapped_field_list_active($mpt_id);
$dateformat = "%Y-%m-%d";
foreach($data["result"] as $row){
   for($j=1;$j<=$i;$j++){
      $bantu = $data["vardate_$i"]
   }
  $datalog = array(
    "trf_mpt_id" => $mpt_id,
    "trf_session" => $restore["restore_id"],
    "trf_offset" => $row->$pk,
);
$this->etl_model-> save_log_transform($datalog); }
}
```

Gambar 9 Kode Program Proses *Transform*

4.2.7 Tahap Load

Pada tahapan *Load,* data hasil *transform* pada tahapan sebelumnya akan dimuat ke *database* tujuan. Digambarkan pada Gambar 10.



Gambar 10 Kode Program Proses *Load*

Gambar 11 menampilkan kode program proses *load*.

```
function process_load()
{
$result_mpt=$this->etl_model->
get_mapped_table_by_id_mpt($mpt_id);
foreach($result_mpt as $rmpt){
  $data["des_table"]=$rmpt-> mpt_des_table;
 }
$data["result_field"] = $this->etl_model->
get_mapped_field_list_active($mpt_id);
for($i=1;$i<=$iter;$i++){
  foreach($data["result_field"] as $rf){
  $varbantu = "var_".$rf->
mpf_src_field."_$i";
  $vardes = $rf->mpf_des_field;
  $datainsert["$vardes"] = $this->input->
post($varbantu);
  }
 $des_id = $this->etl_model->
insert_des($data["des_table"],$datainsert);
if($des_id > 0){$error = 0;}else{$error = 1;}
$datalog = array(
  "load_mpt_id" => $mpt_id,
  "load_session" => $restore["restore_id"],
  "load_offset" => $this->input->post("pk_$i"),
  "load_offset_des" => $des_id,
  "load_error" => $error,
);
  $this->etl_model->  save_log_load($datalog);

}
$restore["restore_total"] = $i;
$where_restore = array("restore_id" =>
$restore["restore_id"]);
$this->db->
update("etl_session_restore",$restore,$where_restore)
;
}
```

Gambar 11 Kode Program Proses *Load*

4.3 Pengujian Aplikasi

Pengujian aplikasi menggunakan metode *blackbox* dimana akan membandingkan antara harapan yang dihasilkan dengan hasil dari uji coba pada setiap proses. Uji coba dilakukan dari *login* ke aplikasi, pemetaan tabel, pemetaan *field*, proses *extract*, proses *transform*, dan proses *load* dengan menggunakan data yang berbeda. Dari hasil pengujian yang telah dilakukan *interface*, struktur data, dan setiap fungsi dari aplikasi sudah berjalan sesuai dengan yang telah dirancang.

5. SIMPULAN DAN SARAN

5.1 Simpulan

Simpulan adalah sebagai berikut.

1. Aplikasi berbasis *web* ini dibuat dengan menggunakan bahasa HTML, PHP, CSS, dan *Javascript*. Alur penggunaan aplikasi ini dirancang mulai dari proses pembuatan profil koneksi, pemetaan tabel, pemetaan *field*, proses *extract, transform,* dan *load*. Semua proses *ETL* disimpan dalam sebuah *log data*.
2. Aplikasi ini dapat menangani proses *ETL* untuk 3 jenis DBMS yaitu MySQL, SQLServer, dan Oracle sehingga membuat aplikasi lebih efektif dalam penggunaannya.
3. Pembuatan *session restore* mempermudah *user* untuk melanjutkan proses *ETL* jika terjadi kegagalan pada proses sebelumnya.
4. Pada aplikasi ini, pemetaan tabel dan *field* dilakukan pada awal proses sehingga proses ETL menjadi lebih efisien dalam penggunaan selanjutnya.

5.2 Saran

Saran adalah sebagai berikut.

1. Melengkapi aplikasi ini dengan penambahan jenis *DBMS* yang dapat digunakan sehingga dapat membuat *user* merasa lebih praktis dalam penggunaannya.
2. Perbaikan fitur *mapping* dengan menggunakan *pointing* sehingga aplikasi dapat terlihat lebih interaktif.

DAFTAR PUSTAKA

[1] Armadiyah A, *Analisis Faktor-Faktor yang Mempengaruhi Proses ETL pada Data Warehouse.* Yogyakarta : Amikom;2010

[2] Mahendra, *Rancang Bangun Aplikasi ETL Untuk Data Warehouse Berbasis Oracle*. Badung :Udayana;2011

[3] Febriani D, Imas S. *Data Warehouse dan OLAP Berbasis Web untuk Tanaman Hortikultura Menggunakan Palo* , Makalah Kolokium. 2014

[4] Inmon W H. *Building the Data Warehouse Fourth Edition*. Wiley Canada : Publishing Inc. 2005

[5] Kimball, Ralph, Caserta*. The Data Warehouse ETL Toolkit Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data*. Canada : Wiley Publishing. Inc. 2004

[6] Turban E, dkk. *Decision Support Systems and Intelligent Systems*. Yogyakarta : Andi Offset; 2005.

.