Pengembangan *Engine* Integrasi Tabel HTML pada Halaman *Web*

Memen Akbar[1], Fazat Nur Azizah[2], G. A. Putri Saptawati[3]

***Abstract*— Two problems are arisen while integrating number of tables from number of web pages, i.e. structural conflict and semantic conflict. To tackle those problems, the proposed study combines some existing methods that are already proven to solve problems in integrating process. The proposed integration process of HTML table consists of 4 phases: (1) locating the table in web pages, (2) separating attributes and data values, (3) integrating the table scheme, (4) migrating the data values into integrated scheme. Table location in web page is determined using heuristic approach. This approach also can separate the attributes and the data values of the table. Semantic conflict that is apparent while integrating the table scheme is handled using domain specific ontology. The resulted data value, then, is migrated to table scheme in line with duplication data checking using vector space model. Result of the integration is presented as single HTML table. This approach is implemented as an engine that is coded using Phyton language. Result of experiment shows that the proposed approach can be used to integrate number of HTML table from number of web pages into a single integrated table.**

***Intisari*— Terdapat dua persoalan dalam mengintegrasikan sejumlah tabel HTML dari beberapa halaman *web*, yaitu: konflik struktural dan konflik semantik. Untuk mengatasi kedua masalah tersebut, makalah ini mengombinasikan beberapa metode yang telah ada dan terbukti menyelesaikan persoalan dalam proses integrasi. Terdapat empat tahapan proses dalam integrasi tabel, yaitu: (1) menentukan lokasi tabel pada halaman *web*, (2) memisahkan bagian atribut dan *data values*, (3) mengintegrasikan skema tabel, dan (4) migrasi *data values* ke skema terintegrasi. Lokasi tabel pada suatu halaman *web* ditentukan dengan pendekatan *heuristic*. Pendekatan ini juga dapat memisahkan atribut dan *data values* dari tabel. Konflik semantik yang muncul saat mengintegrasikan skema tabel diatasi dengan memanfaatkan ontologi untuk suatu domain spesifik. *Data values* hasil integrasi kemudian dimigrasikan ke skema tabel bersamaan dengan pemeriksaan duplikasi data pada tabel dengan menggunakan *vector space model*. Hasil integrasi kemudian ditampilkan dalam sebuah tabel tunggal dalam bentuk HTML. Pendekatan ini diimplementasikan menjadi sebuah *engine* yang dibuat menggunakan Python. Hasil pengujian menunjukkan bahwa pendekatan ini dapat digunakan untuk mengintegrasikan sejumlah tabel dari beberapa halaman *web* menjadi sebuah tabel terintegrasi.**

***Kata Kunci*— integrasi data, tabel HTML, ontologi, integrasi tabel, halaman web**

[1]*Program Studi Magister Informatika Sekolah Teknik Elektro dan Informatika (STEI) Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40135 (memen@pcr.ac.id)*

[2,3]*Sekolah Teknik Elektro dan Informatika (STEI) Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40135 (fazat@informatika.org, putri@informatika.org)*

I. PENDAHULUAN

Tabel masih menjadi bentuk yang paling banyak digunakan untuk menampilkan data relasional termasuk pada halaman *web*. Di dalam tulisannya, [1] menyebutkan bahwa lebih dari 50% halaman *web* memuat tabel untuk menyampaikan data relasional. Seiring dengan pertumbuhan jumlah halaman *web* di internet, terdapat kebutuhan untuk menggabungkan informasi dari sejumlah tabel pada domain tertentu dan menampilkannya ke dalam sebuah tabel tunggal yang dinamakan dengan integrasi tabel.

Kebutuhan untuk mengumpulkan data dari berbagai sumber diperlukan oleh hampir seluruh kegiatan suatu organisasi, termasuk dari tabel HTML [2]. Integrasi sejumlah tabel dari beberapa halaman *web*, dengan memanfaatkan dokumen HTML, mempunyai peranan penting dalam banyak aplikasi potensial. Integrasi tabel dari beberapa *e-commerce* memudahkan seseorang yang berada pada bagian *procurement* untuk membandingkan dan menentukan Harga Perkiraan Sendiri (HPS) barang yang akan diadakan. Integrasi tabel juga memudahkan calon penumpang pesawat terbang untuk melihat informasi penerbangan dari beberapa maskapai dalam satu halaman *web*. Integrasi tabel juga dibutuhkan dalam *web-mining*, *knowledge management*, dan beberapa aplikasi potensial lainnya.

Menurut [3], persoalan integrasi tabel pada halaman *web* dibagi menjadi tiga, yaitu: (1) konflik struktural, (2) konflik semantik, dan (3) konflik penamaan. Konflik struktural terjadi jika terdapat perbedaan struktur tabel yang akan diintegrasikan. Di antara konflik struktural ini, seperti: jumlah dan posisi atribut yang berbeda, serta heterogenitas *layout* tabel. Beberapa masalah yang terkait dengan konflik semantik adalah sinonim, homonim, dan konflik kontekstual. Sedangkan konflik penamaan terkait dengan istilah yang digunakan pada atribut masing-masing tabel.

Penelitian untuk mengatasi persoalan dalam integrasi tabel HTML telah banyak dilakukan. Penelitian tersebut dilakukan oleh beberapa peneliti dan menghasilkan pendekatan yang telah divalidasi untuk mengatasi persoalan dalam integrasi tabel HTML. Namun, solusi yang diberikan hanya untuk persoalan skema data. Belum ada peneliti yang memberikan solusi untuk mengatasi persoalan integrasi pada *data values* dan belum ada juga penelitian yang menghasilkan *engine* untuk integrasi tabel HTML dari beberapa halaman *web*.

Berdasarkan latar belakang permasalahan tersebut, maka pada makalah ini dikembangkan pendekatan untuk mengintegrasikan informasi pada tabel HTML dari sejumlah halaman *web* untuk suatu *domain of interest* (DOI) yang spesifik menjadi sebuah tabel HTML tunggal dengan mengombinasikan beberapa metode. Pendekatan yang

diperoleh kemudian diimplementasikan menjadi sebuah *engine* dengan menggunakan bahasa pemrograman Python.

II. INTEGRASI TABEL

Terkait dengan masalah struktural yang disampaikan oleh [3] pada bagian sebelumnya, khususnya mengenai heterogenitas layout tabel, [4] membagi tabel menjadi lima jenis, yaitu: *column wise*, *row wise*, *column-row wise*, *composite*, dan *mixed-cell table*. *Column wise*, *row wise*, dan *column row wise* terkait dengan orientasi posisi atribut terhadap *data values*. Pada *column wise table*, atribut berada pada baris pertama diikuti dengan *data values* di bawahnya. Sedangkan pada *row wise table* atribut berada pada kolom pertama dan *column row wise table*, atribut berada pada baris dan kolom pertama. Tabel I merupakan contoh tabel dengan *layout column-row wise*.

TABEL I
CONTOH *COLUMN-ROW WISE TABLE*

	2002				**2003**		
	Q1	**Q2**	**Q3**	**Q4**	**Q1**	**...**	**Q4**
dept1	1.30	1.32	1.30	1.35	1.20	...	1.40
dept2	Not available		1.15	1.20	1.10	...	1.25

TABEL II
CONTOH *COMPOSITE TABLE*

Category	**Base Compensation**	**Bonus**	**Total**
Corporate Salaried	$80K	$5K	$85K
	$60K	$0K	$60K
	$95K	$10K	$100K
Category	**Base Compensation**	**Bonus**	**Total**
1099 Contractor	$30/ht	n/a	n/a
	$20/hr	n/a	n/a
	$40/hr	n/a	n/a
Category	**Base Compensation**	**Bonus**	**Total**
W-2 Contractor	$50/hr	n/a	n/a
	$40/hr	n/a	n/a
	$65/hr	n/a	n/a

TABEL III
CONTOH *MIXED-CELL TABLE*

Last: 60.31	**Change:** +0.87	**Open:** 59.95	**High:** 60.65	**Low:** 59.66	**Volume:** 21,914,000
	Percent Change: +1.46%	**Yield:** n/a	**P/E Ratio:** 56.37	**52 Week Range:** 47.50 to 76.15	

Composite table adalah tabel dengan posisi atribut berada pada beberapa baris yang berselang beberapa baris setelahnya. *Data values* berada pada baris setelah tiap baris atribut. *Mixed-cell table* adalah tabel dengan posisi atribut dan data *values* berada pada satu *cell*. Sepasang atribut dan *data values* disusun dalam sebuah *cell*, biasanya dengan menggunakan separator yang identik. Tabel II dan Tabel III merupakan contoh tabel dengan *layout composite* dan *mixed-cell*.

Untuk mengatasi permasalahan dalam integrasi tabel pada halaman *web* tersebut, beberapa peneliti telah menawarkan solusi yang telah divalidasi.

Ada peneliti yang melakukan integrasi data dari sejumlah tabel *web* pada suatu DOI dengan *layout* yang berbeda, khususnya untuk tabel dengan *layout column wise* dan *row wise* [5]. Masalah skema data yang diangkat adalah *multiple heading*. Penelitian ini memberikan tabel standar yang berbentuk *column wise* untuk mengatasi perbedaan *layout* antar tabel. Sedangkan untuk masalah semantik, penelitian ini mendefinisikan himpunan *metadata* atribut yang mewakili suatu DOI dari sejumlah tabel *web* sebagai atribut. Penelitian ini berhasil menangani masalah sinonim antar atribut.

Peneliti lain melakukan integrasi data dari sejumlah tabel *web* yang berbentuk *column* atau *row wise* [6]. Mereka menambahkan kasus jika dalam satu *cell* terdapat beberapa pasangan atribut-*data values*. Untuk mengatasi masalah ini, mereka menggunakan pendekatan teknik *data mining*, *Support Vector Machine* (SVM), dan *Hidden Model Markov* (HMM). Kedua metode ini berguna untuk mengenali bagian mana yang berfungsi sebagai atribut dan bagian mana sebagai *data values*. Namun, pendekatan ini hanya berlaku untuk tabel dengan atribut yang berpasangan hanya dengan satu *data value*.

Peneliti lain, [3], juga melakukan integrasi untuk tabel yang berbentuk *column* dan *row wise*. Masalah struktural diatasi dengan melakukan normalisasi dan pendekatan *heuristic* berbasis *rule* untuk menghilangkan *noisy* pada tabel, seperti *merged cell* dan *no heading*. Sedangkan, konflik semantik diatasi dengan mendefinisikan *Lexical Semantic Set* (LSS). Metode ini digunakan untuk mengatasi masalah sinonim dan himpunan nama atribut yang berhubungan secara kontekstual untuk mengatasi masalah homonim dan konflik kontekstual.

Ada juga peneliti yang melakukan integrasi tabel *web* dengan struktur data yang tidak diketahui, khususnya untuk tabel *column wise* dan *row wise* [7]. Mereka mendefinisikan beberapa operator untuk mengatasi masalah struktural yang terkait dengan perbedaan skema, yaitu *merged attribute*, *attribute as value*, dan atribut *subset*. Terkait dengan masalah semantik mereka menggunakan ontologi yang dapat menemukan atribut-atribut yang sinonim dan homonim.

Peneliti lainnya melakukan integrasi untuk tabel yang terdapat pada *deep web* atau tabel yang diakses melalui sebuah tampilan antarmuka pencarian [1]. Halaman pencarian ini dijadikan sarana untuk menentukan DOI. Struktur tabel yang dibahas pada penelitian ini adalah struktur yang kompleks, yang mana atribut dan *data values* berada pada satu *cell* yang sama. Untuk mengatasi masalah semantik, khususnya yang terkait dengan sinonim dan homonim mereka juga menggunakan ontologi.

III. PERANCANGAN

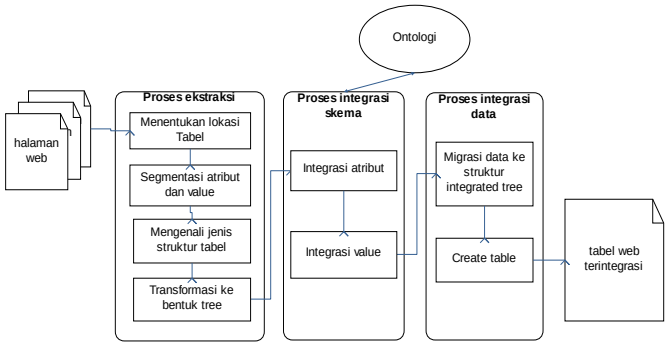
Pada bagian ini disampaikan arsitektur umum proses integrasi tabel yang akan diterapkan pada *engine*. Perancangan *engine* menggunakan paradigma pengembangan prosedural. Pada makalah ini digunakan diagram konteks dan *data flow diagram*.

A. Arsitektur Umum Proses Integrasi Tabel

Gbr. 1 merupakan arsitektur umum proses integrasi tabel HTML. Terdapat tiga proses utama dalam integrasi tabel HTML, yakni: (1) ekstraksi, (2) integrasi skema, dan (3) integrasi data. Masing-masing proses dipecah menjadi beberapa proses yang lebih rinci. Proses-proses inilah yang menjadi acuan untuk mengembangkan *engine* integrasi tabel HTML.

1) Proses Ekstraksi: Ekstraksi informasi dari sebuah halaman *web* dapat dilakukan dengan tiga pendekatan [8], yaitu pendekatan berbasis *rule*, pendekatan statistika, dan pendekatan berbasis *knowledge*. Pada makalah ini, informasi diekstrak menggunakan pendekatan berbasis pengetahuan berdasarkan ontologi.

Pada satu halaman *web*, tidak hanya terdapat tabel. Selain tabel, suatu halaman *web* dapat terdiri atas teks, navigasi, iklan, panel, atau gambar. Langkah ini bertujuan untuk mengambil bagian-bagian yang diperlukan dan mengeluarkan bagian-bagian yang tidak diperlukan oleh *engine*. Bagian yang diperlukan tersebut berupa pasangan atribut dan *data values* yang disusun dalam bentuk struktur *tree*.



Gbr. 1 Arsitektur umum proses integrasi tabel HTML.

Lokasi tabel dapat ditentukan dengan mengenali *tag* `<table>` pada dokumen HTML. Untuk mengenali *tag* `<table>` pada dokumen HTML dapat digunakan pendekatan pengenalan *string*. Keluaran dari proses ini adalah sejumlah bagian pada dokumen HTML yang berada di antara *tag* `<table>` dan `</table>`. Dengan cara yang sama, elemen-elemen lain pada tabel HTML juga dikenali, yaitu *tag* `<tr>`, `<th>`, dan `<td>`.

Pada suatu halaman HTML dimungkinkan terdapat lebih dari satu tabel dengan jenis, ukuran, dan keperluan yang berbeda. Untuk itu, digunakan batasan ukuran tabel seperti yang diterapkan oleh [1] dan [7]. Ukuran minimal sebuah tabel adalah 3 x 3. Jika ukuran tabel kurang dari itu, maka dianggap sebagai bukan tabel yang diinginkan.

Input pada tahap segmentasi atribut dan *data values* adalah tabel berukuran minimal 3 x 3. Pada tahap ini, elemen-elemen tabel diekstrak dan kemudian dipisahkan antara elemen yang berfungsi sebagai atribut dan elemen yang berfungsi sebagai *data values*.

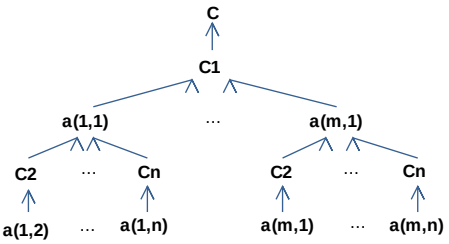
Pada tahap mengenali jenis struktur tabel, *layout* tabel dapat dikenali dengan menelusuri posisi atribut yang telah diperoleh pada tahap sebelumnya. Jika atribut berada pada baris pertama, maka *layout* tabel adalah *column wise*. Jika atribut berada pada kolom pertama, maka *layout* tabel adalah *row wise*. Jika atribut berada pada baris pertama dan kolom pertama, maka *layout* tabel adalah *column-row wise*. Jika atribut terdapat pada beberapa baris, maka *layout* tabel adalah *composite*. Tabel dengan *layout mixed-cell* telah dikenali pada proses sebelumnya dengan menghitung jumlah separator tiap *cell* pada tabel.

Tahap berikutnya adalah transformasi ke bentuk *tree*. Struktur antara yang digunakan pada sistem ini adalah hirarki semantik karena kebutuhan struktur antara ini adalah untuk proses pencocokan atribut antar tabel. Penelusuran pada hirarki semantik langsung dilakukan pada atribut dan *data values* sebuah tabel. Ketika dilakukan penelusuran terhadap suatu atribut, yang pertama kali ditemukan adalah atribut atau *data values*. Berbeda dengan hirarki sintaks, ketika dilakukan penelusuran terhadap suatu atribut, yang pertama kali ditemukan adalah *tag*. Atribut atau *data values* ditemukan pada daun terbawah dari sebuah *tree*.

Namun, sebelum diubah ke bentuk tree, *layout* tabel yang beragam tersebut diubah menjadi bentuk standar. Bentuk standar yang digunakan dinamakan *p-table*, yaitu sebuah tabel dengan *layout column wise* dengan atribut berada pada satu baris pertama, kemudian diikuti dengan *data values* pada baris berikutnya. Gbr. 2 merupakan bentuk umum dari *p-table* dan Gbr. 3 merupakan hirarki semantik dari tabel tersebut.

C		
C_1	…	C_n
$a_{1,1}$	…	$a_{1,n}$
⋮	…	⋮
$a_{m,1}$	…	$a_{m,n}$

Gbr. 2 Tampilan *p-table*.



Gbr. 3 Hirarki semantik dari *p-table*.

2) Proses Integrasi Skema: Input pada proses integrasi atribut ini adalah sejumlah hirarki semantik dari tabel *web* yang direpresentasikan dalam bentuk *tree*. Keluaran yang diharapkan adalah skema global gabungan dari hirarki semantik tabel sumber.

Integrasi atribut dilakukan dengan melihat kemiripan karakter penyusun string atribut antar tabel. Atribut yang secara karakter memiliki kemiripan di atas 50% dianggap atribut yang sama. Kemiripan antar string dihitung berdasarkan *edit distance* menurut (1).

$$labelSim(l_i, A_i) = 1 - \frac{EditDist(l_i, A_i)}{\max(L(l_i), L(A_i))} \quad (1)$$

dengan l_i adalah atribut, A_i adalah atribut lainnya.

Atribut yang secara tekstual sama berkemungkinan merupakan atribut yang homonim. Oleh karena itu perlu diperiksa pada *knowledge* yang menjadi *input* pada tahap ini, yakni ontologi. Jika atribut tersebut terdapat pada dua konsep yang berbeda pada ontologi, maka atribut tersebut homonim dan dianggap dua atribut yang berbeda.

Untuk atribut yang secara tekstual tidak mirip, perlu diperiksa konflik sinonim di antara keduanya. Konflik sinonim diperiksa dengan bantuan ontologi. Dua atribut yang berada pada satu induk konsep dianggap sinonim atau dengan kata lain dianggap sama. Atribut yang dianggap sama tersebut digabungkan menjadi satu atribut pada skema hasil integrasi.

Pada dasarnya, setelah atribut diintegrasikan, *data values* yang bersesuaian mengikuti hasil pemetaan atributnya. Jika atribut A dipetakan ke konsep umum C, maka semua *data values* yang berada di bawah A dipetakan menjadi *data values* dari C. Namun, *data values* yang dipetakan ke sebuah konsep pada ontologi memiliki format berbeda. Oleh karena itu, untuk meningkatkan kualitas data hasil integrasi, sistem harus dapat menetapkan standar format *data values* yang diintegrasikan.

Penelitian ini melakukan pemeriksaan terhadap format data yang berupa tanggal. Penelitian ini menerapkan algoritme pengenalan pola penyusun string berdasarkan urutan karakter huruf dan karakter angka. Format penulisan tanggal yang berbeda tersebut kemudian diubah menjadi format standar. Format standar yang digunakan adalah dd/mm/yyyy.

3) Proses Penggabungan Data: Pada tahap migrasi data ke struktur *tree* terintegrasi, disiapkan struktur *tree* terintegrasi untuk menampung data dari *tree* asal. Struktur *tree* terintegrasi dibangun dari konsep pada ontologi yang digunakan pada integrasi atribut. Untuk kebutuhan pada tahapan selanjutnya, struktur *tree* terintegrasi harus mengikuti format hirarki semantik struktur *tree* pada tahapan ekstraksi. Konsep pada ontologi menjadi *node* pada hirarki semantik *tree* terintegrasi. Konsep ini mengisi *node* pada baris kedua dan keempat *tree*. Sedangkan *node* pada baris ketiga dan kelima diisi oleh *data values* yang bersesuaian.

Untuk menangani duplikasi data, digunakan teknik *vector space model* (VSM) yang mengubah tiap *data values* dalam suatu baris menjadi koordinat sebuah ruang vektor. Kemiripan antar satu baris data dengan yang lainnya dihitung dengan (2). Pemeriksaan duplikasi data hanya dilakukan untuk atribut-atribut yang beririsan antara satu tabel dengan yang lainnya. Oleh karena itu, *engine* harus mendeteksi atribut yang beririsan dari tabel yang akan diintegrasikan.

$$DataRecordSim(DR_i, DR_{i+j}) = \frac{\overline{V_{DR_i}}.\overline{V_{DR_{i+j}}}}{\left|V_{DR_i}\right|\left|V_{DR_{i+j}}\right|} \quad (2)$$

dengan i = 1, 2, 3, ..., (n-1) dan j = (i+1), (i+2), ..., n. $\overline{V_{DR_i}}$ merupakan vektor dari data *record* ke-i, dan $\left|V_{DR_i}\right|$ adalah panjang data *record*.

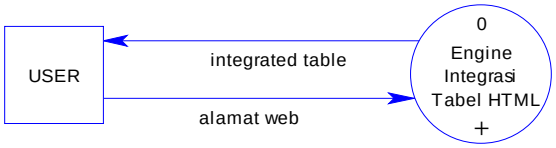
Pada tahap *Create table* ini dilakukan transformasi dari *tree* terintegrasi menjadi struktur HTML. Untuk memetakan *tree* hirarki semantik ke HTML digunakan balikan dari cara yang digunakan pada bagian pertama.

B. Diagram Konteks dan DFD Level 1

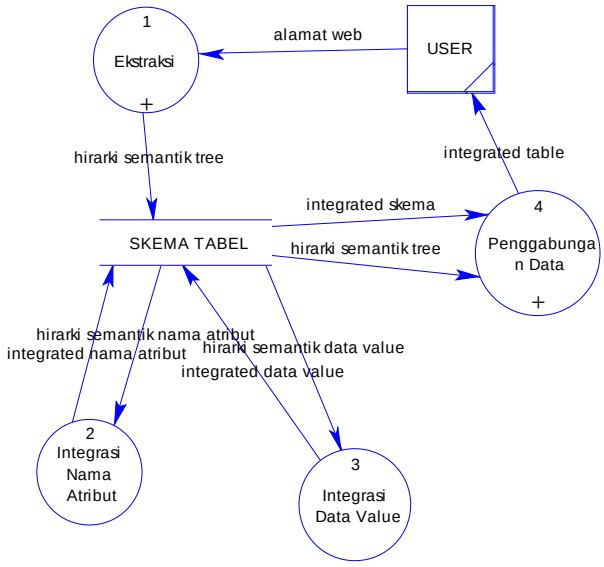
Gbr. 4 merupakan diagram konteks dari *engine* yang dikembangkan. Pengguna (*user*) sebagai entitas eksternal memasukkan sejumlah alamat *web* dan *engine* menampilkan tabel yang telah terintegrasi.

Diagram konteks tersebut kemudian didekomposisi menjadi *Data Flow Diagram* (DFD) level 1 seperti pada Gbr. 5. Terdapat tiga proses yang terjadi pada *engine*, yakni: (1) proses ekstraksi, (2) proses integrasi skema, dan (3) proses penggabungan data.

Input pada proses ekstraksi adalah beberapa alamat *web* yang dimasukkan *user*. Keluaran dari proses ini adalah hirarki semantik dalam bentuk *tree* sejumlah tabel yang berhasil diekstrak. Hirarki semantik memuat dua elemen data dalam tabel, yakni atribut dan *data values*. Hirarki semantik disimpan dalam sebuah tempat penyimpanan data. Sejumlah elemen atribut dan *data values* yang terdapat pada hirarki semantik menjadi *input* untuk proses integrasi skema.



Gbr. 4 Diagram konteks *engine*.

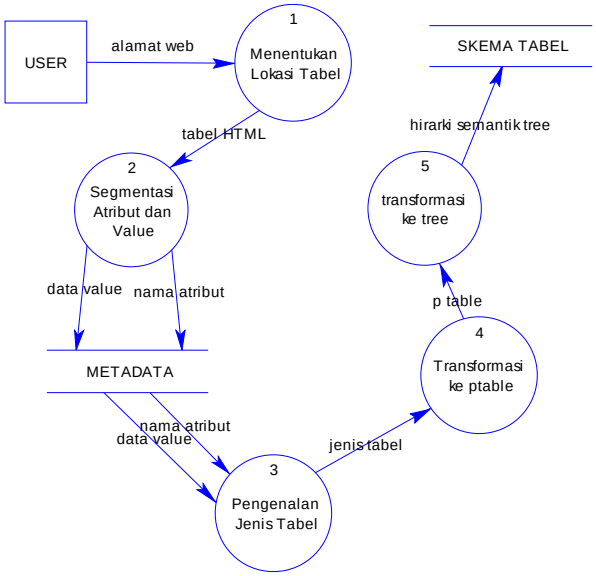


Gbr. 5 DFD level 1.

Input proses integrasi atribut dan integrasi *data values* adalah hirarki semantik *tree* dari penyimpanan data 1. Hirarki semantik *tree* terdiri atas dua komponen, yakni *leaf* yang berperan sebagai atribut dan *leaf* yang berperan sebagai *data values*. Hirarki semantik yang berperan sebagai atribut menjadi *input* untuk proses integrasi atribut. Di sini atribut dari beberapa tabel diintegrasikan sehingga menghasilkan atribut terintegrasi. Hirarki semantik yang berperan sebagai *data values* menjadi *input* untuk proses integrasi *data values*. Di sini *data values* pada atribut yang bersesuaian dari beberapa tabel diintegrasikan sehingga menghasilkan *data values* terintegrasi. Kedua elemen ini kemudian disimpan

dalam tempat penyimpanan 2. Pada kedua proses ini, atribut dan *data values* dari tabel sumber diintegrasikan sehingga menghasilkan sebuah skema global. Skema global ini kemudian disimpan dalam tempat penyimpanan data.

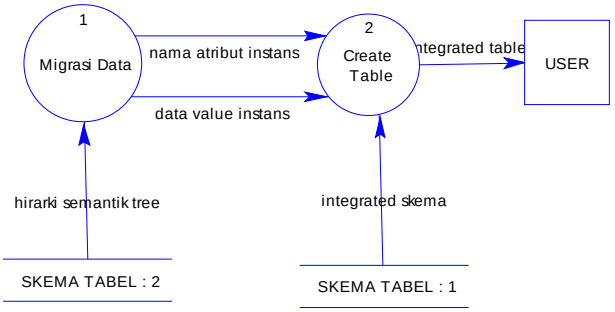
Skema global, bersama dengan atribut dan *data values* pada hirarki semantik, menjadi *input* untuk proses penggabungan data sehingga menghasilkan sebuah skema global lengkap dengan instansnya. Dari proses penggabungan data ini diperoleh skema terintegrasi yang kemudian ditransformasi menjadi bentuk HTML yang dapat dilihat oleh pengguna.



Gbr. 6 DFD level 2 proses ekstraksi.

Gbr. 6 merupakan DFD level 2 untuk proses ekstraksi. Terdapat lima proses yang terjadi pada proses ekstraksi ini, yaitu proses menentukan lokasi tabel, segmentasi atribut dan *data values*, pengenalan jenis tabel, transformasi ke *p-table*, dan proses transformasi ke model *tree*. Pada proses menentukan lokasi tabel, *user* memasukkan alamat *web* yang kemudian diolah untuk mendapatkan lokasi tabel pada halaman *web* tersebut. Bagian pada halaman *web* yang berupa tabel menjadi *input* untuk proses segmentasi atribut dan *data values*. Pada proses ini, elemen pada tabel yang berfungsi sebagai atribut dan *data values* dipisahkan dan kemudian disimpan dalam sebuah tempat penyimpanan data. Kedua elemen dalam sebuah tabel ini, yakni nama atribut dan *data value*s, kemudian digunakan untuk mengenali jenis tabel. Terdapat lima jenis *layout* tabel yang ditangani dalam makalah ini, sebagaimana yang telah dibahas pada bagian sebelumnya. Jenis *layout* tabel dikenali dari posisi atribut terhadap *data values*. Beberapa jenis tabel tersebut kemudian ditransformasi ke *p-table* dan ditransformasi lagi ke *tree*. *Tree* dari masing-masing tabel disimpan dalam sebuah tempat penyimpanan data.

Gbr. 7 merupakan DFD level 2 untuk proses penggabungan data. Terdapat dua proses yang dilakukan, yaitu proses migrasi data dan proses *create table*. *Input* pada proses migrasi data adalah hirarki semantik *tree* pada tempat penyimpanan data. Atribut dan *data values* yang terdapat pada hirarki semantik melewati proses migrasi data, elemen data values diperiksa kualitasnya terutama terkait dengan format data dan duplikasi data. Instans *data values* dan nama atribut dari tabel sumber digabung dengan skema global dari tempat penyimpanan data 2 di dalam proses *create table*. Proses ini mengeluarkan hasil *integrated table* yang ditampilkan kepada *user*.



Gbr. 7 DFD level 2 proses penggabungan data.

IV. HASIL DAN PENGUJIAN

Bagian berikut ini membahas mengenai tahapan pengembangan dan cara mengoperasikan *engine* yang dihasilkan. Pada bagian setelahnya, dijelaskan proses pengujian *engine* dengan beberapa kasus uji.

A. Hasil

Engine dikembangkan pada sistem operasi Windows. *Engine* ini berbasis *desktop*. Koneksi internet dibutuhkan untuk membaca halaman *web* yang dimasukkan pengguna. *Engine* belum mempunyai GUI yang menarik bagi pengguna. *Engine* ini masih dalam *file* Python sehingga untuk mengoperasikannya, pengguna harus telah menginstal Python dengan versi yang sama, yakni Python 2.6. *Engine* ini belum bisa diintegrasikan dengan bahasa pemrograman yang populer saat ini, seperti asp.net dan php.

Input masih dilakukan pada bagian *console* bawaan Python. *Input* dari pengguna harus merupakan URL lengkap di mana terdapat tabel yang akan diintegrasikan. Alamat *web* yang menjadi *input* tidak dibatasi. Pemisah antar alamat *web* menggunakan *semicolon* yang harus diketikkan pengguna pada setiap akhir sebuah alamat *web*. Keluaran tabel hasil integrasi tidak langsung ditampilkan pada layar pengguna, tetapi disimpan menjadi sebuah *file* .html yang berada dalam satu *folder* dengan *engine* ini.

Pada *folder* yang sama juga harus terdapat *file* ontologi untuk suatu domain spesifik tabel yang akan diintegrasikan. Sumber *file* ontologi tidak menjadi pembahasan dalam makalah ini. Namun, dalam *engine* ini, ontologi yang ditangani masih dalam bentuk *file* .txt. *Engine* ini belum dilengkapi dengan fungsi yang dapat mengubah format *file* ontologi yang sudah ada, seperti .owl atau .rdf menjadi bentuk .txt.

B. Pengujian

Tabel pengujian berasal dari beberapa halaman *web*. Terdapat delapan halaman *web* yang menjadi sumber tabel pengujian. Masing-masing halaman *web* memuat satu atau lebih tabel. Ukuran tabel bervariasi dari yang paling kecil 3 x

2 hingga yang paling besar berukuran 101 x 8. Dari delapan halaman *web* tersebut, dua halaman *web* terkait dengan domain *car rental*, empat halaman *web* terkait dengan domain judul tesis mahasiswa, dan dua lainnya terkait peringkat universitas. Atribut dari tabel pada masing-masing domain bervariasi dan terdapat perbedaan pada beberapa bagian. Tabel sumber telah mencakup semua *layout* tabel yang menjadi pembahasan dalam makalah ini, yaitu *column wise*, *row wise*, *column-row wise*, *composite*, dan *mixed-cell table*.

1) Pengujian Heterogenitas Layout Tabel Input: Tujuan pengujian ini adalah mengintegrasikan tabel dengan *layout* yang berbeda. Satu tabel dengan *layout* tertentu diintegrasikan dengan tabel lain dengan *layout* yang berbeda. Tanpa mengurangi tujuan utama dari proses pengujian, tabel dengan beraneka ragam *layout* untuk pengujian ini dibuat dalam sebuah *blog*, yakni memen.komputer.pcr.ac.id. Tabel dari *blog* ini yang kemudian diintegrasikan menggunakan *engine* yang dihasilkan.

Gbr. 8 merupakan tabel dengan *layout column wise* dan Gbr. 9 merupakan tabel dengan *layout row wise*. Domain kedua tabel adalah mengenai tesis mahasiswa. Kedua tabel diintegrasikan dan menghasilkan tabel pada Gbr. 10.

Nama Mahasiswa	NIM	Judul Tesis
Rina Praptini	23512090	Usulan Model Kualitas Website e-Government
Adi Basyudewo	23512041	Sentimen Analisis Twitter
Dini Nurmalasari	23512124	Analisis dan Pemodelan Sequential Pattern sebagai Representasi Data Multimedia
Anggy Trisnadoli	23512116	Pengembangan Playability Quality Model untuk Mobile Game

Gbr. 8 Tampilan tabel pengujian dengan *layout column wise.*

Mahasiswa	Ardianto Wibowo	Danur Wendo	Wiwin Styorini	Kartina Diah
NIM	23512137	23513128	23512001	23512104
Opsi	RPL	CS	Telkom	RPL
Judul Tesis	Realtime Data Loading using SOA	Paralel Programming	Steganografi pada Video Streaming	Text Mining

Gbr. 9 Tampilan tabel pengujian dengan *layout row wise.*

Judul Tesis	Nama Mahasiswa	NIM
Usulan Model Kualitas Website e-Government	Rina Praptini	23512090
Sentimen Analisis Twitter	Adi Basyudewo	23512041
Analisis dan Pemodelan Sequential Pattern sebagai Representasi Data Multimedia	Dini Nurmalasari	23512124
Pengembangan Playability Quality Model untuk Mobile Game	Anggy Trisnadoli	23512116
Realtime Data Loading using SOA	Ardianto Wibowo	23512137
Paralel Programming	Danur Wendo	23513128
Steganografi pada Video Streaming	Wiwin Styorini	23512001
Text Mining	Kartina Diah	23512104

Gbr. 10 Tampilan potongan tabel hasil integrasi.

2) Pengujian Ukuran Tabel Input: Pengujian ini bertujuan untuk mengukur kemampuan engine menangani tabel yang berukuran besar. Pengujian menggunakan tabel yang diambil dari http://webometrics.info pada Gbr. 11. Tabel ini menampilkan daftar peringkat universitas di Indonesia. Dari situs yang sama, juga terdapat tabel yang menampilkan daftar peringkat universitas di Malaysia. Masing-masing tabel berukuran 101x8. Kedua tabel ini diintegrasikan dengan menggunakan engine yang telah dibuat. Hasil integrasi tabel dari kedua halaman web dapat dilihat pada Gbr. 12.

ranking	World Rank	University	Det.	Presence Rank*	Impact Rank*	Openness Rank*	Excellence Rank*
1	763	Universitas Indonesia		357	381	327	1855
2	801	Institute of Technology Bandung		398	409	363	1907
3	807	Universitas Gadjah Mada		368	450	63	2050
4	1448	Universitas Diponegoro		374	840	339	3047
5	1508	Universitas Riau		1439	430	1216	3481
6	1532	Brawijaya University		365	411	248	3960
7	1544	Bogor Agricultural University		379	1250	317	2907
8	1719	Universitas Padjadjaran		704	700	308	3834
9	2072	Universitas Sebelas Maret		386	610	193	4892
10	2157	Universitas Syiah Kuala		616	2866	2324	2881
11	2286	Petra Christian University		727	3190	158	3422
12	2428	Universitas Hasanuddin		656	2550	1082	3654

Gbr. 11 Tampilan tabel dari peringkat univeristas di Indonesia.

University	World Ranking
Universitas Gadjah Mada	781
Institute of Technology Bandung	819
University of Indonesia	909
Airlangga University	1440
Brawijaya University	1517
Diponegoro University / Universitas Diponegoro	1528
Bogor Agricultural University	1554
Institut Teknologi Sepuluh Nopember	1887
Universitas Padjadjaran	1913
Universitas Lampung	2078
Universitas Syiah Kuala	2329
Universitas Pendikan Indonesia / Indonesia University of Education	2342
Petra Christian University	2344
Universitas Sebelas Maret	2367
Universitas Riau	2410

Gbr. 12 Tampilan hasil integrasi tabel peringkat universitas di Indonesia dan Malaysia.

Tabel yang berasal dari halaman *web* pertama akan mengisi baris pertama pada tabel hasil integrasi. Tabel yang berasal dari halaman *web* kedua akan mengisi baris setelahnya.

TABEL IV
PERBANDINGAN DENGAN PENELITIAN LAIN

No	Proses	[5]	[3]	[7]	[4]	[1]	Penelitian ini
1.1	Menentukan lokasi tabel			V		V	
1.2	Segmentasi atribut dan nilai data	V	V		V		Diperbaiki
1.3	Pengenalan jenis *layout*				V		Didefinisikan sendiri
1.4	Transformasi ke bentuk *tree*	V			V		*Row wise*, *composite*, dan *mixed-cell* didefinisikan sendiri
2.1	Integrasi atribut			V		V	
2.2	Integrasi nilai data						Didefinisikan sendiri
3.1	Data migration					V	Didefinisikan sendiri
3.2	Create table						Didefinisikan sendiri

C. Perbandingan dengan Metode Lain

Tabel IV merupakan perbandingan hasil penelitian dengan pendekatan-pendekatan yang sudah ada. Hasil penelitian ini dibandingkan dalam hal metode yang digunakan dalam integrasi tabel, ruang lingkup tabel yang diintegrasikan, dan keluaran dari masing-masing penelitian.

Tanda *check list* (V) menandakan metode yang digunakan dalam penelitian ini. Penelitian ini menggunakan metode yang diterapkan oleh [7] dan [1] untuk menentukan lokasi tabel. Untuk segmentasi atribut dan *data values*, penelitian ini memperbaiki pendekatan yang dilakukan oleh [5], [3], dan [4]. Untuk pengenalan jenis *layout*, penelitian ini mendefinisikan sendiri jenis *layout* yang ditangani. *Layout* yang ditangani pada penelitian ini semua layout yang didefinisikan oleh [4]. Untuk tabel dengan *layout mixed-cell*, *composite*, dan *column row wise*, belum ada penelitian yang menangani proses integrasinya. Selain itu, pada proses integrasi *data values*, migrasi data, dan menampilkan data hasil integrasi dalam tabel HTML lagi, didefinisikan sendiri dalam penelitian ini.

V. KESIMPULAN

Berdasarkan proses analisis dan pengujian yang dilakukan, dapat disimpulkan bahwa algoritme untuk mengintegrasikan sejumlah tabel HTML dari beberapa halaman *web* telah berhasil dikembangkan dengan mengombinasikan beberapa metode yang telah ada. Duplikasi *data values* pada data hasil integrasi dideteksi dengan metode *vector space model*. Untuk meningkatkan kualitas data hasil integrasi, salah satu data yang berulang tersebut dihapus.

Sebuah *engine* sudah berhasil dikembangkan berdasarkan algoritme integrasi tabel HTML yang dihasilkan. *Engine* dapat mengintegrasikan sejumlah tabel HTML dengan beragam *layout* dan ukuran yang berasal dari beberapa halaman *web*. *Engine* berhasil mengubah tabel hasil integrasi menjadi sebuah tabel HTML tunggal tanpa ada duplikasi data di dalamnya.

REFERENSI

[1] Chen Kerui, Zhao Jinchao, Zuo Wanli, He Fengling, and Chen Yongheng, "Automatic table integration by domain-specific ontology," *International Journal of Digital Content Technology and Its Application*, vol. 5, no. 1, pp. 218-226, January 2011.

[2] Eko Prasetyo, Lukito Edi Nugroho, and Marcus Nurtiantara Aji, "Perancangan Data Warehouse Sistem Informasi Eksekutif untuk Data Akademik Program Studi," *JNTETI*, vol. 1, no. 3, pp. 13-20, November 2012.

[3] Shijun Li, Zhiyong Peng, and Mengchi Liu, "Extraction and integration information in HTML tables," in *Fourth International Conference on Computer and Information Technology (CIT)*, 2004.

[4] Yeon-Seok Kim and Kyong-Ho Lee, "Extracting logical structures from HTML tables," *Computer Standards and Interfaces (Elsevier)*, vol. 30, no. 5, pp. 296-308, August 2007.

[5] Seung-Jin Lim, Yiu-Kai Ng, and Xiaochun Yang, "Integrating HTML tables using semantic hierarchies and meta-data sets," in *International Database Engineering and Application Symposium (IDEAS)*, 2002.

[6] Kumi Itai, Atsuhiro Takasu, and Jun Adachi, "Information extraction from HTML pages and its integration," in *Symposium on Application and the Internet Workshops (SAINT-w)*, 2003, pp. 1-6.

[7] David W. Embley, Cui Tao, and Stephen W. Liddle, "Automating the extraction of data from HTML tables with unknown structure," *Data & Knowledge Engineering (Elsevier)*, vol. 54, pp. 3-28, November 2004.

[8] Agny Ismaya, "Algoritma Ekstraksi Informasi Berbasis Aturan," *JNTETI*, vol. 3, no. 4, pp. 242-247, November 2014.