



Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

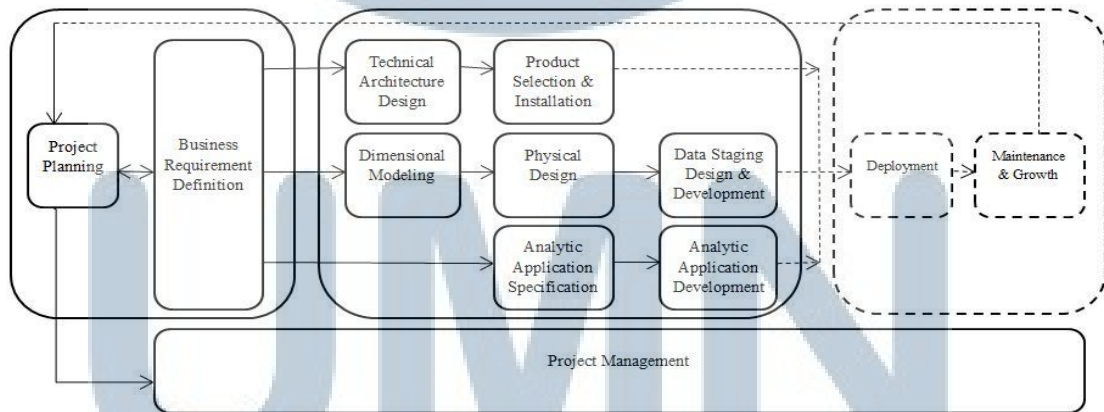
Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

BAB III

METODOLOGI

Kimball's Business Dimensional Lifecycle merupakan salah satu metodologi untuk membuat *data warehouse* yang dikembangkan oleh Ralph Kimball dan koleganya. Pendekatan Kimball ini cocok untuk pengembangan *data warehouse* dengan pendekatan *bottom-up*. Pendekatan *bottom-up* dimulai dengan satu *data mart* (misalnya penjualan), kemudian baru ditambahkan *data mart* lainnya (misalnya pemasaran, penggajian, dll.)



Gambar 3.1 Kimball's Business Dimensional Lifecycle

Penelitian ini bertujuan untuk merancang sebuah aplikasi *business intelligence* untuk bagian pemasaran UMN yang datanya didapat dari sebuah *data warehouse* yang berisi data untuk bagian pemasaran UMN atau dengan kata lain sebuah *data*

mart untuk bagian pemasaran UMN. Dari *data mart* tersebut, kemudian dapat dikembangkan lagi dengan penambahan *data mart* lain sehingga sesuai dengan metodologi yang dibuat Kimball. Dengan menggunakan pendekatan ini, data mengalir langsung dari sumber data ke dalam *data mart*, kemudian baru ke *data warehouse*. Keuntungan dari pendekatan ini adalah dapat melakukan implementasi dengan secara bertahap sehingga implementasi dapat dilakukan dengan lebih cepat.

Penelitian ini melakukan tahap-tahap yang ada di *Kimball's Business Development Lifecycle* kecuali *deployment*, *maintenance*, dan *growth*. Hal itu dilakukan sesuai dengan ruang lingkup dalam penelitian ini.

3.1 Project Planning

Project planning menandakan awal dari sebuah *lifecycle*. *Project planning* dimulai dengan penentuan ruang lingkup dengan memahami *business requirement*. Hal ini dilakukan dengan mewawancarai bagian pemasaran UMN. Kemudian, *project planning* dilanjutkan dengan pemahaman proses bisnis yang terkait dengan *project* yang dikerjakan. Pemahaman proses bisnis ini dilakukan dengan mewawancarai bagian teknologi informasi (IT) UMN.

Selain itu, *project planning* juga mencakup identifikasi pekerjaan yang perlu dilakukan. Kemudian, masing-masing pekerjaan tersebut dibuatkan penjadwalan yang sesuai. Untuk dapat menyelesaikan pekerjaan tersebut, *project planning* juga harus mempersiapkan perencanaan sumber daya yang baik.

3.2 Business Requirements Definition

Keberhasilan dari penelitian ini sangat bergantung pada pemahaman akan *business requirement*. Pemahaman akan faktor-faktor kunci pendorong kegiatan bisnis adalah penting untuk keberhasilan menerjemahkan *business requirement* ke dalam rancangan *data warehouse* dan BI yang akan dibuat.

Business requirements dalam penelitian ini didapat dari hasil wawancara dengan kepala bagian pemasaran UMN yaitu Bapak Iwan Setiawan. Selain itu, untuk dapat lebih memahami kebutuhan bisnis dari bagian pemasaran, wawancara dengan bagian IT UMN, yaitu Sdr. Dwi Kristiawan, juga dilakukan.

Hasil wawancara dengan Bapak Iwan Setiawan dapat dilihat di lampiran. Hasil wawancara tersebut akan dibahas pada bab berikutnya.

3.3 Technology Track

3.3.1 Technical Architecture Design

Technical architecture design harus dibuat berdasarkan masalah bisnis yang ingin dipecahkan. Fokus dari fase ini adalah untuk membuat perencanaan untuk *framework* dan arsitektur aplikasi yang sesuai dengan *business requirements*.

Penelitian yang dilakukan oleh Erick Theodorus dan Bobbi Octavianus Setiawan (2010) menyimpulkan bahwa penyediaan kebutuhan informasi analitik kepada pihak *principal* dapat dilakukan secara otomatis melalui sistem informasi berbasis web. Harland (2011) menambahkan bahwa kebutuhan informasi dan *report* secara *real-*

time dapat disediakan dengan adanya sebuah *business intelligence* (BI) agar dapat memberikan informasi analitik kepada pengambil keputusan.

Kusnawi (2010) menyebutkan bahwa kemampuan BI dalam proses pengambilan keputusan bersumber dari data yang berkualitas, salah satunya adalah *data warehouse*. Dalam penelitiannya, Ditya Octavianto (2006) juga mengatakan bahwa *data warehouse* memungkinkan untuk menghasilkan berbagai bentuk informasi seperti *report* dan data analisis. Ia juga menambahkan bahwa OLAP *report* dapat membantu manajemen menganalisa informasi yang dibutuhkan dilihat dari berbagai dimensi dan melihat informasi secara detail sehingga membantu manajemen membuat keputusan dengan cepat dan tepat.

Berdasarkan *business requirements* dari pihak pemasaran dan hasil penelitian yang telah dilakukan oleh berbagai pihak, penelitian ini berusaha menjawab kebutuhan itu dengan membangun sebuah sumber data yang berisi data analisis dalam bentuk *data warehouse*. Kemudian, untuk menyampaikan informasi tersebut ke pihak pemasaran, dibangun sebuah aplikasi BI untuk pembuatan *report* secara otomatis yang bisa diakses kapan saja oleh pemasaran. *Report* tersebut berupa data analisis yang bisa di-*break down* sesuai dengan kebutuhan pihak pemasaran. Di samping itu, aplikasi BI berupa *interactive dashboard* juga dibangun demi menyediakan informasi yang ringkas dan fleksibel kepada pihak pemasaran. Aplikasi BI tersebut dibuat dengan sistem yang berbasis web.

Arsitektur *data warehouse* yang dirancang pada penelitian ini berupa *data warehouse* terpusat. Adapun alasan pemilihan arsitektur tersebut adalah:

- 1) konsistensi data dapat lebih dijaga sehingga dapat mengurangi masalah redundansi data dan menjamin integritas data;
- 2) integritas data berarti menjamin bahwa informasi yang dihasilkan *reliable*;
- 3) mengakomodasi pengembangan *data warehouse* dengan menyediakan data yang sudah bersih dan siap digunakan.

3.3.2 Product Selection & Installation

Di fase ini, dilakukan pemilihan, instalasi, dan pengujian terhadap *tools* dan produk yang ingin digunakan. Pemilihan tersebut disesuaikan dengan *database environment* yang digunakan oleh UMN pada saat penelitian ini dilakukan.

Hasil penelitian yang dilakukan oleh Antonius Hermawan (2006) berhasil membuktikan bahwa data yang dihasilkan dari proses implementasi *data warehouse* dengan *tools* Oracle Warehouse Builder (OWB) 10g tersebut tetap konsisten. Ia juga menambahkan bahwa OWB merupakan *tools* yang mampu menjaga konsistensi data sekalipun tidak mendukung proses *update* untuk setiap pemetaan. Oleh sebab itu, penelitian ini juga menggunakan OWB untuk perancangan dan pembuatan *data warehouse*-nya.

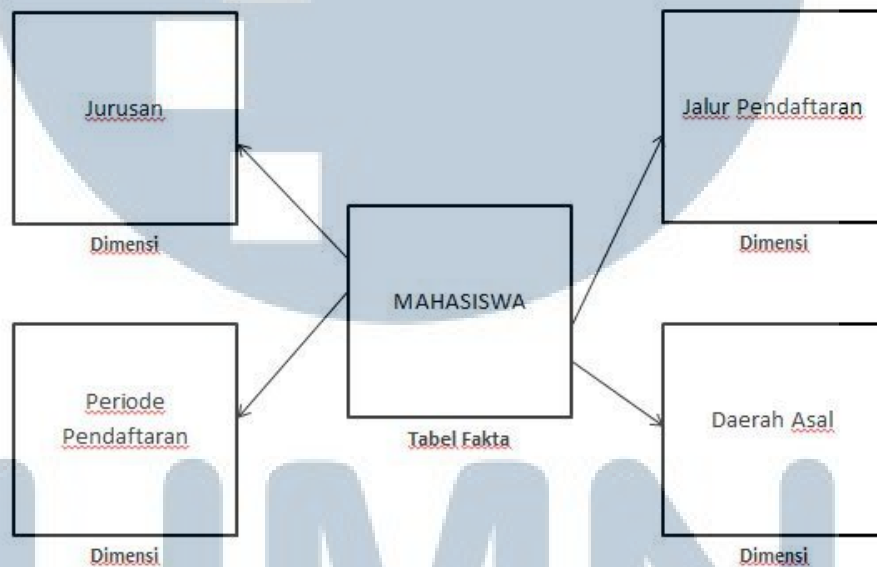
Harland, et al. (2011) dalam penelitiannya membangun sebuah portal BI berhasil menyediakan kebutuhan informasi dan *report* secara real-time dengan menggunakan *tools* Oracle *Business intelligence* Enterprise Edition.

3.4 Data Track

3.4.1 Dimensional Modeling

Dimensional modeling merupakan proses mewujudkan *business requirements* menjadi model dimensional sebuah sistem. Proses ini dilakukan dengan identifikasi tabel-tabel fakta dan dimensi-dimensi terkait, serta atribut-atribut penyertainya.

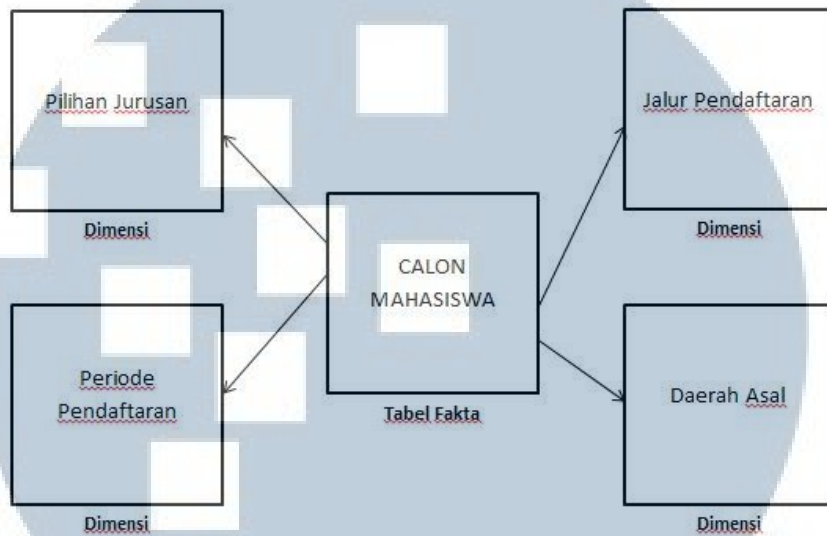
1) Skema Mahasiswa



Gambar 3.2 Model Skema Mahasiswa

Skema mahasiswa ini menjelaskan jumlah mahasiswa yang ditinjau dari periode pendaftarannya, daerah asalnya (negara, propinsi, kota, dan sekolah), jalur pendaftarannya, serta jurusan yang dipilih.

2) Skema calon mahasiswa



Gambar 3.3 Model Skema Candidate

Skema calon mahasiswa ini menjelaskan jumlah calon mahasiswa yang ditinjau dari periode pendaftarannya, daerah asalnya (negara, propinsi, kota, dan sekolah), jalur pendaftarannya, serta jurusan yang dipilih.

3.4.2 Physical Design

Physical design adalah fase di mana *database* dirancang. Desain *database* tersebut mencakup *database environment* yang ingin digunakan. *Database* yang dibuat ini merupakan tempat penyimpanan *data mart* yang akan dibuat.

Pada saat penelitian ini dilakukan, *database* operasional UMN menggunakan platform Oracle Database 10g Release 2 (10.2.0.1.0) Enterprise/Standard Edition For Microsoft Windows (32-bit). Oleh sebab itu, untuk *database* yang dijadikan sebagai *data source* di penelitian ini juga menggunakan platform dan struktur yang

sama. Bukan hanya itu, *database* ini juga berisi data yang sama dengan *database* yang dimiliki UMN. Akan tetapi, *database* tersebut terpisah seutuhnya dengan *database* yang dimiliki UMN dan hanya berisikan data yang berkaitan dengan analisis data. *Data warehouse* di penelitian ini juga menggunakan *database* yang sama. Meskipun secara fisik, *data source* dan *data warehouse* tersimpan dalam *database* yang sama, keduanya terpisah secara logis.

3.4.3 Extract, Transform, Load (ETL) Design & Development

ETL design & development merupakan proses mengekstraksi data, membersihkan dan mentransformasikan data, memasukkan data ke dalam *data warehouse* sesuai dengan desain yang telah dibuat.

Ekstraksi data dilakukan secara *ad-hoc* dengan pengambilan data langsung dari *data source*. Proses pengambilan data dari *data source* dilakukan dengan cara mengimpor tabel yang ingin diproses. Dengan OWB, *data source* tersebut terintegrasi dengan *repository* sehingga setiap perubahan yang dilakukan di *data source* akan menyebabkan perubahan data di *repository* juga. Hal inilah yang memungkinkan adanya konsistensi data.

Proses transformasi data dilakukan dengan *mapping* di OWB. *Mapping* bertujuan untuk menempatkan data dari *data source* ke tabel dimensi dan fakta yang bersesuaian.

Loading data bertujuan untuk memasukkan data yang telah ditransformasi ke dalam *data warehouse*. Proses *loading* tabel dimensi dilakukan secara *ad-hoc*, sedangkan untuk tabel fakta dilakukan secara berkala. Proses berkala tersebut

dilakukan dengan cara menempatkan *schedule* untuk proses *mapping* tabel-tabel fakta. Pembuatan *schedule* tersebut memanfaatkan fitur *scheduler* yang ada di OWB.

Scheduler merupakan fitur di OWB yang dapat dikenai pada suatu proses ETL untuk dijalankan sesuai dengan waktu yang dijadwalkan. Pemanfaatan fitur ini yaitu dengan pembuatan *schedule* secara mingguan dan per semester.

3.5 Business intelligence (BI) Application Track

3.5.1 BI Application Specification

Spesifikasi aplikasi BI harus disesuaikan dengan kebutuhan *business users*. Dari hasil indentifikasi kebutuhan *user*, didapatkan bahwa *user* membutuhkan sebuah aplikasi penyedia *report* yang dapat menampilkan perbandingan antara data mahasiswa dan calon mahasiswa UMN yang dapat ditinjau menurut batasan-batasan yang telah disebutkan sebelumnya. Informasi tersebut diharapkan dapat disajikan secara fleksibel dimana *user* dapat mengubah parameter dari data yang ingin ditampilkan. Untuk itu, aplikasi BI ini dibuat berpusat pada penyediaan informasi yang fleksibel dengan *pivot table* dan *interactive dashboards*.

3.5.2 BI Application Development

3.5.2.1 Report

Untuk memudahkan akses dan penyediaan *report* yang teragregasi, aplikasi BI ini juga menyediakan sebuah *dashboard* yang terhubung langsung dengan *report* terkait. Penyediaan *report* ini dipusatkan dalam tiga laporan utama, yaitu:

- 1) Laporan Calon Mahasiswa

Laporan calon mahasiswa berisi data-data dalam tabel fakta CUBE_CANDIDATE.

2) Laporan Mahasiswa

Laporan mahasiswa berisi data-data dalam tabel fakta CUBE_STUDENT.

3) Laporan Perbandingan Mahasiswa dan Calon Mahasiswa

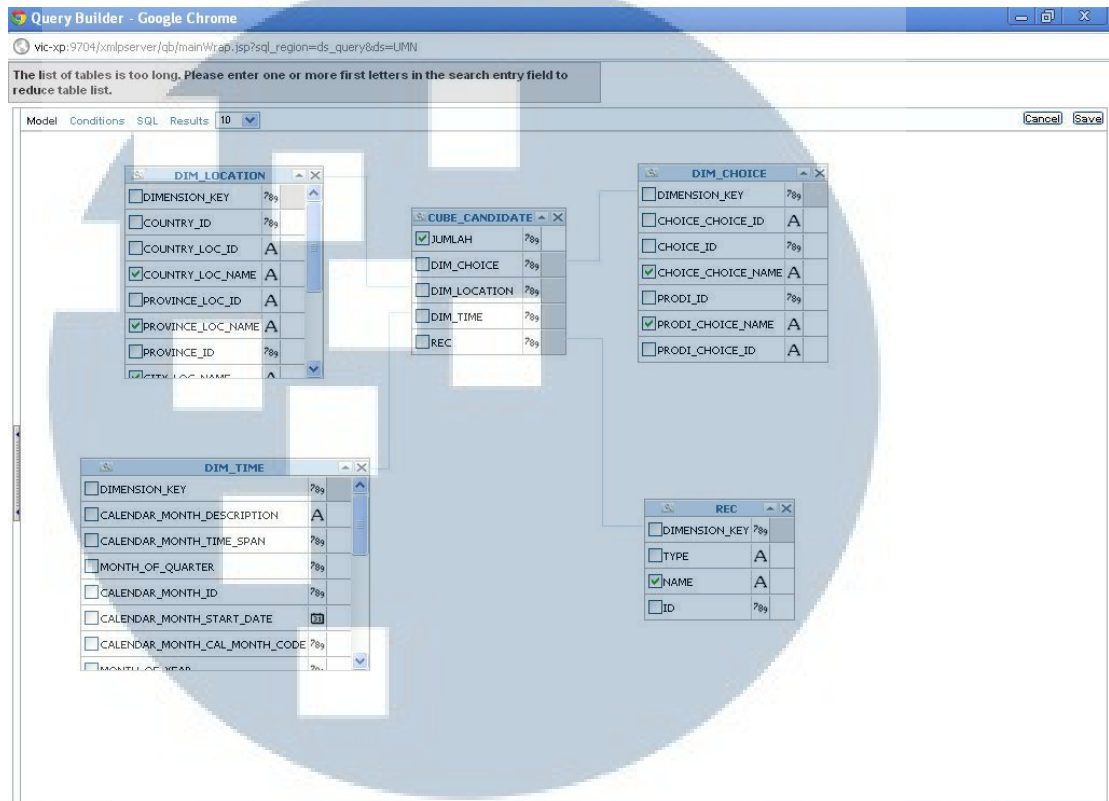
Laporan ini berisi gabungan data-data dari kedua tabel fakta CUBE_CANDIDATE dan CUBE_STUDENT.

Laporan-laporan di atas dibuat dengan *query builder* yang ada di Oracle BI *Publisher* (OBIP). Pada *query builder*, *query* di-generate dengan menggunakan:

1) *Model*

Model memungkinkan *user* menghubungkan masing-masing tabel berdasarkan hubungan logis masing-masing, dan memilih *field* yang akan ditampilkan nantinya.



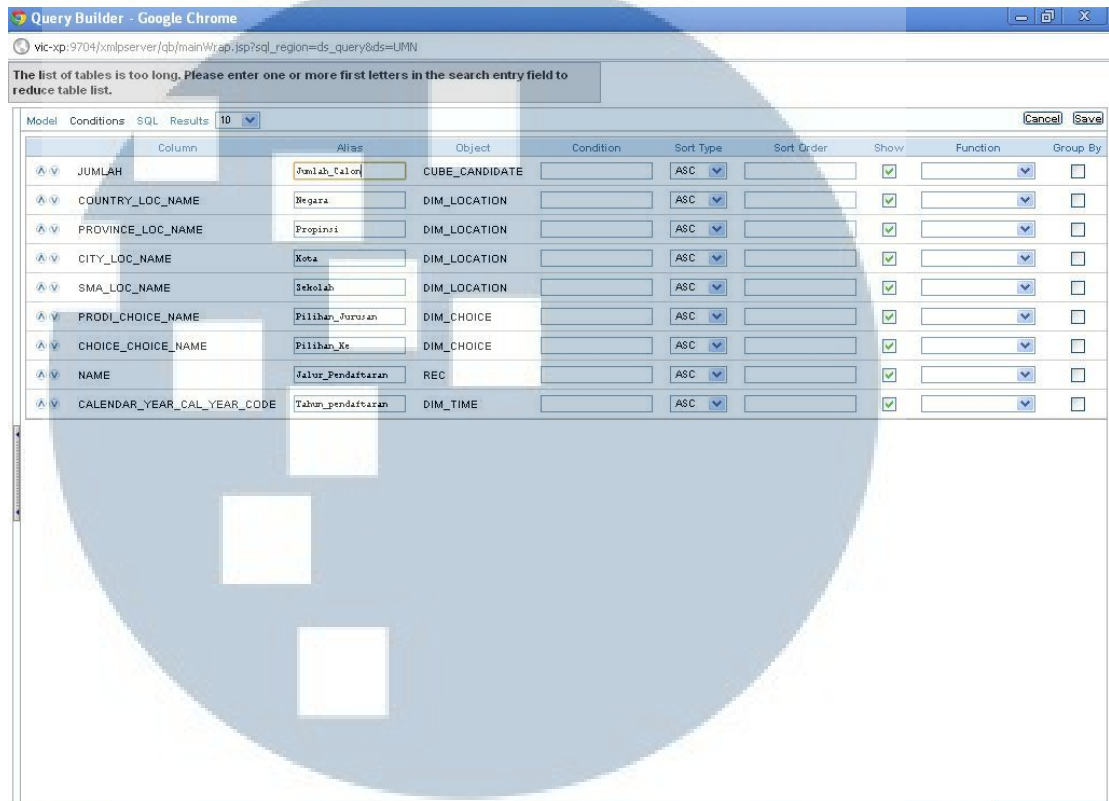


Gambar 3.4 Contoh Pembuatan Model di Query Builder

2) Conditions

Tampilan *conditions* digunakan untuk memberikan alias-alias bagi nama kolom yang akan ditampilkan sehingga dapat dimengerti oleh *user*. Selain itu juga digunakan untuk memberikan kondisi-kondisi tertentu.

UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 3.5 Contoh Pemberian *Conditions* di *Query Builder*

3) SQL

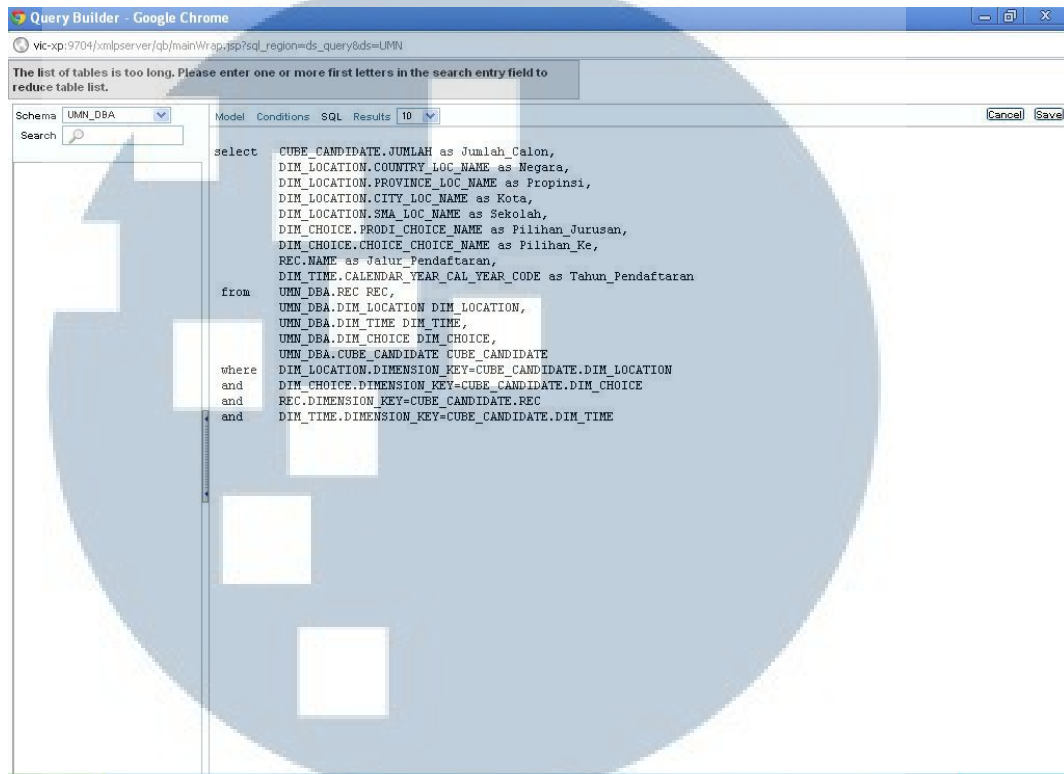
Setelah itu, secara otomatis, *query* yang sesuai dengan pemodelan di atas di-*generate*.

U
M
M
N

U
N
I
V
E
R
S
I
T
A
S

M
U
L
T
I
M
E
D
I
A

N
U
S
A
N
T
A
R
A



Gambar 3.6 Contoh SQL Generation di Query Builder

Report dari hasil query tersebut dapat ditampilkan dalam berbagai format.

Format-format yang ada di OBIP antara lain:

1) *Interactive*

Format *interactive* menyediakan fitur seperti *pivot table* di mana user dapat mengubah informasi yang ditampilkan berdasarkan parameter yang diinginkan.

2) HTML

Format HTML menampilkan hasil query dalam format *webpage*.

3) PDF

Pemilihan format PDF menyebabkan *report* diunduh secara otomatis sebagai *file* PDF.

4) RTF

Pemilihan format RTF menyebabkan *report* diunduh secara otomatis sebagai *file* RTF.

5) Excel

Pemilihan format Excel menyebabkan *report* diunduh secara otomatis sebagai *file* Excel.

6) MHTML

Pemilihan format MHTML menyebabkan *report* ditampilkan dalam format MIME HTML.

7) Powerpoint

Pemilihan format Powerpoint menyebabkan *report* diunduh secara otomatis sebagai *file* Powerpoint.

8) CSV

Pemilihan format CSV menyebabkan data dalam *report* ditampilkan satu persatu dan dipisahkan dengan tanda koma (“,”).

9) Data

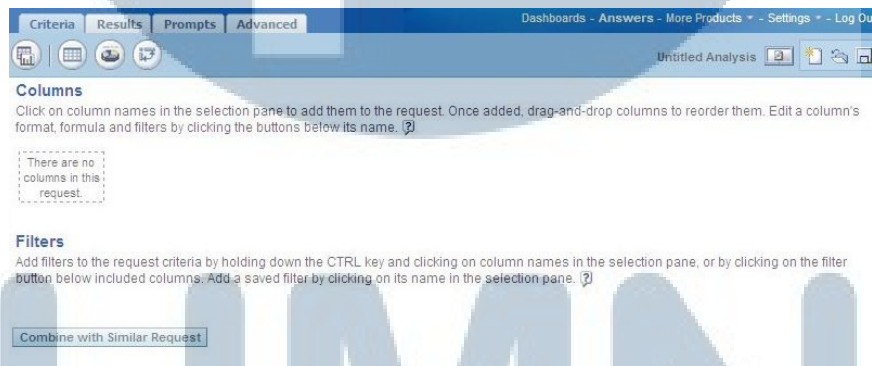
Pemilihan format data menyebabkan data dalam *report* ditampilkan satu persatu tanpa dipisahkan dengan tanda koma (“,”) atau dalam format XML.

3.5.2.2 Dashboard

Dashboard berisi halaman-halaman yang menampilkan analisis jumlah *student* dan *candidate* dalam berbagai macam format yang mudah dibaca dan dipahami oleh *end user*.

Pembuatan *dashboard* di OBIEE dilakukan dengan membuat *request* terlebih dahulu dengan Oracle BI Answers (OBIA). *Request* dimaksudkan untuk menentukan *criteria* kolom-kolom yang akan dianalisis untuk kemudian ditampilkan di *interactive dashboard*. Pembuatan *request* dilakukan dengan langkah:

1) *Criteria*

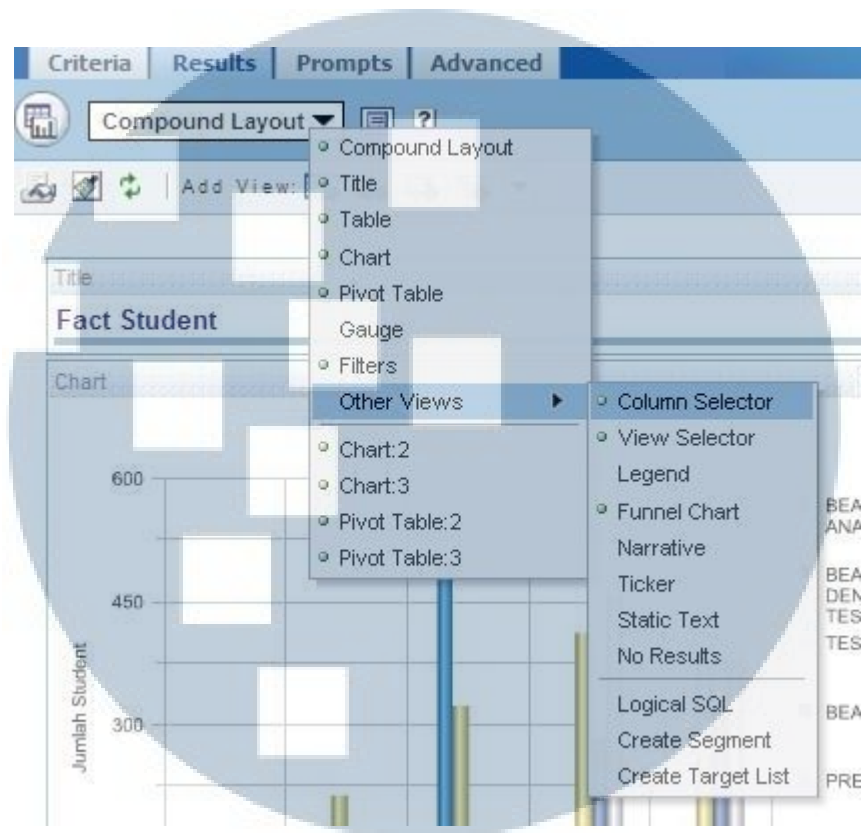


Gambar 3.7 Tampilan *Criteria* di BI Answers

Penentuan *criteria* dimaksudkan untuk melakukan *query* kolom-kolom yang akan dianalisis dan *filter* yang akan digunakan di *dashboard*.

2) *Result*

Setelah *query* berhasil dilakukan, hasilnya dapat ditampilkan dalam berbagai macam bentuk. Bentuk-bentuk itulah yang nantinya dapat dimasukkan ke dalam *dashboard*.



Gambar 3.8 Tampilan *Result* di BI Answers

3) *Prompt*

Pembuatan *prompt* dimaksudkan untuk memasukkan *filter* ke dalam *dashboard*. Dengan adanya *filter*, *dashboard* akan menjadi lebih interaktif. Analisis yang ditampilkan dapat disesuaikan dengan kebutuhan *user*.

Prompt ini dapat diterapkan pada lingkup *dashboard* yang berarti satu *filter* mempengaruhi semua halaman yang ada di *dashboard*. Selain pada lingkup *dashboard*, *prompt* juga dapat diterapkan pada lingkup halaman. Hal itu berarti *filter* hanya bekerja pada halaman itu saja.

3.6 *Deployment, Maintenance, Growth*

Dalam *stage deployment*, dilakukan *testing*, dokumentasi, serta *training* untuk semua *user* yang akan menggunakannya. *Maintenance* meneruskan *deployment* untuk memastikan bahwa sistem dapat terus berjalan dengan baik. Jika masih ada ruang untuk pengembangan, sistem dapat dikembangkan lagi dengan mengulang proses dari awal kembali.

Semua langkah di atas dilakukan dengan tetap menerapkan *project management* yang baik.

