Generator SQL Query

ZAde Chandra [1], Santi Sundari [2]

[1]KBK Sistem Informasi dan Database - Jurusan Teknik Komputer dan Informatika
Politeknik Negeri Bandung, Bandung 40012
Email : chandra@jtk.polban.ac.id

[2]KBK Rekayasa Perangkat Lunak - Jurusan Teknik Komputer dan Informatika
Politeknik Negeri Bandung, Bandung 40012
Email : santi@jtk.polban.ac.id

ABSTRAK

Pengguna informasi dalam sebuah sistem informasi manajemen (SIM) tidak harus selalu memiliki pengetahuan dan keahlian dalam cakupan teknologi Informasi.Sering kali pekerja dalam sebuah organisasi yang memiliki sistem informasi memiliki latar belakang pendidikan non IT.Sebagai pengguna, mereka seringkali dihadapkan pada kebutuhan pembuatan laporan yang sesuai dengan kebutuhan organisasi namun terkendala dengan penulisan sintak*Query* untuk me-*retrieve* data yang sudah tersimpan dalam database.

Artikel ini menguraikan satu model aplikasi yang dapat digunakan pengguna untuk menghasilkan sintak *SQL Query*yang akan digunakan dalam me-*retrieve* data. Melalui artikel ini diharapkan laporan dinamis dari suatu aplikasi pengolahan data dapat direalisasikan walau tidak ada pengguna yang memahami pemodelan data dan sintak *SQL Query*.

Kata Kunci
Generator SQL Query, laporan dinamis, Textual Database

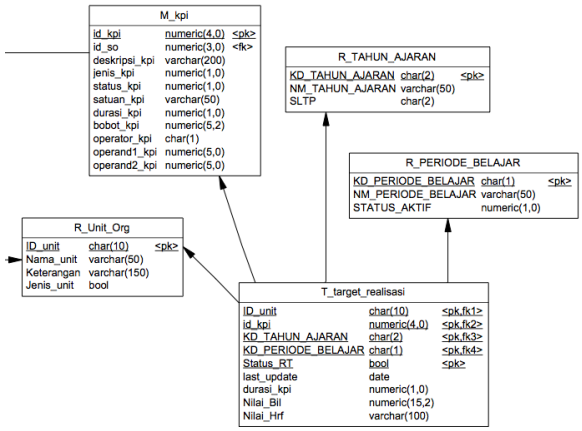
1. PENDAHULUAN

Dalam sebuah organisasi yang sudah menerapkan pengolahan data elektronis, seringkali dibutuhkan adanya laporan yang berasal dari database yang dimiliki. Hal ini mudah dipenuhi jika dimiliki SDM yang kompeten untuk memahami model data dan cara untuk me-*retrieve* nya. Dalam sebuah relational database, akan banyak relasi yang direalisasikan dalam sekumpulan *table*yang saling berelasi. Sehingga jika model data tidak terpahami, tentu pengguna akan cukup sulit untuk menemukan lokasi data yang dicarinya. Selain dari itu juga implementasi database relasional yang menerapkan proses Normalisasi tentunya akan membuat jumlah *table* lebih banyak karena adanya proses dekomposisi *table*untuk menjaga duplikasi yang tidak diperlukan, mengoptimalkan *redundancy* data dengan tetap menjaga konsistensi dan *integritas* data.

Setelah memahami model data dan lokasi *table* tempat menyimpan data yang dicari, maka permasalahan berikutnya adalah : bagaimana untuk mengambil dan menampilkan data yang dicari tersebut ke dalam sebuah laporan. Untuk pengguna yang memahami *Structured Query Language* (SQL) tentu hal ini dapat dengan mudah direalisasikan.

Tidak semua organisasi yang sudah menerapkan pengolahan data elektronik memiliki staf dengan keahlian di bidang teknologi Informasi, dan karenanya sangat dibutuhkan adanya tool atau perangkat lunak aplikasi yang dapat memudahkan mereka dalam mengakses data yang dimiliki. Keberadaan tool atau aplikasi ini tentunya akan meningkatkan kebergunaan data yang sudah dimiliki, baik untuk proses pelaporan, analisa maupun pengembangan kebijakan di organisasi.

Contohnya dalam sebuah sistem pengukuran kinerja organisasi untuk menampilkan data kinerja Akademik tahun 2013/2014 untuk Prodi D3 Teknik Informatika, maka diperlukan akses ke *table* data berikut : T_Target_realisasi, R_tahun_ajaran, R_Periode_belajar, R_unit_org dan M_KPI. Relasi antar *tabel* data ditunjukkan pada Gambar 1. Data utama berada di *tabel* T_Target_realisasi, namun agar lebih informatif dan sesuai dengan kriteria permintaan dalam contoh kasus, maka diperlukan tabel deskripsi yang berada di R_tahun_ajaran dan R_Periode_belajar, sedangkan untuk menampilkan deskripsi Indikator Kinerja dibutuhkan deskripsi dalam M_KPI serta untuk memfilter dan menampilkan nama Unit organisasi diperlukan akses terhadap *tabel* R_Unit_org.



Gambar 1: *Relationship* antar *table* di database Kinerja

Atribut ID di *tabel* Transaksi T_Target_Realisasi memiliki *primary key* yang diturunkan dari *tabel* Referensi dan*tabel* Master. *Tabel* transaksi berisi *Foreign key* yang mengacu ke setiap *tabel* referensi dan master (ditampilkan dengan menggunakan panah relasi). Untuk menampilkan realisasi dari sebuah indikator kinerja organisasi, maka diperlukan *filtering* status_RT = 1. Data nilai indikator yang akan ditampilkan berada di atribut Nilai_bil dan/atau Nilai_Hrf.

Dengan memahami setiap atribut di atas, maka permasalahan berikutnya adalah bagaimana sintak untuk me-*retrieve* setiap data dalam *tabel* database tadi sehingga dapat diproses lebih lanjut ?

Sintaks query (bagi pengguna yang memahami) untuk me - *retrieve* data di atas dapat dituliskan sebagai berikut :

```
SELECT u.Nama_unit, k.deskripsi_Kpi,
          r.Nilai_Bil, r.Nilai_Hrf
FROM R_Unit_org u, M_Kpi k,
    R_tahun_ajaran t,
    R_periode_belajar p,
    T_target_realisasi r
WHERE (r.id_unit = u.id_unit and
    r.id_kpi = k.id_kpi and
    r.kd_tahun_ajaran = t.kd_tahun_ajaran
    and r.status_RT = 1
    and t.Kd_tahun_ajaran = '04'
    and u.id_unit = '2');
```

Penulisan *Query* dalam sintak SQL di atas memerlukan pengetahuan, dan keterampilan pengguna, **bagaimana dengan pengguna yang tidak memahami sintaks SQL** ?

Untuk membantu pengguna, maka penulis mengusulkan untuk mengembangkan perangkat lunak aplikasi yang dapat membantu pengguna dalam menuliskan *sintak Query* tersebut. Pengguna perlu dimudahkan dengan hanya memilih atribut yang akan ditampilkan dan memberikan syarat yang relevan untuk kemudian aplikasi men-*generate* sintak SQL nya.

Tantangan teknis: Tantangan utama adalah bagaimana *User Interface* dari aplikasi dapat memberikan kemudahan bagi pengguna untuk menampilkan kelompok informasi yang dapat diakses. Dengan demikian maka pengguna tidak perlu mengetahui lebih detil struktur database serta relasinya dalam menampilkan informasi.

Dari kelompok informasi ini, berikutnya pengguna diminta untuk memilih tahun dan unit organisasi tertentu yang akan ditampilkan datanya. Proses berikutnya, pengguna akan memilih nama atribut yang akan ditampilkan serta nama atribut yang akan dipergunakan sebagai pem-*filter* dalam syarat *Query* yang akan diberikan (contoh di kasus ini adalah tahun ajaran = 2013/2014 dan Unit Organisasi nya adalah Prodi D3 Teknik Informatika)

Tantangan teknis berikut nya adalah bagaimana menterjemahkan *User Interface* dalam aplikasi ini menjadi sintak SQL yang dapat dieksekusi dalam Query untuk menampilkan data yang sudah dipilih. Sintak SQL ini harus sesuai dengan ANSI SQL tertentu dan harus dapat menampilkan data sesuai dengan semantik yang pengguna inginkan.

Kontribusi dari artikel ini adalah :

- Memberikan pemahaman kepada pengguna untuk mengetikkan sintak SQL dalam me-*retrieve* data sesuai dengan kebutuhan.
- Artikel ini juga diharapkan dapat memudahkan pengguna dalam menampilkan data dari database walaupun dilakukan oleh pengguna yang tidak memiliki keterampilan khusus dalam menuliskan sintak SQL.

2. PEMODELAN DATA DAN SOLUSI

Dalam sub bab ini akan disampaikan beberapa notasi yang dapat memberikan pemahaman dalam memenuhi kebutuhan data akses dari database.
Diawali dengan penulisan Sintak SQL dan tentunya penulis akan memberikan alternatif *generator SQL query* sehingga pengguna dapat dengan mudah menghasilkan sintak SQL tanpa mengingat *keyword* dan urutan perintah yang harus disusun.

Pertama kali akan diperkenalkan notasi dalam pemodelan data. Dalam sub bab ini akan diuraikan cara mendefinisikan notasi untuk melakukan *query* data tertentu. Kemudian, penulis akan memperkenalkan kerangka calon aplikasi untuk *generator SQL Query* dalam menemukan semua pertanyaan yang valid dalam mencapai tantangan teknis di atas.

2.1 Notasi dalam Data Model

Dalam sebuah database *D* dengan *m* buah skema relasi/*table* R_1, R_2, …, R_m, akan diperoleh relasi R, dengan R[i] menunjukkan kolom ke [i] dari skema relasi tersebut , dan col(R) = $\{R[i]\}_{i = 1,2, \ldots}$ | dimana col(R) menyatakan

himpunan kolom R. Untuk *tuple* t dalam R, maka t[i] menunjukkan nilai data di baris pada kolom *R [i]*. [1] [3]
Jika *G (V, E)* menunjukkan skema grafik dari *D*dan simpul *V* mewakili skema relasi/*table* di D, dan *edge* di E merepresentasikan *foreign key* antara dua relasi: maka akan ada anak panah dari Rj ke Rk di E. *Primary key* didefinisikan dengan Rk sedangkan *foreign key* didefinisikan melalui Rj.

Dalam contoh database di gambar 1 diperoleh 4 relasi, dengan garis berpanah menunjukkan referensi *foreign key* (yaitu, tepi dalam skema graf berarah). Ada empat kolom teks: R_Unit_org.ID_unit, M_KPI.ID_KPI,
R_Tahun_ajaran.Kd_tahun_ajaran,
dan R_periode_belajar.Kd_periode_belajar
yang menunjukkan adanya referensi dari *foreign key* di skema relasi T_Target_realisasi ke-4 skema relasi lain dalam menjaga *integrity* data. Dengan demikian tidak akan ada id_unit di *table* T_target_realisasi yang tidak ada di R_unit_org, dst.

2.2 Relationship antar Table

Untuk me-*retrieve* data di setiap *table* yang memiliki *relationship* dalam sebuah *databaseD*, maka perlu diketahui *Relationship* antar *table* tersebut sehingga memudahkan proses *join*dan tidak mengakibatkan perkalian relasi dari kedua isi *tuple* yang dimiliki *databaseD*.

DEFINISI 1.*example table T is a table with rows{r} and columns col(T), where each cell of a row is either a string (i.e., one or more tokens) or empty. For a row r □T , let r[i] denote its value on the column i □ col(T), and let r[i] = □ if r[i] is empty.*[1][8]

Untuk memudahkan pemahaman, penulis mengasumsikan bahwa T tidak mengandung baris kosong atau kolom kosong.

Untuk menggambarkan relasi antar 2 *table* maka proses *join* dilakukan melalui *foreign key* dari *relationship* yang sudah didefinisikan. Sebuah *Project Join Query (J, C)* dapat dituliskan melalui :
i) *join graph* J ⊆ G, yaitu, subgraf dari skema *graf* G dari *database D* yang mewakili semua relasi antar *table* yang dimiliki *database D* dengan *vertex J*serta semua *edge dari J.* Himpunan kolom dari J adalah kolom yang merelasikan *Table* J ke setiap table lain nya yang memiliki *relationship*; dan ii) Sekumpulan kolom C ⊆ col (J) dari table J, yang menghasilkan *Join* akan direlasikan menggunakan kolom tersebut ke table lain. Misalkan A (J, C) adalah sebuah relasi yang dihasilkan.

Secara informal, sebuah *Query Join* dikatakan valid jika setiap baris dari T dapat ditemukan dalam hasil *Query Join di database D*, dengan semua kolom T memiliki peta ke relasi yang dihasilkan.
DEFINISI 2. *Given an example table T in a database D, consider a project join query Q = (J,C,φ) with a mapping φ : col(T) □ C from columns of T to projection C. Let A(Q) be the resulting relation of the project join query. Q is valid w.r.t. a row r ∈ T iff there exists a tuple t □A(Q) s.t. for any column i □col(T),r[i] = □ (r[i] □t[φ(i)], [1]*
where "x □y" denotes that string x is contained in string y. A project join query Q is valid w.r.t. an example table T , iff Q isvalid w.r.t. every row of T .

DEFINITION 3.*(Minimal Valid Project Join Queries) A valid project join query Q = (J , C, φ) w.r.t. an example table T is minimal iff i) without directions on edges, J is an undirected tree; and ii) for any degree-1 vertex (relation) R in J , there exists a column i □col(T) s.t. φ(i) □ col(R), i.e., a column of T is mapped to a column of R.* [1]

2.3 Alternatif *Generator Query Sintax*

Untuk mempermudah pengguna dalam menghasilkan Sintak*Query*, maka dapat dikelompokkan kumpulan data apa saja yang akan dibantu untuk menampilkan datanya. Kelompok kumpulan data ini disiapkan berdasarkan *relationship* antar *table* yang dimiliki dalam suatu *database D*.Setelah pengelompokkan data ini disiapkan, maka berikutnya dari setiap *relationship* antar *table* ditampilkan semua kolom yang dimiliki *table* tsb. Dengan demikian, pengguna dapat memilih data apa saja yang ingin ditampilkan atau ingin dijadikan syarat *query*.
Dengan menyediakan kelompok data yang berelasi serta kolom data yang dapat ditampilkan atau dijadikan persyaratan *query*, maka *Syntax Query* berikut dapat digenerate :

```
SELECT C1, C2, … ,  Cm
    FROM T1, T2,…
Where <expresion_boolean>
```

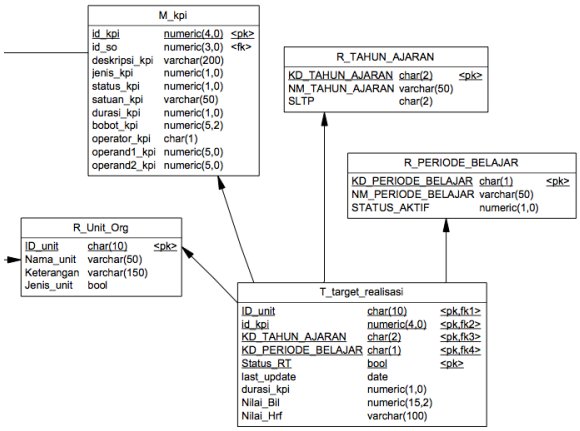
3. ARSITEKTUR SOLUSI

Sesuai dengan alternatif *generator sintax query* yang telah disampaikan, maka perlu dilakukan beberapa rencana pengembangan yang dilakukan dalam penelitian.
Rencana pengembangan itu adalah sbb :

a) Menggabungkan beberapa *tabel* untuk menjawab kebutuhan menampilkan data tertentu.
   Cakupan penampilan data dalam artikel ini adalah menampilkan indikator kinerja organisasi berdasarkan perspektif tertentu dengan memperhatikan Unit Organisasi dan tahun ajaran.

   Dengan demikian akan dilakukan *join query* dengan struktur tabel sebagai mana di gambarkan dalam gambar 2 di halaman berikut.

b) Mempersiapkan sekumpulan *relationship* untuk mendefinisikan *join query* yang dapat memenuhi kebutuhan di point 1.



Gambar 2 :*Join Query* untuk study kasus Pengukuran Kinerja

Implementasi sekumpulan tabel dan kolom untuk kebutuhan point 1 dapat dilakukan dengan mempersiapkan metadata yang akan menyimpan struktur *tabel*, dan struktur kolom dari struktur database yang dimiliki.

Skema Relasi dari meta data di atas adalah :

```
P_daftbl (Kdtbl, NmTbl, Kdjnstbl)
P_jnstbl (Kdjnstbl, KtJnsTbl)
P_keytbl (Kdtbl, NoKol, Nmtbl, NmKol,
JnsData, PjgData)
P_strtbl (Kdtbl, NoKol, Nmtbl, NmKol,
   KtKol1,  KtKol2,  KtKol3,  KtKol4,
   KtKol5, JnsData, PjgData, PjgKol1,
   PjgKol2, PjgKol3, PjgKol4, PjgKol5,
   StaNull, Stakey, TblRef, KdRef,
   NmRef, KtRef)
```

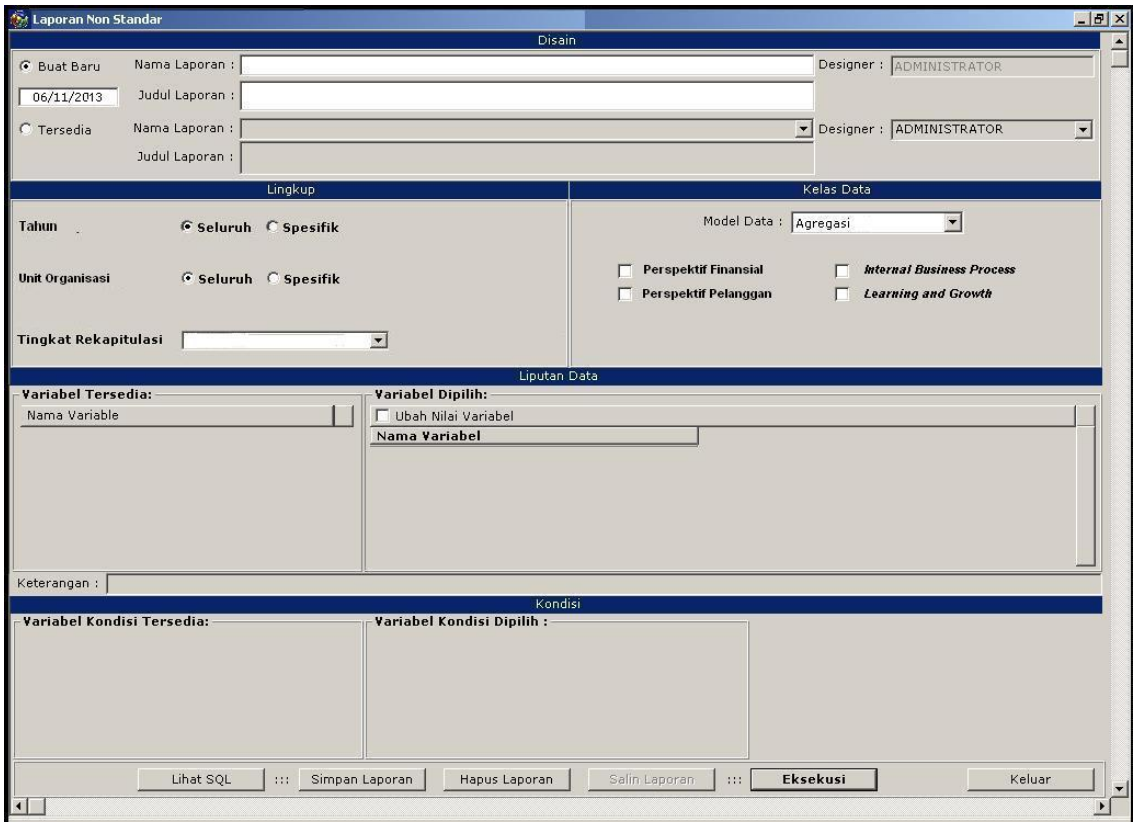
c) Menampilkan semua atribut yang dimiliki oleh *join query* yang didefinisikan untuk menerima pilihan kolom yang akan ditampilkan dan / atau menerima kondisi kolom yang akan diberikan.
Skema relasi dari meta data untuk menampilkan kolom dan kondisi untuk syarat Query adalah :

```
P_daftbl (Kdtbl, NmTbl, Kdjnstbl)
P_jnstbl (Kdjnstbl, KtJnsTbl)
P_jnsstr(Kd_jenis_sklh,KdTbl,NoKol)
P_klstbl (KdKls,KdModKls, KdTbl)
P_keytbl (Kdtbl, NoKol, Nmtbl, NmKol,
JnsData, PjgData)
P_strtbl (Kdtbl, NoKol, Nmtbl, NmKol,
   KtKol1,  KtKol2,  KtKol3,  KtKol4,
   KtKol5, JnsData, PjgData, PjgKol1,
   PjgKol2, PjgKol3, PjgKol4, PjgKol5,
   StaNull, Stakey, TblRef, KdRef,
   NmRef, KtRef)
P_Kndvld (KdTblVar, NoKolKtKnd,
   NmTblVar, NmTblKnd, NmKolKnd,
   KtKolKnd)
```

d) Menampung semua lingkup data dalam variabel untuk di-*concatenate* dengan sintak SQL *query* beserta persyaratan *query* yang dientry kan oleh pengguna.

e) Menyediakan mekanisme eksekusi query dari aplikasi agar *retrieve* data berhasil dilakukan.

Model interface yang dapat dikembangkan dengan arsitektur Solusi di atas dapat ditampilkan sebagai berikut :



Gambar 3 :Model *Interface* untuk *Generator SQL Query*

4. EVALUASI ATAS HASIL PERCOBAAN

Sintak Query dalam SQL yang dihasilkan oleh aplikasi yang diuraikan dalam sub bab sebelumnya adalah :

```
SELECT u.Nama_unit, k.deskripsi_Kpi,
          r.Nilai_Bil, r.Nilai_Hrf
FROM R_Unit_org u, M_Kpi k,
    R_tahun_ajaran t,
    R_periode_belajar p,
    T_target_realisasi r
WHERE (r.id_unit = u.id_unit and
    r.id_kpi = k.id_kpi and
    r.kd_tahun_ajaran = t.kd_tahun_ajaran
    and r.status_RT = 1
    and t.Kd_tahun_ajaran = '04'
    and u.id_unit = '2');
```

Dan setelah Sintak SQL di eksekusi, ditampilkan lah data dengan isi sebagai berikut :

Nama Unit	Deskripsi Kpi	Nilai Bil	Nilai Hrf
Prodi D3 Teknik Informatika	Rata-rata masa studi lulusan	3.00	
Prodi D3 Teknik Informatika	Rata-rata IPK lulusan	2.98	
Prodi D3 Teknik Informatika	Pencapaian prestasi mahasiswa di tingkat propinsi/ wilayah, nasional, dan inte	7.00	
Prodi D3 Teknik Informatika	Rasio keberhasilan lulusan sesuai dengan permintaan stakeholder.	0.85	
Prodi D3 Teknik Informatika	Rasio jumlah mahasiswa yang diterima terhadap jumlah mahasiswa yang ikut	0.10	
Prodi D3 Teknik Informatika	Jumlah penelitian dosen tetap selama tiga tahun terakhir.	0.75	
Prodi D3 Teknik Informatika	Jumlah kegiatan PkM dosen tetap selama tiga tahun terakhir	0.75	
Prodi D3 Teknik Informatika	Kegiatan kerjasama dengan instansi di dalam negeri dalam tiga tahun terakhir	9.00	
Prodi D3 Teknik Informatika	Kegiatan kerjasama dengan instansi di luar negeri dalam tiga tahun terakhir	2.00	
Prodi D3 Teknik Informatika	Sistem penjaminan mutu perguruan tinggi yang mencakup kebijakan dan pera	0.85	
Prodi D3 Teknik Informatika	Keberadaan dan keefektifan sistem audit internal, dilengkapi dengan kriteria d	0.85	
Prodi D3 Teknik Informatika	Persentase dana perguruan tinggi yang berasal dari mahasiswa (SPP dan dar	0.20	
Prodi D3 Teknik Informatika	Dana penelitian dalam tiga tahun terakhir.	750000.00	
Prodi D3 Teknik Informatika	Kecukupan dan mutu prasarana yang dikelola perguruan tinggi.	0.90	
Prodi D3 Teknik Informatika	Sistem informasi dan fasilitas yang digunakan perguruan tinggi dalam adminis	8.00	
Prodi D3 Teknik Informatika	Pelaksanaan survei kepuasan dosen, pustakawan, laboran, teknisi, tenaga ad	98.00	
Prodi D3 Teknik Informatika	Manfaat dan kepuasan mitra kerja sama.	85.00	
Prodi D3 Teknik Informatika	Rata-rata masa studi lulusan	3.00	
Prodi D3 Teknik Informatika	Rata-rata IPK lulusan	2.98	
Prodi D3 Teknik Informatika	Pencapaian prestasi mahasiswa di tingkat propinsi/ wilayah, nasional, dan inte	7.00	
Prodi D3 Teknik Informatika	Rasio keberhasilan lulusan sesuai dengan permintaan stakeholder.	0.85	
Prodi D3 Teknik Informatika	Rasio jumlah mahasiswa yang diterima terhadap jumlah mahasiswa yang ikut	0.10	
Prodi D3 Teknik Informatika	Jumlah penelitian dosen tetap selama tiga tahun terakhir.	0.75	
Prodi D3 Teknik Informatika	Jumlah kegiatan PkM dosen tetap selama tiga tahun terakhir	0.75	
Prodi D3 Teknik Informatika	Kegiatan kerjasama dengan instansi di dalam negeri dalam tiga tahun terakhir	9.00	
Prodi D3 Teknik Informatika	Kegiatan kerjasama dengan instansi di luar negeri dalam tiga tahun terakhir	2.00	
Prodi D3 Teknik Informatika	Sistem penjaminan mutu perguruan tinggi yang mencakup kebijakan dan pera	0.85	
Prodi D3 Teknik Informatika	Keberadaan dan keefektifan sistem audit internal, dilengkapi dengan kriteria d	0.85	
Prodi D3 Teknik Informatika	Persentase dana perguruan tinggi yang berasal dari mahasiswa (SPP dan dar	0.20	
Prodi D3 Teknik Informatika	Dana penelitian dalam tiga tahun terakhir.	750000.00	
Prodi D3 Teknik Informatika	Kecukupan dan mutu prasarana yang dikelola perguruan tinggi.	0.90	

Nama Unit	Deskripsi Kpi	Nilai Bil	Nilai Hrf
Prodi D3 Teknik Informatika	Sistem informasi dan fasilitas yang digunakan perguruan tinggi dalam adminis	8.00	
Prodi D3 Teknik Informatika	Pelaksanaan survei kepuasan dosen, pustakawan, laboran, teknisi, tenaga ad	98.00	
Prodi D3 Teknik Informatika	Manfaat dan kepuasan mitra kerja sama.	85.00	

5. SIMPULAN DAN PENGEMBANGAN

Melalui artikel ini telah disampaikan satu alternatif pemberian perangkat lunak aplikasi yang dapat menghasilkan sintaks SQL Query. Sintaks SQL Query ini telah menerapkan join query berbasis table contoh yang didefinisikan.

Kunci utama dalam menjawab tantangan teknis yang disampaikan adalah dengan membuat meta data untuk menelusuri table, relasi antar table, kolom dan keberadaan foreign key . Dengan memodelkan meta data tersebut maka aplikasi dapat mengarahkan pengguna untuk memilih daftar kolom dalam table untuk dipilih pengguna dalam memenuhi kebutuhan data dari database yang dimiliki.

Hasil pengamatan dan penelitian yang dilakukan , filtering dalam SQL sintaks masih dapat dikembangkan dengan menambahkan operator relasional dengan jumlah yang dapat ditambahkan sesuai dengan kebutuhan pengguna, begitu pula untuk kebutuhan group query yang melibatkan fungsi-fungsi agregation masih perlu disempurnakan.

DAFTAR PUSTAKA

[1] Yanyan Shen, S. Chaudhuri, and Bolin Ding. *Discovering Queries based on Example Tuples*. In SIGMOD, 2014.

[2] S. Agrawal, S. Chaudhuri, and G. Das. *Dbxplorer: A system for keyword-based search over relational databases*. In ICDE, 2002.

[3] L.Blunschi,C.Jossen,D.Kossmann,M.Mori,andK.Stockinger. *Soda: Generating sql for business users*. PVLDB, 2012.

[4] J. Fan, G. Li, and L. Zhou. *Interactive sql query suggestion: Making databases user-friendly.* In ICDE, 2011.

[5] V. Hristidis and Y. Papakonstantinou. *Discover: Keyword search in relational databases.* In VLDB, 2002.

[6] H. V. Jagadish, A. Chapman, A. Elkiss, M. Jayapandian, Y. Li, A. Nandi, and C. Yu. *Making database systems usable*. In SIGMOD, 2007.

[7] L. Qian, M. J. Cafarella, and H. V. Jagadish. *Sample-driven schema mapping.* In SIGMOD, 2012.

[8] L. Qin, J. X. Yu, and L. Chang. *Keyword search in databases: the power of rdbms*. In SIGMOD, 2009.

[9] Q. T. Tran, C.-Y. Chan, and S. Parthasarathy. *Query by output*. In SIGMOD, 2009.

[10] M.Zhang,H.Elmeleegy,C.M.Procopiuc,andD.Srivastava. *Reverse engineering complex join queries.* In SIGMOD, 2013.