MODEL KONTROL TRANSAKSI RDBMS MENGGUNAKAN TRIGGER DAN WAKTU SERVER

Joko Triyono[1], Prita Haryani[2], Aditya Padmanaba[3]
[1,2]Teknik Informatika, Fakultas Teknologi Industri, IST AKPRIND Yogyakarta
[3]Mahasiswa Teknik Informatika, Fakultas Teknologi Industri, IST AKPRIND Yogyakarta
jack@akprind.ac.id[1], pritaharyani@akprind.ac.id[2], adityapadmanaba@gmail.com[3]

ABSTRACT

The development of technology has become extraordinary, in the 4.0 era, the field of information technology has become a very important part in every activity or job. In the development of information systems at this time needed in addition to a user friendly user interface as well as data management that is always changing. In RDBMS there is a trigger facility that allows capturing all behavior that occurs with data in an RDBMS. In the development of current information systems the time of events (server time) is also a very important component to assess whether the data is valid and current. So the RDBMS design must be able to capture and control all activities that occur in an information system without having to interfere with even touching the information system application side. The security and data history of an information system must always be maintained in a sustainable and systematic manner, so that the data can be analyzed to obtain patterns in subsequent system development. This research produces control modeling in an RDBMS-based transactional information system using triggers and server time to record all events in an RDBMS so that an RDBMS development data is obtained from time to time that can be used for further analysis.
Keywords: *rdbms, trigger, server.*

INTISARI

Perkembangan teknologi menjadi sangat luar biasa, di era 4.0 maka bidang teknologi informasi menjadi bagian yang sangat penting peranannya dalam setiap kegiatan atau pekerjaan. Dalam pengembangan sistem informasi saat ini dibutuhkan selain user interface yang user friendly juga pengelolaan data yang selalu berubah-ubah. Dalam RDBMS telah tersedia fasilitas trigger yang memungkinkan menangkap semua perilaku yang terjadi terhadap data pada sebuah RDBMS. Dalam perkembangan sistem informasi saat ini waktu kejadian (waktu server) juga menjadi komponen yang sangat penting untuk menilai apakah data yang ada itu valid dan terkini. Sehingga perancangan RDBMS harus bisa menangkap dan mengontrol semua kegiatan yang terjadi pada sebuah sistem informasi tanpa harus mengganggu bahkan menyentuh sisi aplikasi sistem informasi. Keamanan dan histori data sebuah sistem informasi harus selalu terjaga secara berkelanjutan dan tersistem, sehingga data tersebut bisa dianalisis untuk mendapatkan pola dalam pengembangan sistem selanjutnya. Penelitian ini menghasilkan pemodelan kontrol dalam sebuah sistem informasi transaksional berbasis RDBMS dengan menggunakan trigger dan waktu server untuk merekam semua kejadian dalam sebuah RDBMS sehingga diperoleh sebuah data perkembangan RDBMS dari waktu ke waktu yang bisa digunakan untuk analisis selanjutnya.
Kata Kunci: rdbms, trigger, server

PENDAHULUAN

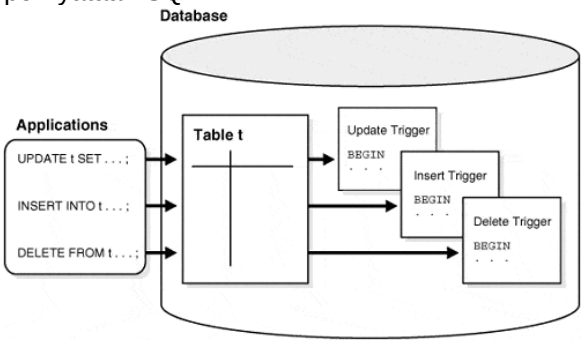
Perkembangan teknologi menjadi sangat luar biasa, di era 4.0 maka bidang teknologi informasi menjadi bagian yang sangat penting peranannya dalam setiap kegiatan atau pekerjaan. Seiring dengan meningkatnya kebutuhan akan teknologi informasi tersebut, kemajuan teknologi informasi juga mengalami perkembangan yang sangat pesat, tidak luput dalam perancangan RDBMS tidak hanya dalam merancang tabel dan relasi antar tabel saja tetapi telah merambah ke bagian untuk mencatat histori transaksi. Sehingga rancangan RDBMS saat ini harus sudah bisa mengakomodir dan mencatat semua kejadian yang terjadi pada tabel-tabel pada RDBMS dan juga kebutuhan akan kapan kejadian dari perubahan tersebut dilakukan. RDBMS sudah seperti benda hidup yang selalu bergerak dan berkembang dan harus memiliki rekam medis. Trigger sebagai bagian yang include dalam paket RDBMS bisa mengatasi dan menghandel semua kejadian dengan mengambil event-event yang terjadi pada tabel-tabel yang ada.

Menjadi sesuatu yang sulit dibayangkan jika output sebuah sistem informasi menjadi tidak konsisten atau bahkan menyesatkan, dimana terjadinya perubahan pada data tetapi tidak pernah tercatat apa yang berubah dan kapan terjadinya perubahan data tersebut. Dan pasti data itu akan bergerak dan berubah baik itu karena penambahan, perubahan maupun penghapusan data. Perkembangan sistem informasi yang begitu cepat perlu mendapak

dukungan dari sistem pencatatan yang tepat dan aman, sehingga keberadaan sistem informasi bisa membantu dalam pengambilan sebuah kebijakan manajerial. Jangan sampai keberadaan sistem informasi malah membebani manajerial.

Penelitian telah menghasilkan sebuah pemodelan kontrol dalam sebuah sistem informasi transaksional berbasis RDBMS dengan menggunakan trigger dan waktu server untuk merekam semua kejadian dalam sebuah RDBMS sehingga diperoleh sebuah data perkembangan RDBMS dari waktu ke waktu yang bisa digunakan untuk analisis data yang disajikan dalam sebuah sistem informasi tersendiri. Begitu juga dengan model aplikasi system informasi dan aplikasi analisis system informasi.

Dalam sebuah penelitian tentang Sistem *Trigger Database* Pada SIAKAD Informatika oleh (Dedianto, 2013) disimpulkan bahwa Penerapan *trigger* pada *database* membantu dalam proses pengamanan data secara rutin karena proses backup dilakukan setiap terjadi perubahan pada *database*. Dengan adanya *trigger* maka beban dari admin bisa dikurangi karena tidak perlu membackup data secara manual. *Trigger* yang dibuat akan membuat *backup* seluruh data dengan tidak mengubah sedikit pun data aslinya. Jika terjadi kegagalan saat melakukan kueri pada MySQL maka *trigger* tidak akan tereksekusi. Dari sisi keamanan, dalam sebuah penelitian tentang Analisis Keamanan Serangan *Sql Injection* Berdasarkan Metode Koneksi *Database* oleh (S, Suwanto, & Triyono, 2017) disimpulkan bahwa metode koneksi *database* menjadi salah satu faktor penentu dalam menjaga keamanan RDBMS dari serangan *SQL Injection*, sehingga diperlukan sebuah metode koneksi yang paling tepat dana aman. Dalam penelitian yang berbeda (Triyono, 2015) di peroleh hasil tentang pengembangan sebuah *prototype* sistem informasi yang di kombinasikan dengan jejaring sosial *twitter*, dimana jejaring *twitter* digunakan petani untuk melaporkan semua kegiatannya ke sistem informasi, dengan menggunakan fasilitas *APIs* (*Aplication Programming Language*) maka informasi yang masuk akan di kirimkan ke sistem informasi dengan menggunakan *account* dari *twitter* pengirim. Dengan metode ini secara teknologi dan biaya petani tidak mengalami kesulitan dalam melaporkan kegiatannya, sedangkan dari sisi investor akan bisa melihat perkembangan investasinya. Tentang kepuasan pengguna *website* (Haryani, 2016) disimpulkan bahwa dari hasil uji hipotesis menunjukkan bahwa dimensi *citizen support*, *content and appreance of information, reability, functionality of the interaction environment, trust, dan dimensi ease of use* berpengaruh terhadap kualitas layanan *e-government*. Kualitas layanan *e-government* berpengaruh terhadap kepuasan pengguna *website*, kualitas layanan *e-government* berpengaruh terhadap intensitas pengguna *website* dan kepuasan pengguna *website* berpengaruh terhadap intensitas penggunaan *website* Pemerintah Kota Yogyakarta. Dari beberapa penelitian yang telah ada di dapat disimpulkan bahwa metode penampilan maupun pengiriman informasi masih terkait dengan sebuah aplikasi tertentu, atau tersaji pada sebuah aplikasi. Kelengkapan informasi di *website* akan meningkatkan tingkat kepuasan pengguna, sehingga di harapkan tingkat kunjungan *website* akan meningkat. Dalam sebuah buku dengan judul SQL The Complete Refference Third Edition (Groff, Weinberg, & Oppel, 2010) didefinisikan tentang konsep *trigger* yaitu semua kejadian apa pun yang menyebabkan perubahan dalam isi tabel, pengguna dapat menentukan tindakan terkait yang harus dilakukan oleh DBMS. Tiga peristiwa yang dapat memicu tindakan adalah upaya untuk menyisipkan, menghapus, atau memperbarui baris dari tabel. Tindakan yang dipicu oleh suatu peristiwa ditentukan oleh urutan pernyataan SQL.



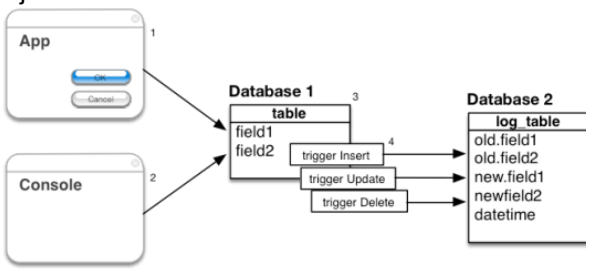
Gambar II.1. Skema *Trigger* (Syafitri, 2019)

Dalam DBMS (*Database Management Sistem*), *trigger* merupakan kumpulan *script* yang berhubungan dengan tabel, view ataupun skema yang dijalankan secara otomatis ketika terdapat *event* yang dijalankan, gambar II.1 menunjukkan Skema *trigger*. (Syafitri, 2019) *Event* tersebut meliputi operasi yang biasa dilakukan dalam mengolah *database*, seperti :

- DML (*Data Manipulation Language*) yang meliputi *DELETE, INSERT* atau *UPDATE*
- DDL (*Data Definition Language*) yang meliputi *CREATE, ALTER* atau *DROP*

- Operasi *Database* lainnya, seperti *SERVERERROR, LOGON, LOGOFF, STARTUP* atau *SHUTDOWN*.

Pada umumnya dalam *relasional database, trigger* dapat ditemui ketika melakukan perintah eksekusi tabel. Hal itu menjadi lebih bermanfaat *trigger* menjadikan penulisan pemrograman yang sederhana dan dapat menjaga informasi agar tetap konsisten dalam *database*. Selain itu, terdapat beberapa fungsi *trigger* pada gambar II.2 yang akan dijelaskan dibawah ini.



Gambar II.2. Fungsi *Trigger*

1. *Integritas Data*

Dengan adanya *trigger*, Anda dapat mempertahankan *integritas tabel* yang terdapat dalam *database*. Sebab, Anda dapat melakukan operasi-operasi yang berkaitan dengan pengolahan *database*, seperti *INSERT, UPDATE* dan *DELETE*

2. Mencegah *Error*

Trigger dalam MySQL dapat mencegah terjadinya *error* dalam pengoperasian data. Jika terjadi *error* dalam pendefinisian *trigger*, *error* tersebut tidak mengganggu *trigger* yang sedang berjalan.

3. Membuat Tugas Kerja menjadi Terjadwal

Ketika *trigger* telah dijalankan, maka Anda dapat menggunakan berbagai bahasa pemrograman tanpa harus bingung bagaimana cara mengaksesnya.

4. Mencegah Proses Transaksi yang Tidak Sah

Dalam praktiknya, biasa *trigger* digunakan untuk melakukan proses transaksi. Anda dapat menyimpan record transaksi tersebut ke tabel lain (*history*) tanpa harus takut jika data tersebut di-*update* atau *delete*. Semua perubahan yang terjadi juga dapat diacak berdasarkan waktu pembuatannya.

Secara ringkas, *trigger* sebagai sebuah fasilitas RDBMS juga memiliki keuntungan dan kerugian, seperti berikut:

1. Kelebihan *Trigger*

- *Trigger* menyediakan cara *alternative* untuk memeriksa integritas.
- Trigger biasa menangkap kesalahan dalam *business logic* pada tingkat *database*.
- *Trigger* menyediakan cara *alternative* untuk menjalankan tugas-tugas yang dijadwalkan
- *Trigger* sangat berguna untuk mengaudit perubahan data dalam tabel *database*

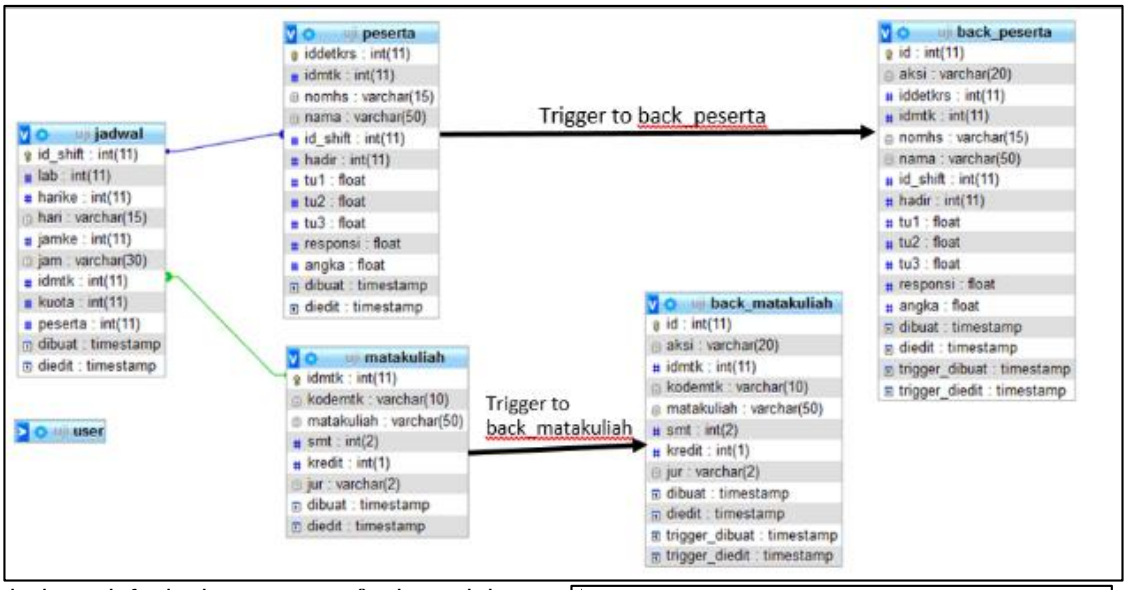
2. Kelemahan *Trigger*

- *Trigger* hanya bisa menyediakan validasi tambahan tapi tidak dapat menggantikan semua validasi
- Beberapa validasi sederhana dapat dilakukan di level aplikasi, sbg Contoh : Kita dapat menvalidasi inputan di sisi klien menggunakan *javascript* atau di sisi *server* dengan menggunakan *script* PHP atau ASP.NET.
- *Trigger* mengeksekusi secara tak terlihat dari klien-aplikasi yang terhubung ke *database server* sehingga sulit untuk mencari tahu apa yang terjadi di level *database*.
- *Trigger* berjalan setiap *update* yang dibuat ke tabel sehingga menambah beban kerja ke *database* dan menyebabkan sistem berjalan lebih lambat.

Adapun manfaat dari penggunaan *trigger* adalah sebagai berikut:

- Meng-generate nilai kolom turunan (*derived column value*)
- Mencegah transaksi yang tidak valid
- Mengerjakan otorisasi keamanan yang kompleks
- Mengerjakan aturan bisnis (*business rule*) yang kompleks
- Menyediakan pencatatan *event* (*event logging*) scr transparant
- Menyediakan audit
- Mengerjakan *referential integrity* ke seluruh *node* dalam sebuah basis data terdistribusi
- Menjaga replikasi tabel secara *syncronous*
- Mengumpulkan statistic dari pengaksesan tabel
- Modifikasi data tabel kerika DML dijalankan pada view
- Mempublikasikan informasi ketika ada *database event user* dan pernyataan SQL untuk suatu aplikasi (yang berlanganan/*subscribe*)

Dimasa sekarang waktu transaksi menjadi salah satu kunci atau bukti digital yang tidak bisa ditinggalkan, sehingga waktu server menjadi sangat penting tetapi pada kenyataannya waktu server kadang akan berbeda dengan waktu lokal karena adanya zona waktu. Konsekuensi dari zona waktu ini adalah terjadinya perbedaan waktu antara waktu pada komputer server dengan waktu yang digenerate oleh PHP. Penggunaan



fungsi date_default_timezone_set() dan ini _set() hanya akan berpengaruh pada script setelah fungsi tersebut dipanggil, meskipun kita letakkan diawal script, pada sistem yang kompleks, bisa jadi terdapat *script* yang tidak terpengaruh. Untuk itu, kita dapat mengubah zona waktu secara global melalui konfigurasi pada php.ini. Seperti pada pembahasan pada bagian awal, buka file php.ini dan cari kata-kata *date.timezone*, kemudian ganti nilainya sesuai dengan yang diinginkan, misal kita ganti dengan Asia/Jakarta. Gambar II.3 menunjukkan pengaturan zona waktu server di *apache*. (Hadi, 2019)

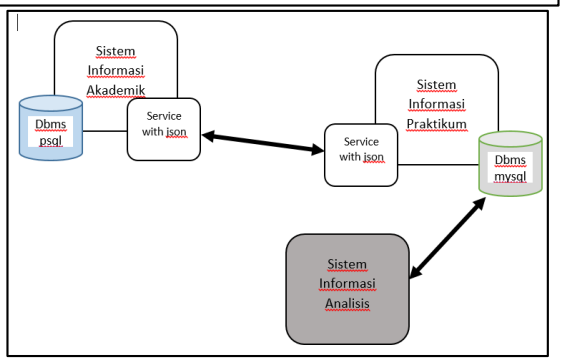
```
[Date]
; Defines the default timezone used by the date functions
; http://php.net/date.timezone
date.timezone = Asia/Jakarta

; http://php.net/date.default-latitude
;date.default_latitude = 31.7667
```

Gambar II.3. Pengaturan Zona Waktu

HASIL DAN PEMBAHASAN

Dalam melakukan penelitian ini, peneliti menggunakan sumber informasi dari Sistem Informasi Akademik dengan RDBMS PsQL yang memuat tentang KRS mahasiswa untuk kemudian diambil melalui service menggunakan JSON dan ditransaksikan ke Sistem Informasi Praktikum dengan RDBMS MySQL, Sistem informasi praktikum digunakan untuk pelayanan praktikum mulai dari pendaftaran praktikum sampai praktikan memperoleh nilai diakhir praktikum. Data dari system informasi praktikum ini yang diterapkan control trigger dan waktu server yang akhirnya bisa dimonitor menggunakan aplikasi system informasi analisis praktikum. Gambar konsep sistem



Gambar konsep Sistem

Adapun perancangan database dari system informasi praktikum memuat trigger dimana table peserta dan matakuliah akan di trigger tiap ada event atau kejadian, yaitu saat terjadinya perintah *INSERT*, *UPDATE* dan *DELETED.* Gambar Skema RDBMS menunjukkan relasi antar table dan juga trigger yang terjadi.

Trigger yang terjadi pada table matakuliah saat terjadinya penambahan data dari service akan direspon oleh trigger dengan status *after insert* sehingga setelah melakukan insert pada table *matakuliah* langsung akan di lakukan insert pada table *back_matakuliah* adalah sebagai berikut:

```
DELIMITER $$
CREATE TRIGGER `tambah_matakuliah` AFTER INSERT ON `matakuliah` FOR EACH ROW insert into back_matakuliah(aksi,idmtk,kodemtk,matakuliah,kredit,smt,jur,dibuat,diedit)
values('tambah',new.idmtk,new.kodemtk,new.matakuliah,new.kredit,new.smt,new.jur,new.dibuat,new.diedit)
$$
DELIMITER ;
```

Begitu juga dengan kejadian saat melakukan perubahan data, maka sebelum dilakukan

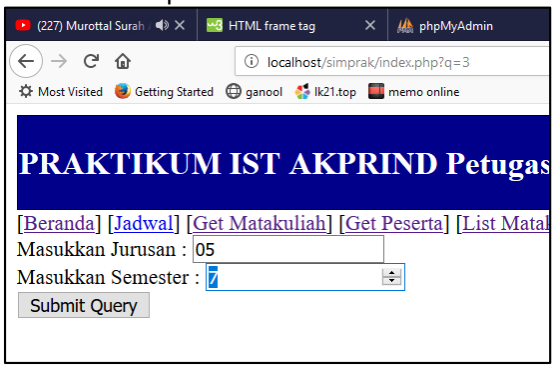
perubahan data asli akan di*insert*kan ke table *back_matakuliah.*

```
DELIMITER $$
CREATE TRIGGER `update_matakuliah` BEFORE UPDATE ON `matakuliah` FOR EACH ROW insert into back_matakuliah(aksi,idmtk,kodemtk,matakuliah,kredit,smt,jur,dibuat,diedit)
values('ubah',old.idmtk,old.kodemtk,old.matakuliah,old.kredit,old.smt,old.jur,old.dibuat,old.diedit)
$$
DELIMITER ;
```

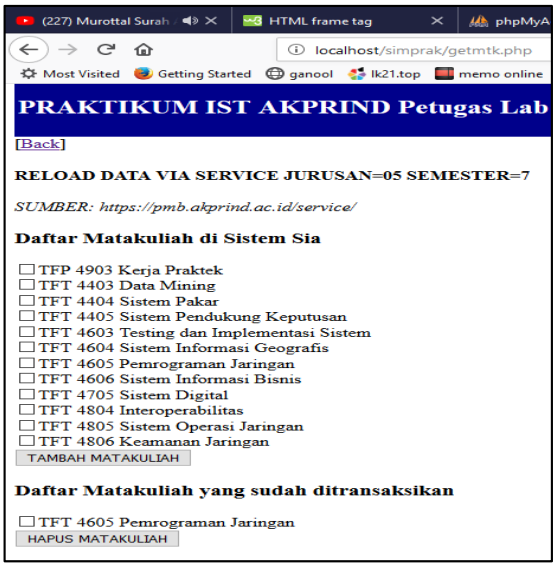
Dan juga pada saat terjadinya penghapusan data, maka sebelum dilakukan penghapusan data akan di*insert*kan ke table *back_matakuliah.*

```
DELIMITER $$
CREATE TRIGGER `hapus_matakuliah` BEFORE DELETE ON `matakuliah` FOR EACH ROW insert into back_matakuliah(aksi,idmtk,kodemtk,matakuliah,kredit,smt,jur,dibuat,diedit)
values('hapus',old.idmtk,old.kodemtk,old.matakuliah,old.kredit,old.smt,old.jur,old.dibuat,old.diedit)
$$
DELIMITER ;
```

Dengan cara yang sama, juga dilakukan terhadap table *peserta* ke table *back_peserta*.
Sistem informasi Praktikum saat akan melakukan pengambilan data Matakuliah maupun data Peserta melalui menu aplikasi tersebut.
Proses mengambil Data Matakuliah.
Untuk proses ini akan ditanyakan jurusan dan semester dari matakuliah tersebut, lalu akan ditampailkan datanya untuk kemudian dipilih melalui menu pilihan. Gambar

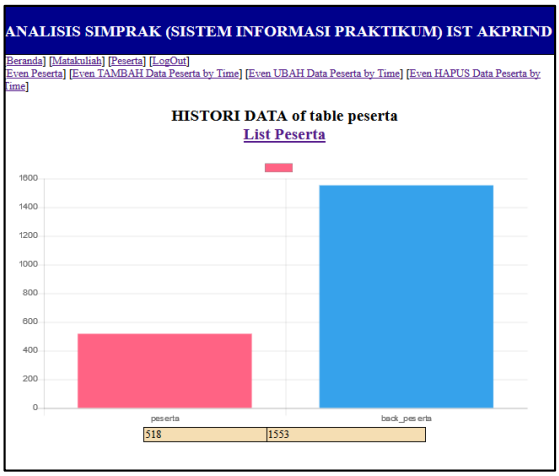


Step 1 Pengambilan Matakuliah



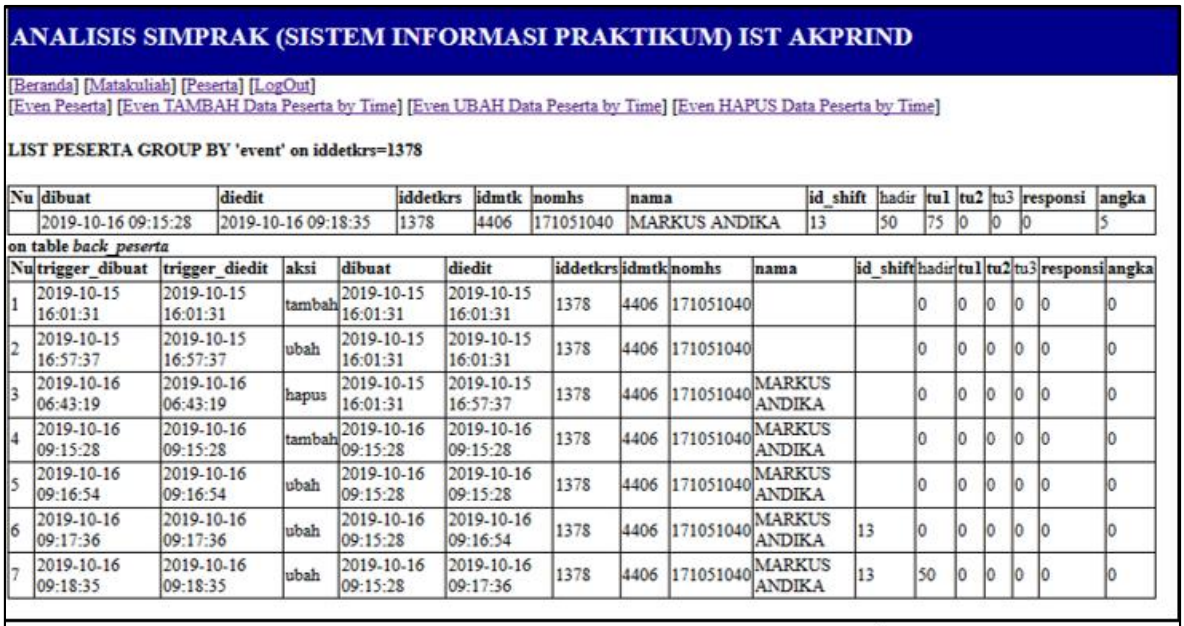
Step 2 Pengambilan matakuliah

Begitu juga dengan pengambilan data peserta, dengan metode yang tidak jauh berbeda. Sedangkan proses penjadwalan dan pemberian nilai juga mengikuti pola yang hampir sama dengan proses diatas.
Pada aplikasi analisis data bisa dilihat semua kejadian yang telah dilakukan di system informasi praktikum.



Tampilan Analisis Histori data tabel Peserta

Gambar menunjukkan tampilan grafik perbedaan antara table peserta (518 record) dengan back_peserta (1553 record). Disini menunjukkan bahwa dari table utama telah terjadi transaksi even baik itu *insert, update dan delete* yang jumlahnya lebih dari 3x table utama.



ANALISIS SIMPRAK (SISTEM INFORMASI PRAKTIKUM) IST AKPRIND

[Beranda] [Matakuliah] [Peserta] [LogOut]
[Even Peserta] [Even TAMBAH Data Peserta by Time] [Even UBAH Data Peserta by Time] [Even HAPUS Data Peserta by Time]

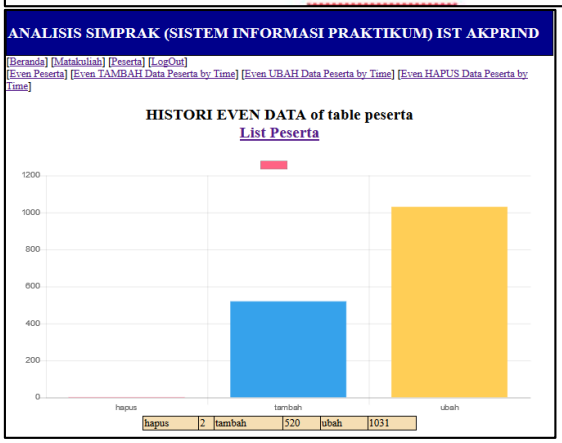
LIST PESERTA GROUP BY 'event' on iddetkrs=1378

Nu	dibuat	diedit	iddetkrs	idmtk	nomhs	nama	id_shift	hadir	tu1	tu2	tu3	responsi	angka
	2019-10-16 09:15:28	2019-10-16 09:18:35	1378	4406	171051040	MARKUS ANDIKA	13	50	75	0	0	0	5

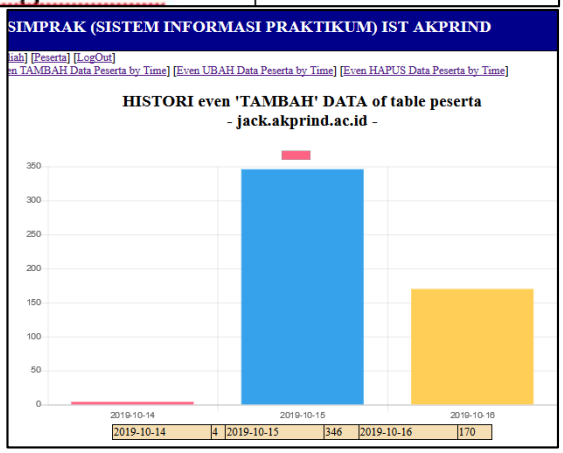
on table *back_peserta*

Nu	trigger_dibuat	trigger_diedit	aksi	dibuat	diedit	iddetkrs	idmtk	nomhs	nama	id_shift	hadir	tu1	tu2	tu3	responsi	angka
1	2019-10-15 16:01:31	2019-10-15 16:01:31	tambah	2019-10-15 16:01:31	2019-10-15 16:01:31	1378	4406	171051040			0	0	0	0	0	0
2	2019-10-15 16:57:37	2019-10-15 16:57:37	ubah	2019-10-15 16:01:31	2019-10-15 16:01:31	1378	4406	171051040			0	0	0	0	0	0
3	2019-10-16 06:43:19	2019-10-16 06:43:19	hapus	2019-10-15 16:01:31	2019-10-15 16:57:37	1378	4406	171051040	MARKUS ANDIKA		0	0	0	0	0	0
4	2019-10-16 09:15:28	2019-10-16 09:15:28	tambah	2019-10-16 09:15:28	2019-10-16 09:15:28	1378	4406	171051040	MARKUS ANDIKA		0	0	0	0	0	0
5	2019-10-16 09:16:54	2019-10-16 09:16:54	ubah	2019-10-16 09:15:28	2019-10-16 09:15:28	1378	4406	171051040	MARKUS ANDIKA		0	0	0	0	0	0
6	2019-10-16 09:17:36	2019-10-16 09:17:36	ubah	2019-10-16 09:15:28	2019-10-16 09:16:54	1378	4406	171051040	MARKUS ANDIKA	13	0	0	0	0	0	0
7	2019-10-16 09:18:35	2019-10-16 09:18:35	ubah	2019-10-16 09:15:28	2019-10-16 09:17:36	1378	4406	171051040	MARKUS ANDIKA	13	50	0	0	0	0	0

Gambar Detail pergerakan data



Gambar Event Peserta



Gambar Perubahan data per tanggal

Dari gambar tersebut terlihat kegiatan yang terjadi pada table peserta. Proses Tambah 520x, proses Ubah 1031x dan Hapus 2x. Dengan informasi tersebut maka harus terlihat data apa saja yang berubah dan seperti apa perubahannya dari waktu ke waktu. Pada gambar berikut diperlihatkan bahwa perubahan update data terhadap table peserta terjadi pada tiga tanggal yaitu 14-Okt (4data), 15-okt (346data) dan 16-Okt (170data). Dengan informasi tersebut jelas terlihat trend dari perubahan data yang terjadi.

KESIMPULAN

Setelah melakukan penelitian ini, peneliti dapat menyimpulkan sebagai berikut:

- Dengan penerapan service antar system melalui API atau JSON, maka komunikasi data antar system/server bisa dilakukan lebih mudah.
- Dengan penerapan Trigger pada untuk mengelola event sebuah table, maka perkembangan informasi bisa dilacak jika terjadi permasalahan inkonsistensi data.
- Membangun Sistem informasi tidak harus BESAR, tetapi cukup dengan modular atau sub-system kecil dan bahkan bisa LOCALHOST jika managemen server dan database dikelola dengan baik.

Saran

- Belum semua unit menerapkan dan mau interkoneksi antar informasi

- Masih sulit untuk mengubah wacana bahwa data bisa disharing untuk kebutuhan yang lebih luas
- Penggunaan enkripsi dan hak akses data belum dimasukkan dalam penelitian ini

DAFTAR PUSTAKA

Dedianto. (2013). Sistem Trigger Database Pada SIAKAD Informatika. *JUSTIN (Jurnal Sistem dan Teknologi Informasi)* Vol 1 No 1 2013, 17-20.

Groff, J. R., Weinberg, P. N., & Oppel, A. J. (2010). *SQL The Complete Refference - Third Edition. United States*: The McGraw-Hill Companies.

Hadi, A. P. (2019, 05 10). *Memahami Zona Waktu (Timezone) dan Selisih Waktu Pada PHP. Retrieved from Jago Web Dev*: https://jagowebdev.com/timezone-selisih-waktu-pada-php/

Haryani, P. (2016). Evaluasi Kualitas Layanan E-Government Pemerintah Kota Yogyakarta Dengan Metode E-GovQual *Modifikasi. Simposium Nasional Ke-15 RAPI* 2016 (pp. 379-386). Surakarta: Fakultas Teknik Universitas Muhammadiyah Surakarta.

S, A. S., Suwanto, R., & Triyono, J. (2017). Analisis Keamanan Serangan SQL Injection berdasarkan Metode Koneksi Database. *Jurnal Script* Jilid 4 Terbitan ke1.

Syafitri, I. (2019, 05 14). *Mengenal Pengertian Trigger dalam Database Beserta Fungsi dan Contohnya. Retrieved from Mengenal Pengertian Trigger Beserta Fungsi dan Contohnya dalam Database* : https://www.nesabamedia.com/pengertian-trigger/

Triyono, J. (2015). Sistem Informasi Agroteknologi Berbasis Web Dan Jejaring Sosial Twitter. *Seminar Nasional IENACO*, 205-212.