JTH: Journal of Technology and Health *ISSN 3025-1168*
Vol. 3, No. 2, 2025, pp. 120 ~ 132
Open Acces: *https://doi.org/10.61677/jth.v3i2.365*

DATA SECURITY STRATEGY BASED ON STORED PROGRAMS: A CASE STUDY OF DSS FOR TEACHING ASSISTANT SELECTION

Prahenusa Wahyu Ciptadi[1], Tri Hastono[2], Gema Kharismajati[3], Firdiyan Syah[4], Nayaka Anung Fahriza[5]

[1,2,3,4]Informatika, Universitas PGRI Yogyakarta, Indonesia
[5]TKIT Al Farabi, Indonesia
nusa@upy.ac.id, trihastono.13@gmail.com, gemakharismajati@upy.ac.id, nayaka.anung@gmail.com
**Corresponding author*

Received February 5, 2025; Revised October 12, 2025; Accepted October 26, 2025; Published October 30, 2025

ABSTRACT

Data security is increasingly crucial in managing information systems, especially in academic environments handling sensitive data. The reliance on digital systems demands protection of data integrity, confidentiality, and availability. Educational institutions must secure student, faculty, and administrative data to prevent unauthorized access and manipulation. One system requiring high security is the Decision Support System (DSS) for Teaching Assistant selection. DSS helps determine candidates based on GPA, experience, and skills. However, it is vulnerable to unauthorized data changes, access abuse, SQL Injection, and cyber threats that may affect decision accuracy. Stored Programs in databases provide an effective solution for enhancing data security. With Stored Procedures, Functions, Triggers, and Events, data management becomes more secure and structured. These features enable stricter access control, database operation automation, and direct validation to reduce errors and misuse. In the DSS case study, Stored Programs ensure that only authorized users can access and modify data. Authentication and authorization mechanisms restrict access based on user roles. This study analyzes security risks, system architecture design, Stored Program implementation, and its effectiveness in maintaining data integrity. Research results confirm that the developed Stored Programs meet expectations. The CRUD feature operates optimally with a processing time of 0.0059 seconds.

Keywords: *Data security strategy; stored Programs; DSS teaching assistant selection*

ABSTRAK

Keamanan data semakin krusial dalam pengelolaan sistem informasi, terutama di lingkungan akademik yang menangani informasi sensitif. Ketergantungan pada sistem digital menuntut perlindungan integritas, kerahasiaan, dan ketersediaan data. Institusi pendidikan harus mengelola data mahasiswa, dosen, dan administrasi dengan sistem aman untuk mencegah akses tidak sah dan manipulasi data. Salah satu sistem yang membutuhkan keamanan tinggi adalah Sistem Pendukung Keputusan (SPK) untuk pemilihan Asisten Dosen. SPK membantu menentukan kandidat berdasarkan IPK, pengalaman, dan keterampilan. Namun, sistem ini rentan terhadap ancaman seperti perubahan data tidak sah, penyalahgunaan akses, SQL Injection, dan serangan siber yang dapat mempengaruhi keputusan. Stored Programs dalam database menjadi solusi efektif untuk meningkatkan keamanan data. Dengan Stored Procedures, Functions, Triggers, dan Events, pengelolaan data menjadi lebih aman dan terstruktur. Fitur ini memungkinkan kontrol akses yang lebih ketat, otomatisasi operasi database, serta validasi langsung untuk mengurangi risiko kesalahan dan penyalahgunaan. Dalam studi kasus SPK Asisten Dosen, Stored Programs memastikan hanya pengguna berwenang yang dapat mengakses dan memodifikasi data.

Mekanisme autentikasi dan otorisasi membatasi akses sesuai peran pengguna. Penelitian ini menganalisis risiko keamanan, desain arsitektur sistem, implementasi Stored Programs, dan evaluasi efektivitasnya dalam menjaga integritas data. Hasil penelitian menunjukkan bahwa Stored Programs yang dikembangkan telah memenuhi ekspektasi. Fitur CRUD berjalan optimal dengan waktu proses 0,0059 detik.

Keywords: *Strategi keamanan data; stored programs; SPK penentuan asisten dosen*

PENDAHULUAN

Keamanan data merupakan aspek yang semakin krusial dalam pengelolaan sistem informasi, terutama dalam lingkungan akademik yang sering menangani informasi sensitif (Putri et al., 2024; Saputra, 2023; Satria et al., 2024). Seiring dengan meningkatnya ketergantungan pada sistem digital, perlindungan terhadap integritas, kerahasiaan, dan ketersediaan data menjadi tantangan utama.

Dalam konteks institusi pendidikan, data mahasiswa, dosen, dan administrasi harus dikelola dengan sistem yang aman untuk mencegah akses yang tidak sah serta potensi manipulasi data yang dapat merugikan berbagai pihak (Yolanda Sari Ks et al., 2022). Salah satu implementasi sistem berbasis data yang membutuhkan keamanan tinggi adalah Sistem Pendukung Keputusan (SPK) untuk penentuan Asisten Dosen. SPK digunakan untuk membantu menentukan kandidat yang paling sesuai berdasarkan kriteria tertentu seperti IPK, pengalaman, dan keterampilan teknis. Namun, dalam implementasinya, sistem ini rentan terhadap ancaman seperti perubahan data yang tidak sah, penyalahgunaan akses, SQL Injection, serta serangan siber yang dapat mengakibatkan keputusan yang tidak valid atau manipulatif (Aulia et al., 2023; Dm et al., n.d.; Septiadi & Tripustikasari, 2022; Yolanda Sari Ks et al., 2022).

Sejumlah penelitian sebelumnya telah membahas penggunaan Stored Procedures atau teknik pengamanan database untuk meningkatkan efisiensi dan kecepatan akses data (Rohmana et al., 2019; Aulia et al., 2023). Namun, masih terbatas studi yang secara spesifik mengintegrasikan seluruh komponen Stored Programs termasuk Functions, Triggers, dan Events dalam konteks sistem pendukung keputusan akademik yang membutuhkan tingkat integritas data tinggi, seperti penentuan asisten dosen. Selain itu, kebanyakan studi menekankan pada aspek performa sistem (kecepatan query), bukan pada strategi pencegahan manipulasi data secara sistemik di lingkungan basis data. Oleh karena itu, riset ini mengisi kekosongan tersebut dengan fokus pada desain dan penerapan strategi keamanan menyeluruh menggunakan Stored Programs dalam sistem informasi berbasis SPK.

Untuk mengatasi tantangan tersebut, *Stored Programs* dalam database dapat menjadi solusi yang efektif dalam meningkatkan keamanan data (Fathulloh & Adauwiyah, 2021; Hartono, 2021). *Stored programs* memungkinkan eksekusi serangkaian perintah SQL dalam satu unit yang dapat digunakan kembali tanpa perlu menuliskan ulang kode secara langsung di aplikasi. *Stored Programs*, yang mencakup *Stored Procedures*, *Functions*, *Triggers*, dan *Events*, memungkinkan pengelolaan data dengan cara yang lebih aman dan terstruktur. Dengan menggunakan fitur ini, akses ke

data dapat dikontrol dengan lebih ketat, operasi database dapat diotomatisasi untuk meningkatkan efisiensi, dan validasi dapat diterapkan secara langsung di tingkat database untuk mengurangi risiko kesalahan atau penyalahgunaan data (Aulia et al., 2023; Rohmana et al., 2019; Septiadi & Tripustikasari, 2022).

Dalam studi kasus penentuan Asisten Dosen, penggunaan *Stored Programs* dapat membantu memastikan bahwa hanya pengguna yang berwenang yang dapat mengakses dan memodifikasi data (Hartono, 2021). Sebagai contoh, mekanisme autentikasi dan otorisasi berbasis *Stored Programs* dapat diterapkan untuk membatasi hak akses pengguna berdasarkan peran mereka dalam system (Triyono, 2023). Hal ini memastikan bahwa hanya administrator yang dapat mengubah kriteria seleksi, sementara dosen hanya dapat menilai mahasiswa tanpa dapat mengedit data kandidat secara langsung (Fathulloh & Adauwiyah, 2021; Firdaus, 2024; Hartono, 2021).

Penelitian ini menawarkan pendekatan baru dengan mengimplementasikan strategi keamanan data yang holistik berbasis Stored Programs di lingkungan akademik, di mana kerahasiaan dan keandalan data sangat krusial. Kebaruan utama terletak pada integrasi fungsi validasi, kontrol akses, dan pengendalian manipulasi data langsung pada tingkat basis data, bukan hanya pada lapisan aplikasi. Hal ini memberikan lapisan pertahanan tambahan terhadap potensi serangan siber dan penyalahgunaan akses internal. Tujuan utama penelitian ini adalah untuk merancang, membangun, dan mengevaluasi efektivitas Stored Programs dalam menjaga keamanan data SPK pemilihan asisten dosen di Universitas PGRI Yogyakarta. Dengan demikian, hasil dari penelitian ini diharapkan dapat menjadi rujukan dalam pengembangan sistem informasi akademik yang aman dan efisien.

Melalui penelitian ini, akan dibahas bagaimana strategi keamanan data berbasis *Stored Programs* dapat diterapkan dalam SPK penentuan Asisten Dosen. Fokus dari penelitian adalah analisis risiko keamanan, desain arsitektur sistem yang lebih aman, implementasi *Stored Programs* dalam database, serta evaluasi efektivitas metode ini dalam menjaga integritas dan kerahasiaan data. Dengan pendekatan ini, diharapkan sistem dapat berfungsi dengan lebih andal dan aman tanpa mengorbankan efisiensi dalam pengambilan keputusan

METODE PENELITIAN

Pada artikel dengan "Strategi Keamanan Data Berbasis *Stored Programs*: Studi Kasus SPK Penentuan Asisten Dosen" ini membahas mengenai perancangan *Stored Programs* pada proses CRUD (*Create*, *Read*, *Update*, dan *Delete*) data pada tabel yang telah dibuat dalam SPK penentuan asisten dosen. Untuk metode yang dipilih pada perancangan adalah *Waterfall*. Untuk tahapan dari penelitian adalah : analisis kebutuhan, desain system, implementasi, pengujian, dan penerapan.

Pada tahap analisis kebutuhan dilakukan proses pengumpulan dan pendokumentasian kebutuhan pengguna dan system. Untuk data kebutuhan pengguna dilakukan dengan proses wawancara dengan calon dari pengguna. Untuk wawancara dilakukan oleh peneliti dan ketua program studi Informatika UPY. Hasil dari proses

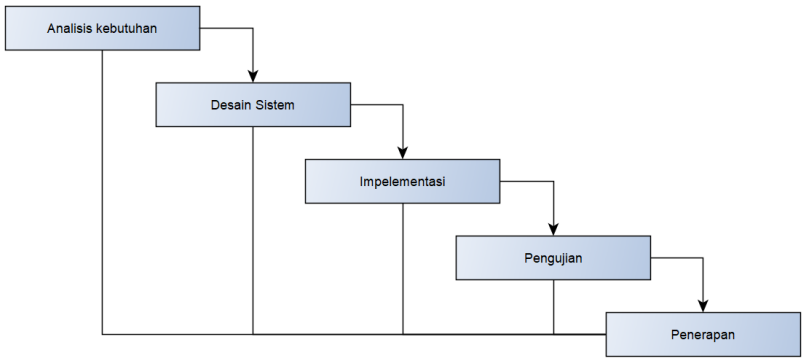
wawancara yang dilakukan dihasilkan spesifikasi *Stored Programs* yang diperlukan pada SPK penentuan asisten dosen di UPY.

Desain system adalah tahap lanjutan penelitian setelah tahap analisis kebutuhan pengguna dan system. Pada tahap ini merupakan penuangan hasil proses analisis kebutuhan dalam bentuk Gambaran kasar mengenai *Stored Programs* yang akan dibuat. Gambaran kasar yang dimaksud adalah arsitektur system, desain basisdata, diagram alur data (DFD), dan Entity Relationship Diagram (ERD).

Tahap implemantasi adalah tahap dilakukan penuangan hasil dari tahap sebelumnya, yaitu tahap desain system. pada tahap ini dilakukan perancangan dan pembuatan kode program *Stored Programs* untuk basisdata SPK penentuan asisten dosen. Untuk jenis *Stored Programs* yang dibuat pada tahap ini adalah procedure, function, cursor, struktur control dan penanganan eror.

Tahap pengujian adalah sebuah tahap yang ada penelitian dimana dilakukan pengujian terhadap *Stored Programs* yang dibuat. Pada tahap ini dipastikan ada tidaknya bug dari kode program Stored Programs yang dirancang, baik berupa kesalahan penulisan kode program *Stored Programs*, kesalahan logika atau kesalahan yang lainnya. Ketika sudah melewati tahap pengujian, hasilnya dapat digunakan untuk tahap terakhir yaitu tahap penerapan.

Pada tahap penerapan adalah sebuah tahap dalam penelitian dimana dilakukan proses instalasi dari *Stored Programs* yang sudah lolos uji pada tahap pengujian. Dan *Stored Programs* yang dihasilkan adalah *Stored Programs* yang sudah siap diintegrasikan dengan Bahasa pemrograman. Untuk alur penelitian dapat disajikan pada gambar 1 dibawah ini.



Gambar 1. Alur Penelitian

RESULTS AND DISCUSSION

a. Data Penelitian

Seperti yang telah dijelaskan diawal, pada analisis kebutuhan telah dilakukan dengan salah satu ketua program studi di Universitas PGRI Yogyakarta. Adapun narasumber yang telah dipilih Ketua Program Studi Informatika UPY yaitu Ibu Puji Handayani Putri, M.Kom. Salah satu hasil dari wawancara yang telah dilakukan adalah data primer dari penelitian dan algoritma menggunakan Stored Programs. Untuk data pendukung penelitian adalah data kriteria asisten dosen. Tabel 1 dibawah ini adalah tabel

data kriteria asisten dosen yang ada di program studi Informatika UPY

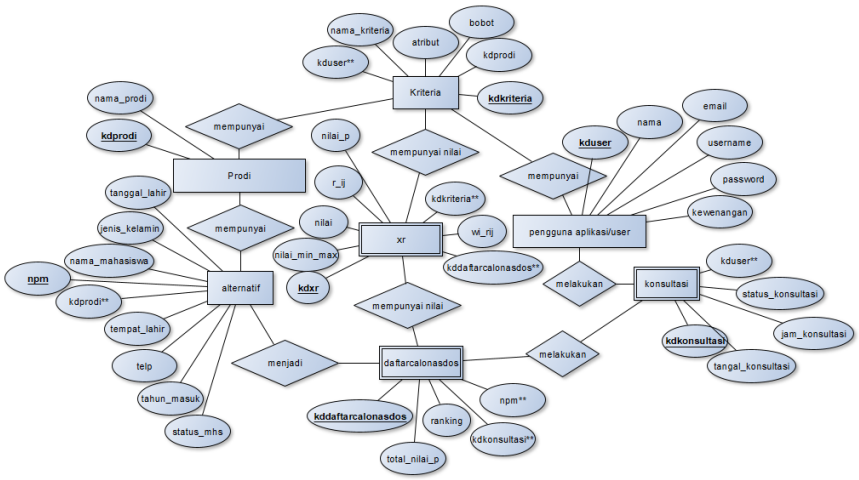
Tabel 1. Data Kriteria SPK Penentuan Asisten Dosen

No	Kode Krieria	Nama Kriteria	Exist
1	CRT1	Ipk	Benefit
2	CRT2	Keterampilan Komunikasi	Benefit
3	CRT3	Keterampilan Penyampaikan Materi Secara Efektif	Benefit
4	CRT4	Kemampuan Berkolaborasi	Benefit
5	CRT5	Pengetahuan Teknologi	Benefit
6	CRT6	Kemampuan Bekerja dalam Tim	Benefit
7	CRT7	Keterampilan Memberikan Bimbingan Akademis	Benefit
8	CRT8	Kemampuan Mengelola Waktu	Benefit
9	CRT9	Keaktifan dalam Organisasi	Cost
10	CRT10	Jarak Domisili dengan Kampus	Cost

Dengan berbekal data dari proses wawancara dapat dibuat basisdata dan tabel-tabelnya. Namun sebelum dibuat basisdatannya, terlebih dahulu dilakukan pembuatan *Entity Relationship Diagram*.

b. Entity Relationship Diagram (ERD)

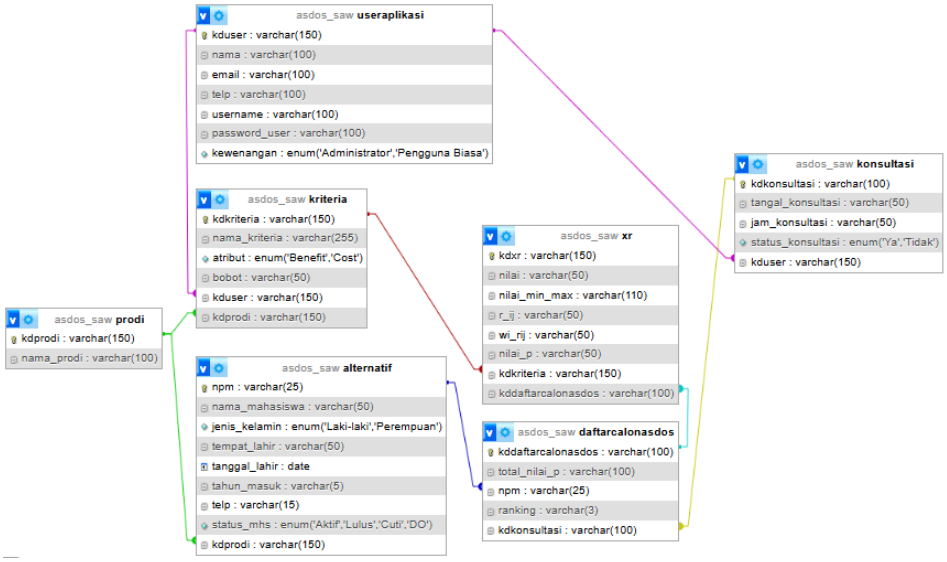
Pada penelitian ini terdapat beberapa entitas kuat. Untuk entitas kuat pada penelitian ini adalah entitas kriteria, entitas alternatif, entitas prodi dan entitas pengguna aplikasi/user. Dan untuk entitas lemah pada penelitian ini adalah entitas konsultasi, entitas daftarcalonasdos, dan entitas xr. Untuk gambar dari ERD penelitian dapat dilihat pada gambar 2.



Gambar 2. ERD Penelitian

c. Desain basisdata SPK

Setelah ERD penelitian sudah jadi, maka dibuatlah desain tabel-tabel basisdata beserta relasi antar tabelnya. Dari semua entitas yang ada pada ERD baik yang kuat maupun lemah nantinya akan turun mejadi tabel-tabel basisdata penelitian. Untuk tabel-tabel basisdata beserta relasinya dapat dilihat pada gambar 3.



Gambar 3. Desain Basisdata SPK Pemilihan Asisten Dosen

Berikut ini adalah penjelasan dari stiap tabel yang ada dalam basisdata Sistem Pendukung Keputusan pemilihan asisten dosen UPY.

- Tabel prodi
  Tabel prodi adalah tabel yang difungsikan untuk menyimpan data program studi. Pada tabel ini hanya memiliki 2 kolom, yaitu : kdprodi dan nama_prodi.
- Tabel alternatif
  Tabel alternatif adalah tabel yang dibuat untuk menampung data mahasiswa yang memungkinkan sekali masuk sebagai bursa asdos ditahun yang bersangkutan.
- Tabel kriteria
  Tabel kriteria adalah tabel yang difungsikan untuk menyimpan data kriteria dari system pendukung Keputusan pemilihan asisten dosen.
- Tabel useraplikasi
  Tabel useraplikasi adalah tabel yang difungsikan untuk menyimpan data pengguna aplikasi, baik dosen, administrator prodi, maupun dosen.
- Tabel konsultasi
  Tabel konsultasi adalah tabel yang digunakan untuk menyimpan data konsultasi pengguna pada system pendukung Keputusan pemilihan asisten dosen.
- Tabel daftarcalonasdos
  Tabel daftarcalonasdos adalah tabel yang dibuat untuk menyimpan data calon asisten dosen yang dipilih oleh pengguna atau administrator.
- Tabel xr
  Tabel xr adalah tabel yang difungsikan sebagai matrik nilai dari setiap calon asisten dosen berdasarkan kriteria asisten dosen.

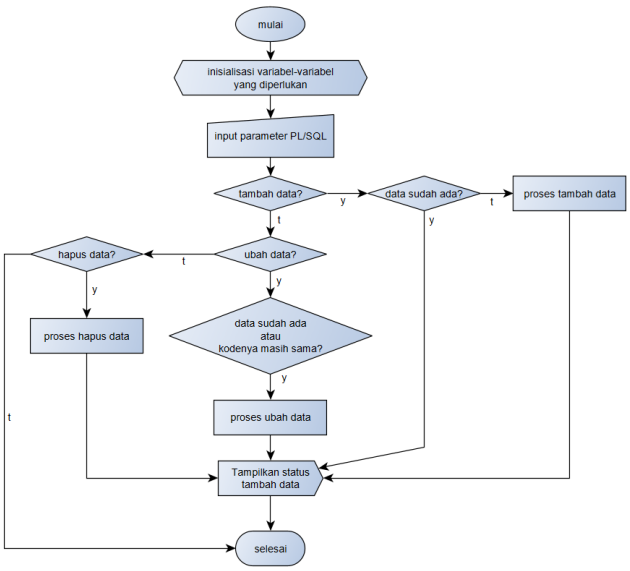
Setelah tabel dalam basisdata system pendukung Keputusan pemilihan calon asisten dosen terbuat, selanjutnya dibuat *Stored Programs*-nya

d. *Stored Programs* CRUD SPK Penentuan Asisten Dosen

Penelitian ini berfokus pada strategi keamanan data input data ditabel system pendukung Keputusan pemilihan asisten dosen menggunakan *Stored Programs*. Keamanan yang dimaksud tersebut karena data yang diproses bukan sintaks queri biasa. Seperti yang kita ketahui Bersama, gaya *programming* setiap *programmer* berbeda satu dengan yang lainnya dalam perancangan Stored Programs (Fathulloh & Adauwiyah, 2021; Hartono, 2021).

Sehingga ketika seseorang yang tidak bertanggung jawab akan memanipulasi data yang berada dibasisdata akan kesulitan, karena harus mengartikan *Stored Programs* yang dibuat *programmer* tersebut, setelah itu baru bisa melakukan aksi kejahatannya. Kode *Stored Programs* yang dibuat bukan berada pada aplikasi yang dirancang, tapi berada disisi server, pelaku kejahatan tersebut juga harus membobol *server* dimana basisdata tersebut berada baru melakukan modifikasi data (Fathulloh & Adauwiyah, 2021; Hartono, 2021).

Disini akan dijelaskan mengenai penerapan *Stored Programs* operasi CRUD pada tabel dalam basisdata system pendukung Keputusan pemilihan asisten dosen, namun karna proses CRUD pada setiap tabel hampir sama untuk semua proses-prosesnya, maka disini hanya akan dijelaskan satu *Stored Programs* untuk mewakili *Stored Programs* ditabel yang lain. Untuk Gambaran mengenai proses pada input data setiap tabel dibasisdata menggunakan *Stored Programs*, dapat dilihat padat flowchart dibawah ini.



Gambar 4. *Flowchart Stored Programs* CRUD Tabel Basisdata SPK Penentuan Asisten Dosen

Dari flowchart yang ditunjukan pada gambar 4, selanjutnya dibuatlah *Stored Programs* untuk input data ke tabel basisdata. Untuk salah satu *Stored Programs* yang akan dibedah contohnya adalah *Stored Programs* fkriteria. Untuk kode *Stored Programs* fkriteria ditunjukan pada gambar 5 dibawah ini.

```
202 drop function if exists fkriteria;
203 DELIMITER $$
204 create function fkriteria(proses varchar(1),nmkrit varchar(100),atrbt varchar(100),bbt varchar(5),adm varchar(230),prd varchar(230),kd
    varchar(230)) returns varchar(15)
205 begin
206     set @a=(SELECT concat('CRT',seq) FROM seq_1_to_1000 where seq not in(SELECT DISTINCT(RIGHT(kdkriteria,(length(kdkriteria )-3)))*1 FROM
        kriteria) limit 0,1);
207     set @adarecord=(SELECT count(*)from kriteria k,prodi p where k.kdprodi=p.kdprodi and k.nama_kriteria=nmkrit and p.nama_prodi=prd);
208     set @kdprdi=(SELECT distinct(kdprodi)from prodi where nama_prodi=prd);
209     set @kdkrit=(SELECT distinct(k.kdkriteria)from kriteria k,prodi p where k.kdprodi=p.kdprodi and k.nama_kriteria=nmkrit and
        p.nama_prodi=prd);
210     if proses='t' then
211     begin
212         if @adarecord=0 then
213         begin
214             insert into kriteria values(@a,nmkrit,atrbt,bbt,adm,@kdprdi);
215         end;
216         end if;
217         return concat(proses,row_count());
218     end;
219     elseif proses='e' then
220     begin
221         if (@adarecord=0)or(@kdkrit=kd) then
222         begin
223             update kriteria set nama_kriteria=nmkrit,atribut=atrbt,bobot=bbt,kdprodi=@kdprdi where kdkriteria=kd;
224         end;
225         end if;
226         return concat(proses,row_count());
227     end;
228     elseif proses='h' then
229     begin
230         delete from kriteria where kdkriteria=kd;
231         return concat(proses,row_count());
232     end;
233     END IF;
234 end $$
235 DELIMITER ;
```

Gambar 5. Kode program *Stored Programs* fkriteria

fkriteria adalah salah satu *Stored Programs* basisdata berjenis *function* yang dibuat untuk proses CRUD ditabel kriteria Sistem Pendukung Keputusan Pemilihan Asisten Dosen yang dibangun. *Function fkriteria* akan memberikan nilai kembalian yang dapat dimanfaatkan lebih. Salah satu contoh pemanfaatan dari hasil eksekusinya adalah sebagai status keberhasilan atau kegagalan proses CRUD ditabel.

Sebelum menuju pada kode utama dari *function fkriteria*, terdapat kode 2 baris kode yang mengawali kode *function fkriteria* (baris 202 dan 203 dari gambar 5). Untuk baris 202 adalah sintaks untuk menghapus *function fkriteria* jika *function fkriteria* sudah ada sebelum dibuat yang baru. Hal ini untuk mencegah eror yang mungkin muncul, ketika kita akan menulis ulang kode *function fkriteria* karena ada perubahan pada kode *function fkriteria*.vKata kunci "delimiter" pada baris 203 difungsikan untuk pemisah sintaks *query* satu dengan yang lainnya. Pada baris tersebut dijelaskan untuk pemisah sintaks yang biasanya adalah titik koma (";") dirubah menjadi "$$". Hal ini menguntungkan terutama dilakukan penulisan ulang *function fkriteria* karena ada perubahan kode atau yang lainnya.

Secara umum struktur dari *function Stored Programs fkriteria* ada 2 bagian (nama *function Stored Programs* dan bodi kode *function Stored Programs*). Pada bagian nama terdapat sintaks untuk membuat *function fkriteria* yang diikuti parameter input dengan nilai kembalian yang bertipe karakter. Parameter yang dipilih pada *function Stored Programs fkriteria* adalah parameter inputan dan berjumlah 7 parameter (proses, nmkrit, atrbt, bbt, adm, prd, kd). Kode program *function fkriteria* dan ketujuh parameter input dapat dilihat pada baris 204 dari gambar 5 diatas. Deskripsi 7 parameter inputan pada function Stored Programs fkriteria disajikan pada tabel 2.

Tabel 2. Deskripsi 7 parameter inputan pada Function Stored Programs

No	Nama Parameter input	Keterangan
1	proses	Parameter input yang digunakan untuk menerima inputan jenis proses CRUD apa yang dininputkan oleh pengguna.
2	nmkrit	Parameter input yang digunakan untuk menerima inputan nama kriteria yang dininputkan oleh pengguna.
3	atrbt	Parameter input yang digunakan untuk menerima inputan jenis atribut ("*Benefit*" dan "*Cost*") untuk kriteria yang dininputkan oleh pengguna
4	bbt	Parameter input yang digunakan untuk menerima inputan bobot dari kriteria yang diinputkan oleh pengguna.
5	adm	Parameter input yang digunakan untuk menerima user/administrator siapa yang menginputkan kriteria.
6	prd	Parameter input yang digunakan untuk menerima inputan nama prodi untuk kriteria yang diinputkan oleh pengguna
7	kd	Parameter input yang digunakan untuk menerima inputan kode dari kriteria.

Untuk bagian bodi pada *function fkriteria* diawali kata kunci "*begin*" dan diakhiri dengan kata kunci "*end*". Proses yang pertama kali dilakukan pada *function fkriteria* Ketika dijalankan adalah dilakukannya inisialisasi variabel-variabel yang diperlukan. Dan kode program untuk proses pertama kali *function fkriteria* ditunjukan pada baris 206-209 dari gambar 5. Untuk variable yang diberi nilai awal atau inisialisasi adalah variabel *@a*, variabel *@adarecord*, variabel *@kdprdi*, dan variabel *@kdkrit*.

Variable *@a* dibuat untuk menyimpan hasil *query* yang mencari data angka yang belum ada pada tabel kriteria. Jika dilihat secara seksama baris 206 dari gambar 5 mempunyai arti bahwa dilakukan pemotongan karakter dari karakter sebelah kanan setiap baris data kolom kdkritera. Untuk jumlah karakter yang dipotong adalah panjang karakter setiap baris data kolom kdkriteria yang dikurangi dengan 3 karakter (jumlah karakter untuk string "*CRT*"). Ketika sudah didapatkan hasil pengurangannya lalu dikalikan angka 1 agar hasil pemotongan karakter tersebut berubah tipe datanya menjadi integer. Setelah berubah menjadi data integer baru dilakukan pencarian sembarang angka yang belum masuk tidak sama dengan data hasil pemotongan tersebut. Untuk proses akhirnya dilakukan penggabungan (fungsi tercadang *concat*) dengan string "CRT" sebagai calon kode kriteria yang baru.Variable *@adarecord* digunakan untuk menyimpan hasil *query* yang pencarian banyaknya data ditabel kriteria dengan kondisi kolom tabel nama_kriteria adalah isi dari parameter "*nmkrit*" dan kolom tabel prodi adalah isi dari parameter input "*prd*". Variable *@kdprdi* digunakan untuk menyimpan hasil *query* pencarian data kode prodi dari tabel prodi dengan kondisi nama prodi datanya adalah isi dari parameter input "prd". Variable *@kdkrit* dibuat untuk menyimpan hasil query pencarian kode kriteria dengan kondisi nama kriteria adalah isi dari parameter input "*nmkrit*" dan prodi kriteria berisi data dari parameter "prd". Ketiga variable tersebut digunakan untuk proses CRUD ditabel kriteria.

Setelah dilakukan inisialisasi 3 variabel yang dibutuhkan pada proses CRUD, salanjutnya dilakukan proses pengecekan isi dari parameter input proses (baris kode 210-233 dari gambar 5). Jika isi dari parameter input proses adalah karakter "t", maka dilakukan proses pengecekan isi variable *@adarecord*. Jika isi variable *@adarecord* adalah 0, maka dilakukan proses tambah data baru ditabel kriteria (baris 214 dari gambar 5). Tapi, jika variable *@adarecord* bukan 0, maka tidak ada proses yang dilakukan pada tabel kriteria.

Selanjutnya dilakukan proses kombinasi antara isi dari parameter input proses dengan nilai hasil proses fungsi tercadang dari basidata "*row_count()*" (baris 217 gambar 5). Untuk nilai balik fungsi tercadang "*row_count()*" dapat dilihat pada tabel 3.

Tabel 3. Nilai Balik Fungsi *row_count()*(*Information Functions*, n.d.)

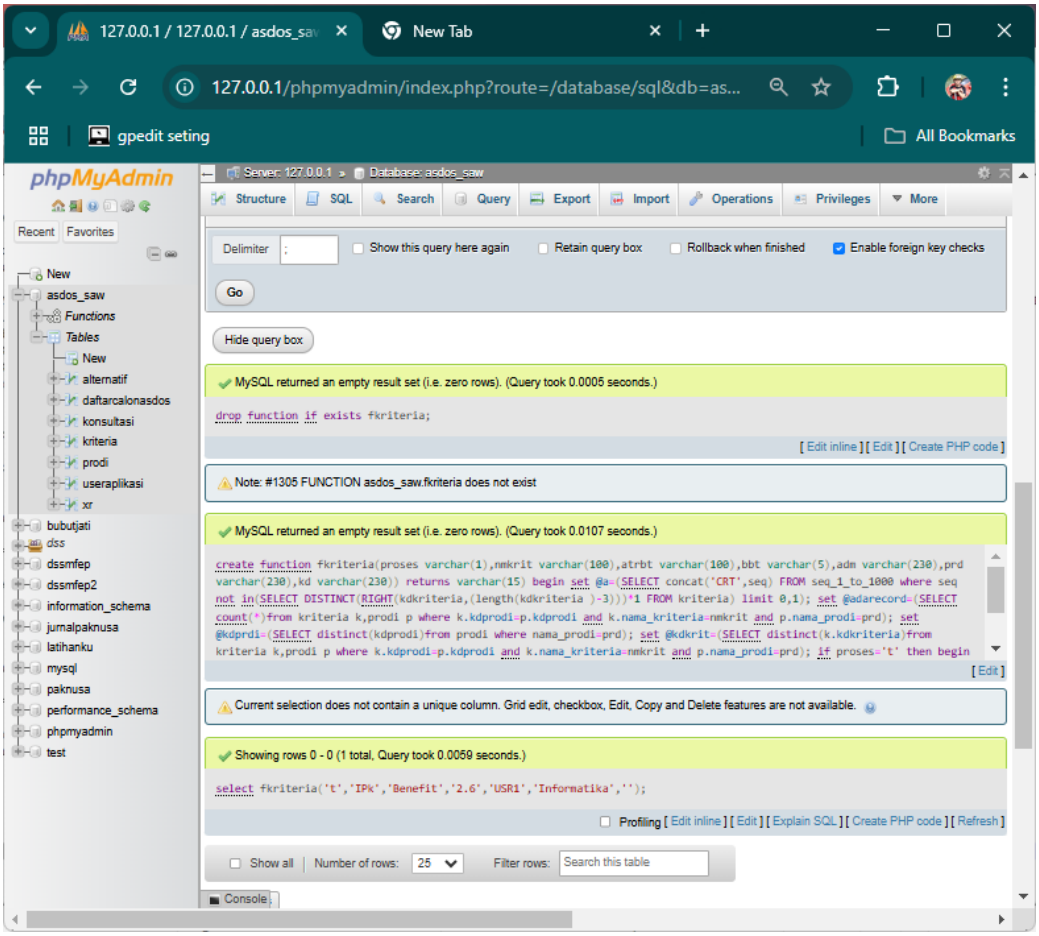
No	Proses	Nilai balik fungsi row_count
1	DML (select, delete, update, insert)	– Nilai fungsi "row_count()" pada proses insert, update, dan delete adalah jumlah baris yang terdampak dari query dan jika tidak ada yang terdampak akan bernilai 0. – Untuk select umumnya bernilai -1.
2	DDL (create table, drop table)	Bernilai 0

Jika melihat data dari tabel 3, maka bisa dipastikan proses tambah data kriteria berhasil jika nilai yang dikembalikan "*row_count()*" bernilai selain 0 dan parameter input proses berisi karakter "t" atau "e" atau "h". Hasil kombinasi inilah yang dikembalikan ketika *function fkriteria* dijalankan. Dan hasil kombinasi ini, bisa diolah dan dimanfaatkan lebih lagi, misalnya sebagai informasi yang ditampilkan pada *dialog box* aplikasi.

Dan Jika isi dari parameter input proses adalah karakter "*e*", maka dilakukan proses pengecekan isi variable *@adarecord* dan *@kdkrit* (baris 219-227 dari gambar 5). Jika isi variable *@adarecord* adalah 0 atau isi dari variable *@kdkrit* sama dengan isi dari parameter input *kd*, maka dilakukan proses ubah data ditabel kriteria berdasarkan kriteria yang dipilih. Dan jika sebaliknya, maka tidak aksi yang dikerjakan. Selanjutnya mengembalikan nilai hasil eksekui *function fkriteria*. Dan Jika isi dari parameter input proses adalah karakter "*h*", maka dilakukan proses hapus data ditabel kriteria berdasarkan kriteria yang dipilih. Dan jika sebaliknya, maka tidak aksi yang dikerjakan. Selanjutnya mengembalikan nilai hasil eksekui *function fkriteria* sebagai informasi untuk pengguna.

e. Pengujian stored program

Untuk pengujian *stored program*, akan dilakukan salah satu proses CRUD, yaitu tambah data kriteria. Untuk menjalankan *function fkriteria* diperlukan sintaks tambahan. Sintaks tersebut adalah "select" yang diikuti fungsi yang akan dijalankan. Semisal, akan ditambahkan data kriteria baru yaitu kriteria ipk, bobotnya 2.6 untuk prodi informatika, maka sintaksnya adalah "*select fkriteria('t','IPk','Benefit','2.6','USR1','Informatika','')*". Hasil tangkapan layar penulisan ulang *function fkriteria* dan eksekusi tambah data kriteria ditunjukan pada gambar 6.



Gambar 6. Penulisan ulang stored program fkriteria dan pengujiannya

Dari gambar 6 diatas dapat terlihat dengan jelas bahwa untuk penulisan ulang dari function fkriteria hanya memerlukan waktu 0.0107 detik dan untuk waktu untuk tambah data menggunakan function fkriteria memerlukan waktu 0.0059 detik dengan nilai kembalian dari function fkriteria adalah "t1". Hasil "t1" membuktikan bahwa proses input data berhasil dan hasil *stored program* yang dibangun sesuai harapan.

f. Keamanan data

Jika melihat baris kode untuk tambah data menggunakan menjalankan *stored program fkriteria*, pada sintaks tersebut tidak terlihat nama tabel yang dituju, kolomnya dan informasi tabel yang berhungungan dengan tabel kriteria. Dan jika kita melihat dari sisi basisdata, keamanan basisdata juga menjadi perhatian tersendiri bagi para programmer agar data pada basisdata aman. Kedua baik keamanan basis data maupun stored programs, memiliki kontribusi nyata untuk keamanan aplikasi dan data.

Penelitian ini menawarkan kebaruan dengan mengintegrasikan secara menyeluruh berbagai komponen Stored Programs yaitu *Stored Procedures*, *Functions*, *Triggers*, dan *Events* dalam sistem basis data untuk SPK akademik. Berbeda dari penelitian sebelumnya yang hanya membandingkan efisiensi waktu eksekusi query (Rohmana et al., 2019; Aulia et al., 2023), studi ini menekankan pada desain strategi keamanan data berbasis server-

side yang langsung mengontrol akses dan validasi data pada tingkat basis data. Dengan demikian, sistem menjadi lebih tahan terhadap penyalahgunaan akses dan serangan SQL Injection, sekaligus mempertahankan efisiensi sistem. Pendekatan ini juga menghindari ketergantungan pada logika keamanan di sisi aplikasi yang umumnya lebih mudah disusupi (Fathulloh & Adauwiyah, 2021; Hartono, 2021).

Secara global, model sistem yang dikembangkan dalam penelitian ini dapat diterapkan pada berbagai sektor yang memerlukan proses pengambilan keputusan berbasis data sensitif, seperti perguruan tinggi, instansi pemerintahan, dan korporasi yang bergerak di bidang sumber daya manusia. Dengan meningkatnya ancaman kejahatan siber yang menyasar informasi pribadi dan institusional (Satria et al., 2024; Yolanda Sari Ks et al., 2022), implementasi Stored Programs seperti yang diusulkan dalam penelitian ini dapat menjadi solusi praktis dalam memperkuat sistem dari sisi internal (database). Selain itu, pendekatan ini mendukung prinsip-prinsip *data governance* dan *privacy-by-design* yang mulai menjadi standar global dalam pengelolaan sistem informasi.

KESIMPULAN

Keamanan data pada basisdata yang merupakan tujuan utama dari penelitian terpenuhi, karena kode untuk *stored program* yang dibuat tersimpan baik disisi server. Diperlukan usaha lebih bagi orang yang tidak bertanggung jawab untuk melakukan kejahatan manipulasi data. Stored program yang dibuat sudah sesuai dengan harapan, stored program dapat berjalan untuk proses CRUD dan waktu yang diperlukan sangat singkat.

DAFTAR PUSTAKA

Aulia, C. P., Pratama, M. Y., & Dewi, H. L. (2023). Perbandingan Performa Query Select Dasar, View, Dan Stored Procedure Pada Database Mysql. *Prosiding Seminar Nasional Teknologi dan Sistem Informasi*, *3*(1), 456–464. https://doi.org/10.33005/sitasi.v3i1.413

Dm, M. Y., Agustantia, M., & Zulaiha, S. (n.d.). Tindak Pidana Kejahatan Pemalsuan data (Data Forgery) dalam Bentuk Kejahatan Siber (Cyber Crime). *Jurnal Pendidikan dan Konseling*, *4*(6), 6635–6640. https://doi.org/10.31004/jpdk.v4i6.9365

Fathulloh, A. H., & Adauwiyah, H. I. (2021). Perbandingan Tingkat Efisiensi Waktu Query SELECT pada Database Interface Navicat dan SQLYog di MySQL DBMS. *Applied Information System and Management (AISM)*, *4*(2), 101–105. https://doi.org/10.15408/aism.v4i2.18369

Firdaus, M. (2024). Penerapan Stored Procedure untuk Memudahkan Modifikasi dan Update Formula Logic pada ETL. *Remik: Riset dan E-Jurnal Manajemen Informatika Komputer*, *8*(3). https://doi.org/10.33395/remik.v8i3.13752

Hartono, N. (2021). Comparison of Stored Procedures on Relational Database Management Systems. *JOURNAL TECH-E*, *4*(2), 8–15. https://doi.org/10.31253/te.v4i2.529

Information Functions. (n.d.). [MySQL Official Website]. Information Functions. Retrieved January 23, 2025, from https://dev.mysql.com/doc/refman/8.4/en/information-functions.html#function_row-count

Putri, C. A., Melan, M., Ulgina, S., Lestari, A., Firmansyah, D., & Zaliman, I. (2024). Meningkatkan Kesadaran Dan Pemahaman Mengenai Keamanan Data Pribadi Melalui Kegiatan Sosialisasi Di SMPN 2 Merawang. *IJDe: Indonesian Journal of Dedication and Education*, *3*(1), 18–23. https://doi.org/10.33019/ijde.v3i1.50

Rohmana, A. D. A., Mubarok, H., & Gunawan, R. (2019). Pengukuran Kinerja Stored Procedure Pada Database Relasional. *Jurnal Siliwangi Seri Sains dan Teknologi (Saintek)*, *5*(2), 51–55. https://doi.org/10.37058/jssainstek.v5i2.1197

Saputra, D. F. (2023). Literasi Digital Untuk Perlindungan Data Pribadi. *Jurnal Ilmu Kepolisian*, *17*(3), 1–8. https://doi.org/10.35879/jik.v17i3.454

Satria, A., Ulina, N. P. H., Safira, P., & Pangestu, B. (2024). Perspektif Hukum Terhadap Keamanan Data: Tantangan Dan Solusi Di Era Teknologi Informasi. *Warta Dharmawangsa*, *18*(1), 177–192. https://doi.org/10.46576/wdw.v18i1.4264

Septiadi, A. D., & Tripustikasari, E. (2022). Analisa Perbandingan Kinerja Penyimpanan Query Database Antara Stored Procedure dengan Function. *Peluang dan Tantangan Teknologi Terapan Untuk Mendukung Pembangunan Nasional Berkelanjutan*, *1*, 1–8. https://ejournal.pnc.ac.id/index.php/senovtek/article/view/1590

Triyono, J. (2023). Penerapan Hak Akses pada Perancangan Database Akademik untuk Meningkatkan Keamanan Data: Application of Access Rights in Academic Database Design to Improve Data Security. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, *3*(1), 50–59. https://doi.org/10.57152/malcom.v3i1.747

Yolanda Sari Ks, Madiasa Ablisar, Mahmud Mulyadi, & Jelly Leviza. (2022). Analisis Yuridis Terhadap Tindak Pidana Manipulasi Informasi Pengguna E-Commerce Menurut Undang-Undang Nomor 11 Tahun 2008 Tentang Informasi Dan Transaksi Elektronik Studi Putusan No. 542/Pid.Sus/2019/PN.Mlg. *Locus: Jurnal Konsep Ilmu Hukum*, *2*(2), 53–67. https://doi.org/10.56128/jkih.v2i2.22