

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

Prahenusa Wahyu Ciptadi¹, Tri Hastono^{2*}, Gema Kharismajati³, Firdiyan Syah⁴,
Nayaka Anung Fahriza⁵

^{1,2,3,4}Informatika, Universitas PGRI Yogyakarta, Indonesia

⁵TKIT Al Farabi, Indonesia

nusa@upy.ac.id¹, trihastono.13@gmail.com^{2*}, gemakharismajati@upy.ac.id³,
ryuakendent@upy.ac.id⁴, nayaka.anung@gmail.com⁵

*Corresponding Author

Manuscript Received February 10, 2024; Revised February 18, 2024; Accepted February 18, 2024;

Published July 26, 2024

ABSTRACT

Data security is becoming increasingly crucial in information system management, especially in academic environments that handle sensitive information. The reliance on digital systems demands the protection of data integrity, confidentiality, and availability. High-level data security is also required in the Decision Support System (DSS) application for selecting Teaching Assistants at Universitas PGRI Yogyakarta. The DSS helps determine candidates based on GPA, experience, and skills. However, this system is vulnerable to threats such as unauthorized data modifications, access abuse, SQL Injection, and cyber-attacks that can impact decision-making. Stored Programs in databases provide an effective solution to enhance data security. This research aims to design Stored Programs as a security strategy for data input into database tables. The types of Stored Programs designed in this study include Stored Procedures, Functions, and control programs. The design methodology for the Stored Programs follows the waterfall method. The test results indicate that the Stored Programs function as expected and execute in a very short time. The execution of Stored Programs takes only 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. Keamanan data yang tinggi juga diperlukan pada aplikasi Sistem Pendukung Keputusan (SPK) untuk pemilihan Asisten Dosen di Universitas PGRI Yogyakarta. 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. Penelitian ini bertujuan untuk merancang Stored Programs sebagai strategi keamanan input data kedalam tabel berbasis data. Untuk jenis Stored Programs yang dirancang pada penelitian ini adalah stored procedure dan functions serta kontrol program. Dan metode perancangan dari Stored Programs adalah metode waterfall. Hasil pengujian dari Stored Programs sudah berjalan sesuai dengan harapan dan waktu yang sangat singkat. Untuk eksekusi Stored Programs memerlukan waktu sebanyak 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 (Rathore & Sharma, 2017; 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.; Ujung & Nasution, 2023; Yolanda Sari Ks et al., 2022).

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; Rathore & Sharma, 2017; Rohmana et al., 2019; Ujung & Nasution, 2023).

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 sistem (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).

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.

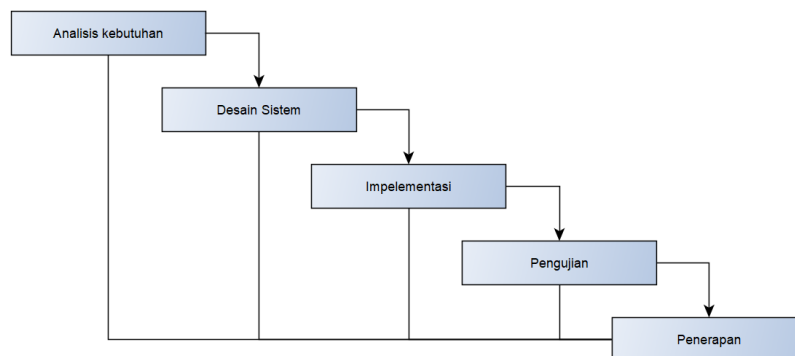
METODE PENELITIAN

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. Sebagai sumber data pada proses wawancara adalah ketua Program Studi Informatika UPY. Hasil dari proses wawancara yang dilakukan dihasilkan spesifikasi basisdata dan *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

HASIL DAN PEMBAHASAN

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

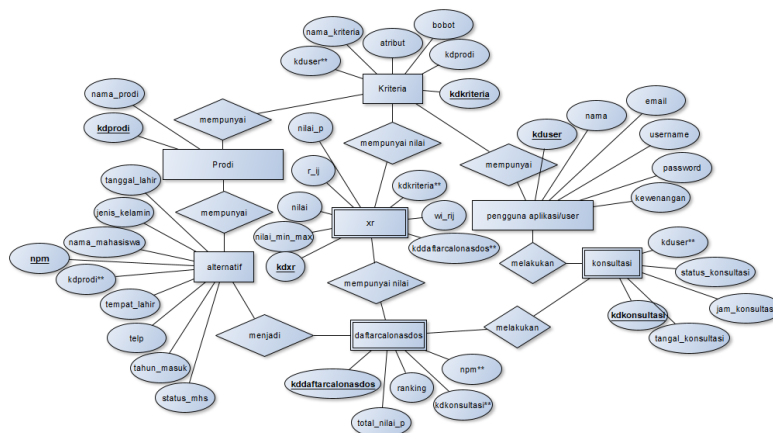
data kriteria asisten dosen yang ada di program studi Informatika UPY.

Tabel 1. Data Kriteria SPK Penentuan Asisten Dosen

NO	Kode Kriteria	Nama Kriteria	Exist
1	CRT1	Ipk	Benefit
2	CRT2	Keterampilan Komunikasi	Benefit
3	CRT3	Keterampilan Menyampaikan 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*.

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 daftarcalonados, dan entitas xr. Untuk gambar dari ERD penelitian dapat dilihat pada gambar 2.



Gambar 2. ERD Penelitian

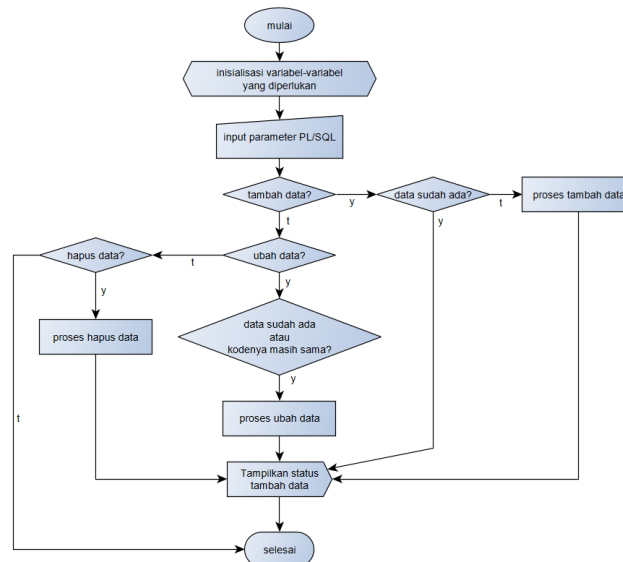
Setelah ERD penelitian sudah jadi, maka dibuatlah desain tabel-tabel basisdata beserta relasi antar tabelnya. Dari semua entitas yang ada pada ERD baik yang kuat maupun lemah nantinya akan turun mejadi tabel-tabel basisdata penelitian. Untuk tabel-tabel basisdata beserta relasinya dapat dilihat pada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



- 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.

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 ditunjukkan 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** ditunjukkan pada gambar 5 dibawah ini.

```

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

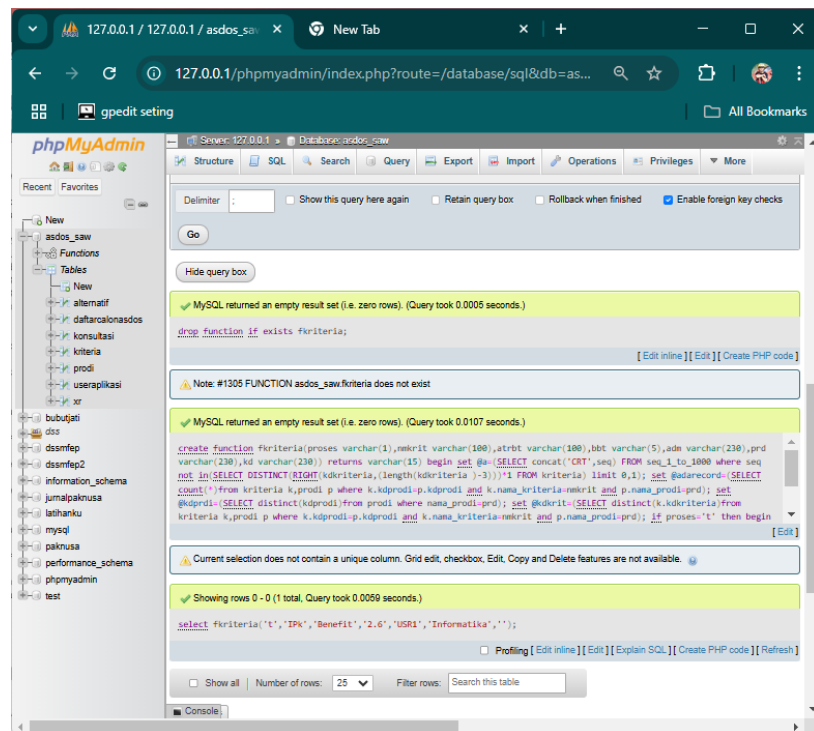
Tabel 3. Nilai Balik Fungsi *row_count()*(*Information Functions*, n.d.)

No	Proses	Nilai balik fungsi <i>row_count</i>
1	DML (select, delete, update, insert)	<ul style="list-style-type: none"> – Nilai fungsi “<i>row_count()</i>” 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 f kriteria* dijalankan. Dan hasil kombinasi ini, bisa diolah dan dimanfaatkan lebih lagi, misalnya sebagai informasi yang ditampilkan pada *dialog box* aplikasi.

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 eksekusi *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 eksekusi *function fkriteria* sebagai informasi untuk pengguna. 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 ditunjukkan 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.

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 berhubungan dengan tabel kriteria. Dan jika kita melihat dari sisi basisdata, keamanan basisdata juga menjadi perhatian tersendiri pengelola basisdata tersebut. Kedua hal ini, baik dari sisi *programmer aplikasi* dan dan keamanan basisdata disisi server merupakan kombinasi keamanan data yang cukup mumpuni (Kuntadi & Feriandi, 2023; Sidharta & Wibowo, 2020; Ujung & Nasution, 2023). Dari beberapa penelitian pendahulu yang telah dilakukan dan menjadi rujukan penelitian ini, peneliti belum menemukan penggunaan *stored program* yang kompleks untuk proses CRUD

dalam tabel basisdata untuk keamanan input tabel basisdata (Aulia et al., 2023; Firdaus, 2024; Hartono, 2021; Rohmana et al., 2019). Sejauh yang peneliti ketahui, penelitian ini adalah penelitian dengan penggunaan *stored program* yang mengkombinasikan kontrol program, *stores procedure*, dan *function*.

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.

REFERENSI

- 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., Agustiantia, 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
- Kuntadi, A., & Feriandi, Y. (2023). Penggunaan SQL Server dalam Pengolahan Data Warehouse yang Praktis dan Berkelanjutan. *Jurnal Algoritma*, 20(2), 457–468. <https://doi.org/10.33364/algoritma/v.20-2.1522>
- 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>

- Rathore, S., & Sharma, A. (2017). Database Security- Attacks, Threats and Challenges. *International Journal of Engineering Research*, 5(10), 1–4. <https://doi.org/10.17577/IJERTCONV5IS10008>
- Rohmana, A. D. A., Mubarak, 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>
- Sidharta, K., & Wibowo, T. (2020). *Studi Efisiensi Sumber Daya Terhadap Efektivitas Penggunaan Database : Studi Kasus Sql Server Dan*. 1(1), 508–515.
- 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>
- Ujung, A. M., & Nasution, M. I. P. (2023). Pentingnya Sistem Keamanan Database Untuk Melindungi Data Pribadi. *Jurnal Sistem Informasi Dan Informatika*, 1(2), 44–47. <https://doi.org/10.47233/jiska.v1i2.929>
- 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>