

**LAPORAN PRAKTIKUM BASIS DATA**  
**TRIGGER**



**DISUSUN OLEH :**

Restu Wibisono

2340506061

**PROGRAM STUDI S1 TEKNOLOGI INFORMASI**  
**JURUSAN TEKNIK ELEKTRO FAKULTAS TEKNIK**  
**UNIVERSITAS TIDAR**

2024

**LAPORAN**  
**PRAKTIKUM BASIS DATA**



<b>Diisi Mahasiswa Praktikan</b>								
Nama Praktikan	Restu Wibisono							
NPM	2340506061							
Rombel	03							
Judul Praktikum	Trigger							
Tanggal Praktikum	30 Mei 2024							
<b>Diisi Asisten Praktikum</b>								
Tanggal Pengumpulan								
Catatan								

PENGESAHAN		NILAI
Diperiksa oleh :	Disahkan oleh :	
Asisten Praktikum	Dosen Pengampu	

Nanda Cahya Septiawan	Imam Adi Nata, S.Kom., M.Kom.	

**PROGRAM STUDI S1 TEKNOLOGI INFORMASI**  
**FAKULTAS TEKNIK**  
**UNIVERSITAS TIDAR**  
**Genap 2023/2024**

## I. Tujuan Praktikum

1. Mahasiswa mampu menjelaskan trigger dalam SQL
2. Mahasiswa mampu mengimplementasikan trigger dalam SQL

## II. Dasar Teori

1. **TRIGGER** Trigger dapat didefinisikan sebagai himpunan kode (prosedural) yang dieksekusi secara otomatis sebagai respon atas suatu kejadian berkaitan dengan tabel basis data. Kejadian (event) yang dapat membangkitkan trigger umumnya berupa pernyataan INSERT, UPDATE, dan DELETE. Berdasarkan ruang lingkupnya, trigger diklasifikasikan menjadi dua jenis: row trigger dan statement trigger. Trigger baris (row) mendefinisikan aksi untuk setiap baris tabel; trigger pernyataan hanya berlaku untuk setiap pernyataan INSERT, UPDATE, atau DELETE. Dari sisi perilaku (behavior) eksekusi, trigger dapat dibedakan menjadi beberapa jenis; namun umumnya ada dua jenis: trigger BEFORE dan AFTER. Sesuai penamaannya, jenis-jenis ini merepresentasikan waktu eksekusi trigger—misalnya sebelum atau sesudah pernyataan-pernyataan yang berkorespondensi.

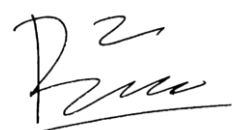
Adakalanya trigger dipandang sebagai bentuk spesifik dari stored procedure (terkait pendefinisian body). Bagaimanapun, trigger akan dipanggil (secara otomatis) ketika event terjadi, sedangkan stored procedure harus dipanggil secara eksplisit.

2. Trigger dalam MySQL MySQL mendukung fitur trigger termasuk juga stored procedure dan view sejak versi 5.0.2 Sebagaimana objek-objek lainnya, trigger diciptakan menggunakan pernyataan CREATE. Sintaks pendefinisian trigger diperlihatkan sebagai berikut:

```
CREATE
[DEFINER = { user | CURRENT_USER }]
TRIGGER trigger_name trigger_time trigger_event
ON tbl_name FOR EACH ROW trigger_stmt
```

MySQL tidak mengizinkan multiple trigger dengan waktu aksi dan event sama per tabel. Misalkan di tabel A sudah didefinisikan trigger AFTER INSERT, maka kita tidak boleh mendefinisikan trigger AFTER INSERT lagi; namun AFTER EDIT, AFTER DELETE, atau BEFORE (INSERT, EDIT, dan DELETE) bisa diterima.

Tanda Tangan



### III. Metode Praktikum

#### A. Alat dan bahan

Alat :

1. PC (Komputer)
2. Keyboard
3. Mouse

Bahan :

1. Operating System Windows 10
2. File Materi Praktikum
3. Aplikasi Paket Web server XAMPP
4. Aplikasi Kantor (Microsoft Office/Libre Office/WPS Office/etc)

#### B. Langkah kerja (diisi seperti dengan modul)

Dalam praktikum ini digunakan dua buah tabel bernama barang dan pembelian dengan data sebagai seperti berikut:

Tabel barang

id_brg	nama_brg	stok
A10	Mouse	10
A11	Keyboard	15
A12	DVD R-W	10

Tabel pembelian

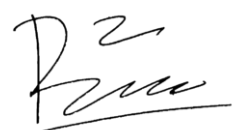
id_pem	id_brg	jml_beli
1	A10	5

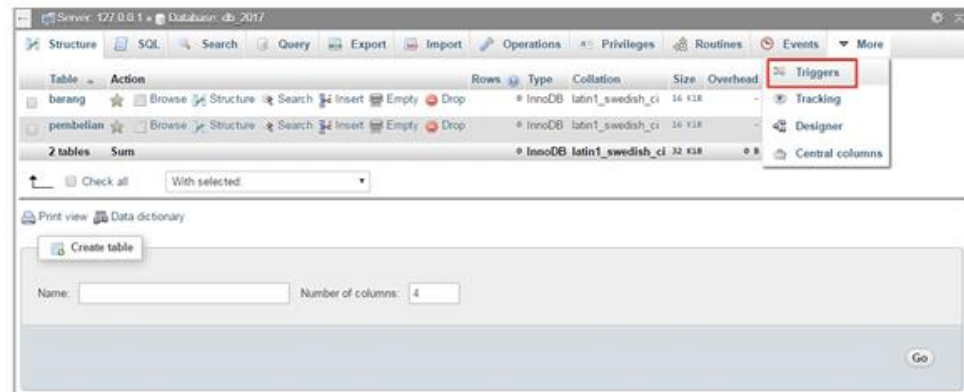
1. Menggunakan Trigger

Operasi-operasi berkenaan dengan pendefinisian trigger tidak berbeda dengan objek-objek database lainnya.

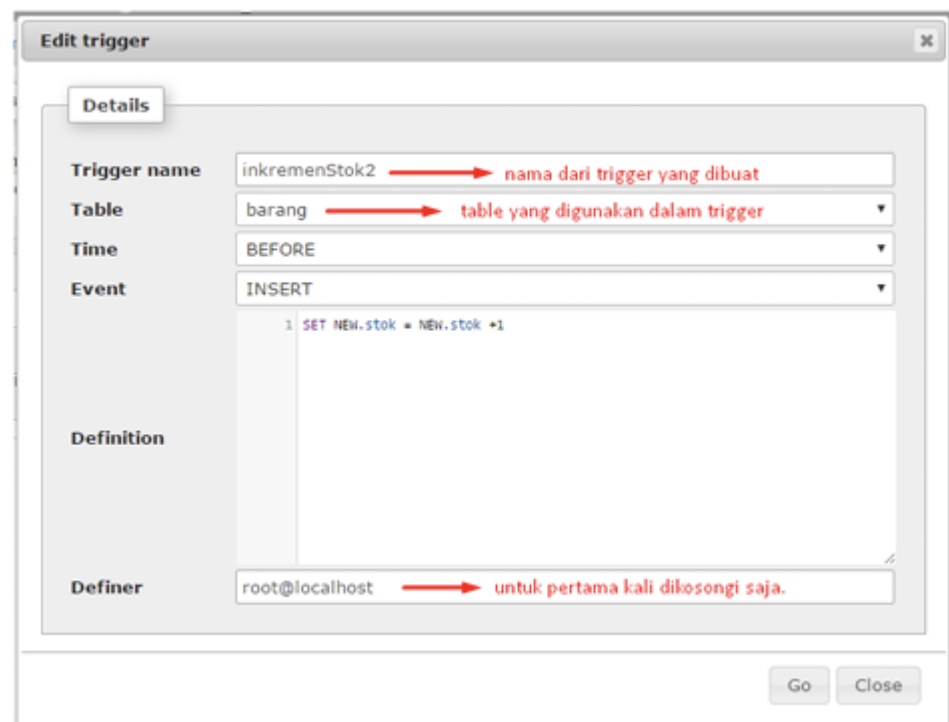
- a. Masuk kedalam database yang digunakan untuk menciptakan tabel tadi, kemudian masuk ke menu Trigger.

Tanda Tangan





- b. Pilih Add Trigger pada bagian New. Dan isikan sesuai dengan gambar berikut.



- c. Setelah berhasil membuat trigger. Kembali ke SQL database dan tuliskan sintak berikut:

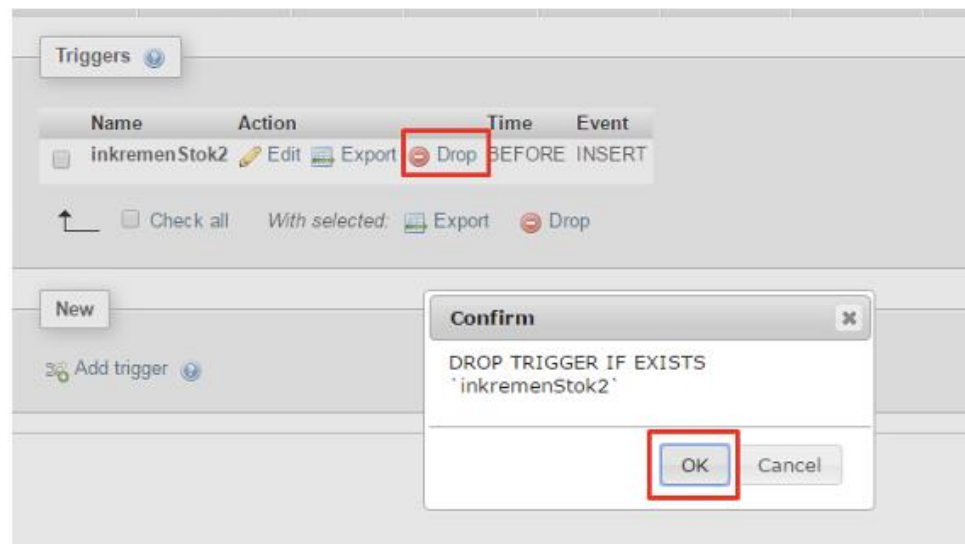
***INSERT INTO barang VALUES('A13', 'Modem', 5);***

- d. Jika berhasil maka akan menampilkan data seperti pada gambar berikut



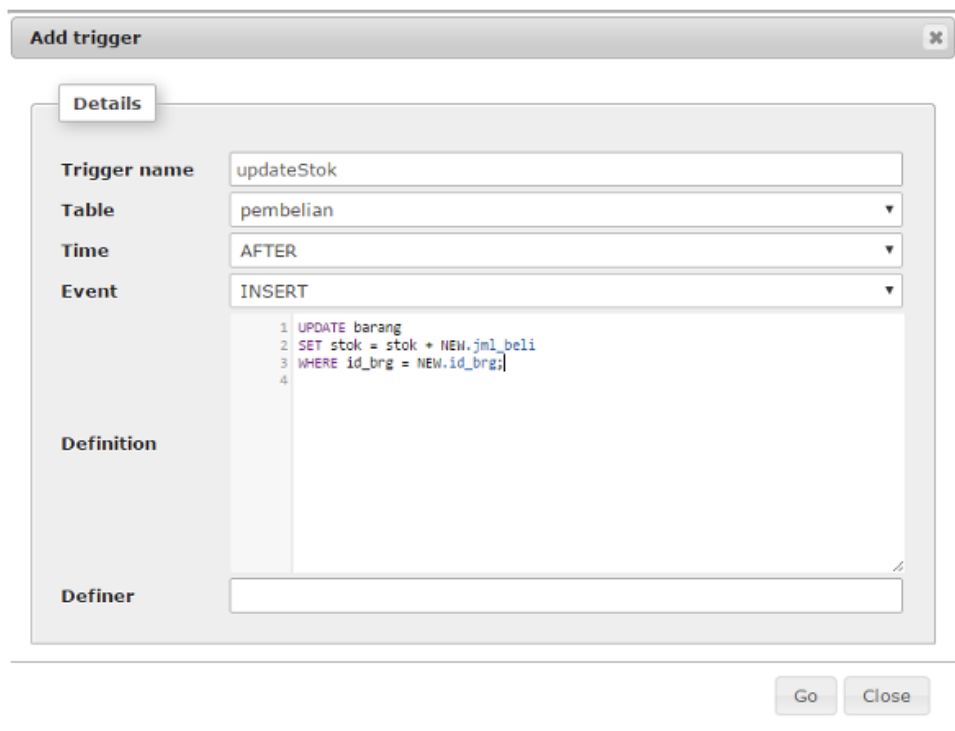
- e. Sebagaimana objek-objek database lainnya, kita menghapus trigger dengan menggunakan perintah DROP dengan ketentuan DROP TRIGGER nama\_trigger. Atau bisa langsung ke menu trigger dan menghapusnya dengan cara berikut.

Tanda Tangan



## 2. Keyword OLD dan NEW

Untuk merujuk ke kolom-kolom tabel yang diasosiasikan dengan trigger, kita menggunakan keyword OLD dan NEW. Keyword OLD mengacu pada nilai lama, sedangkan NEW merepresentasikan nilai baru. Di trigger INSERT, kita hanya dapat menggunakan keyword NEW karena tidak ada data lama.



Pada contoh di atas, penambahan data pembelian akan mengakibatkan nilai stok barang berubah menyesuaikan banyaknya nilai jumlah pembelian.

### a. Cek tabel barang

Tanda Tangan

***SELECT \* FROM barang***

`SELECT * FROM barang`

☐ Show all | Number of rows: 25 | Filter rows: Sea

Sort by key: None

+ Options

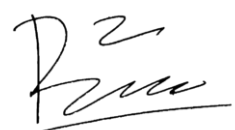
				id_brg	nama_barang	stok
<input type="checkbox"/>	Edit	Copy	Delete	A10	Mouse	16
<input type="checkbox"/>	Edit	Copy	Delete	A11	Keyboard	16
<input type="checkbox"/>	Edit	Copy	Delete	A12	DVD R-W	11
<input type="checkbox"/>	Edit	Copy	Delete	A13	Modem	6

- b. Kemudian masukkan data kedalam table pembelian

```
INSERT INTO `pembelian` (`id_pembelian`, `id_brg`, `jml_beli`) VALUES ('2', 'A10', '10');
```

- c. Cek kembali data dalam table barang

Tanda Tangan





```
SELECT * FROM `barang`
```

☐ Show all | Number of rows: 25 | Filter rows:

Sort by key: 

None

+ Options

					id_brg	nama_barang	stok		
<input type="checkbox"/>		Edit		Copy		Delete	A10	Mouse	20
<input type="checkbox"/>		Edit		Copy		Delete	A11	Keyboard	16
<input type="checkbox"/>		Edit		Copy		Delete	A12	DVD R-W	11
<input type="checkbox"/>		Edit		Copy		Delete	A13	Modem	6

Untuk kasus trigger DELETE, keyword yang bisa digunakan hanya OLD. Misalkan kita ingin mendefinisikan trigger untuk menghapus semua data pembelian manakala data barang yang sesuai—diindikasikan melalui primary key dan foreign key—dihapus.

- Membuat trigger baru seperti langkah sebelumnya

Tanda Tangan

**Edit trigger**

**Details**

Trigger name: deleteChild

Table: barang

Time: AFTER

Event: DELETE

Definition:

```
1 -- Hapus data pembelian yang berkorespondensi
2 DELETE FROM pembelian
3 WHERE id_brg = OLD.id_brg
```

Definer: root@localhost

Go Close

b. Hapus data pada tabel barang

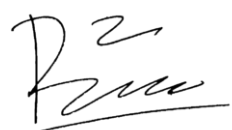
```
DELETE FROM barang WHERE id_brg = 'A10'
```

c. Cek apakah trigger berjalan dengan baik dengan menampilkan tabel pembelian

```
SELECT * FROM `pembelian`
```

Khusus untuk operasi UPDATE, kita bisa memanfaatkan keyword NEW maupun OLD.

Tanda Tangan



**Add trigger**

**Details**

Trigger name: updateStokEdit

Table: pembelian

Time: AFTER

Event: UPDATE

Definition:

```

1 -- Update nilai stok barang
2 UPDATE barang
3 SET stok = stok + (NEW.jml_beli - OLD.jml_beli)
4 WHERE id_brg = NEW.id_brg;
5

```

Definer:

Go Close

- a. Lakukan update pada tabel pembelian

```

UPDATE pembelian SET jml_beli= 20 WHERE id_pembelian=3

```

- b. Cek apakah trigger berjalan dengan baik dengan cara tampilkan pada tabel barang

```

SELECT * FROM `barang`

```

☐ Show all | Number of rows: 25 | Filter rows: Search

Sort by key: None

Options

				id_brg	nama_barang	stok
<input type="checkbox"/>	Edit	Copy	Delete	A10	Mouse	30
<input type="checkbox"/>	Edit	Copy	Delete	A11	Keyboard	16
<input type="checkbox"/>	Edit	Copy	Delete	A12	DVD R-W	11
<input type="checkbox"/>	Edit	Copy	Delete	A13	Modem	6

Tanda Tangan

*[Handwritten Signature]*

### 3. Trigger Kompleks

Keberadaan trigger secara nyata mampu mengatasi berbagai persoalan sulit, misalnya berkaitan dengan integritas atau audit data. Ini tentu sangat masuk akal, karena trigger juga bisa mengandung pernyataan-pernyataan yang kompleks termasuk pencabangan, pengulangan, fungsi-fungsi agregat, bahkan kode pemanggilan prosedur. Sebagai ilustrasi sederhana, kita bisa mendefinisikan trigger untuk memeriksa operasi penambahan data barang. Skenarionya, jika data sudah ada, berikan status eksistensi barang; sebaliknya, data bisa langsung dimasukkan.

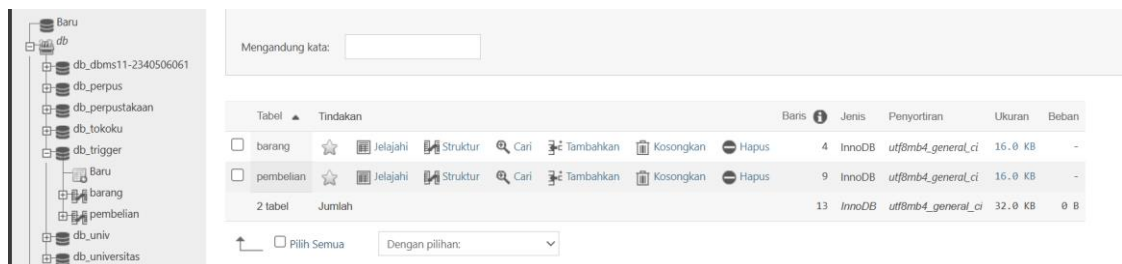
```
CREATE TRIGGER auditBarang
BEFORE INSERT ON barang

FOR EACH ROW BEGIN

    IF NOT EXISTS (SELECT id brg FROM barang WHERE id brg =
NEW.id_brg)
    THEN
        SET NEW.nama_brg = NEW.nama_brg, NEW.stok = NEW.stok;
    ELSE
        SET @status = CONCAT('Id ', NEW.id_brg, ' sudah ada');
    END IF;
```

## IV. Soal

### 1. Membuat database



(Gambar 4.1.1)

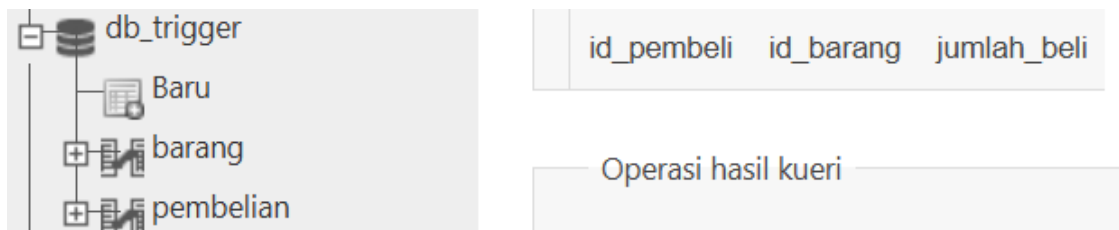
Membuat database 'Trigger' yang berisikan 2 tabel yaitu table 'barang' dan table 'pembelian'.

Tanda Tangan

				id_barang	nama_barang	stok
<input type="checkbox"/>		Ubah		Salin		Hapus
				A10	Mouse	10
<input type="checkbox"/>		Ubah		Salin		Hapus
				A11	Keyboard	15
<input type="checkbox"/>		Ubah		Salin		Hapus
				A12	DVD R-W	21343
<input type="checkbox"/>		Ubah		Salin		Hapus
				A13	Modem	1112

(Gambar 4.1.2 TABEL BARANG)

Tabel barang berisikan 3 kolom, dan kolom tersebut diisi dengan barang barang yang memiliki id dan jumlah stok yang ada.



(Gambar 4.1.3 TABEL PEMBELIAN)

Tabel pembelian nantinya akan direlasi dengan table barang, dan jika ada barang yang dibeli akan tercatat dalam table pembelian.

## 2. Membuat TRIGGER

Detail

Nama trigger

Tabel

Waktu

Kejadian

Definisi

```

1 IF NEW.jumlah_beli > 50 AND NEW.jumlah_beli <= 100 THEN
2   SET NEW.jumlah_beli = NEW.jumlah_beli +5;
3   UPDATE barang
4   SET stok = stok - (NEW.jumlah_beli+5)
5   WHERE id_barang = NEW.id_barang;
6 ELSEIF NEW.jumlah_beli > 100 AND NEW.jumlah_beli <= 150 THEN
7   SET NEW.jumlah_beli = NEW.jumlah_beli +10;
8   UPDATE barang
9   SET stok = stok - (NEW.jumlah_beli+10)
10  WHERE id_barang = NEW.id_barang;
11 ELSEIF NEW.jumlah_beli > 150 THEN
12   SET NEW.jumlah_beli = NEW.jumlah_beli +20;
13   UPDATE barang
14   SET stok = stok - (NEW.jumlah_beli+20)
15   WHERE id_barang = NEW.id_barang;
16 END IF

```

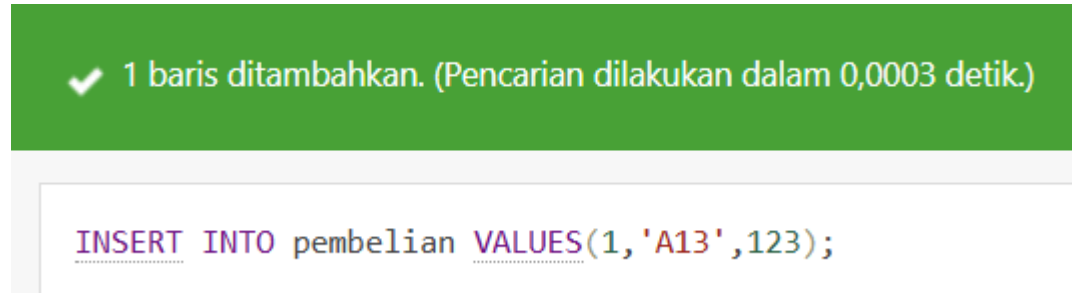
Menetapkan

(Gambar 4.2.1)

Tanda Tangan

Membuat sebuah trigger INSERT pembelian untuk menambahkan fitur bonus di dalamnya. Aturannya adalah jika pembelian > 50 dan < 100, maka bonus = 5; jika pembelian > 100 dan < 150, maka bonus = 10; jika pembelian > 150, maka bonus = 20. Ingat, aturan penyesuaian stok barang masih berlaku, setiap ada pembelian maka stok akan berkurang.

### 3. Membuat perintah SQL



(Gambar 4.3.1)

Pada VALUES terdapat 3 bagian, pertama adalah id prmbeli, selanjutnya id barang yang dibeli dan terakhir adalah jumlah barang yang dibeli.

## V. Kesimpulan

Dengan memanfaatkan trigger, mahasiswa dapat melihat bagaimana perubahan data dapat dikelola secara efisien dan otomatis sesuai dengan aturan yang telah ditetapkan. Namun, penting juga untuk memahami bahwa penggunaan trigger harus dilakukan dengan hati-hati untuk menghindari dampak negatif pada kinerja basis data. Praktikum ini memberikan pemahaman praktis tentang bagaimana merancang dan mengimplementasikan trigger dengan tepat untuk meningkatkan efisiensi dan keandalan sistem basis data.

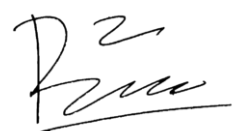
## VI. Referensi

Van Rossum, G. (2003). An introduction to Python (p. 115). F. L. Drake (Ed.).

Bristol: Network Theory Ltd..

<http://atk.fam.free.fr/fichiers/stage/Python/JF/site/pytut.pdf>

Tanda Tangan



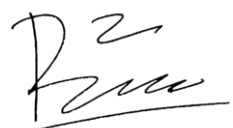
Kuhlman, D. (2009). A python book: Beginning python, advanced python, and python exercises (pp. 1-227). Lutz: Dave Kuhlman.

[https://www.davekuhlman.org/python\\_book\\_01.pdf](https://www.davekuhlman.org/python_book_01.pdf)

Python, W. (2021). Python. Python Releases for Windows, 24.

<https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=1f2ee3831eebfc97bfafd514ca2abb7e2c5c86bb>

Tanda Tangan

A handwritten signature in black ink, appearing to be 'R. Z. Rana', is written over a horizontal line.