**LAPORAN PRAKTIKUM BASIS DATA**

**PENERAPAN RECORD**



**DISUSUN OLEH :**

Restu Wibisono

2340506061

**PROGRAM STUDI S1 TEKNOLOGI INFORMASI**

**JURUSAN TEKNIK ELEKTRO FAKULTAS TEKNIK**

**UNIVERSITAS TIDAR**

**2024**

**LAPORAN**

**PRAKTIKUM BASIS DATA**



|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Diisi Mahsiswa Praktikan** | | | | | | | | |
| Nama Praktikan | Restu Wibisono | | | | | | | |
| NPM | 2340506061 | | | | | | | |
| Rombel | 03 | | | | | | | |
| Judul Praktikum | Penerapan Record | | | | | | | |
| Tanggal Praktikum | 18 April 2024 | | | | | | | |
| **Diisi Asisten Praktikum** | | | | | | | | |
| 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**

## Tujuan Praktikum

1. Mampu menjelaskan record pada tabel basis data
2. Mampu menerapkan record pada tabel basis data

## Dasar Teori

Dalam pembahasan sebelumnya kita telah belajar untuk mengelola basis data dengan menggunakan aplikasi GUI yaitu dbeaver. Aplikasi tersebut dapat mempermudah kita dalam pengelolaan basis data. Akan tetapi sebagai orang yang bekerja pada lingkungan basis data kita dituntut untuk mengelola basis data dengan menggunakan command line interface, sehingga dalam pembahasan mata kuliah praktikum basis data ini kita harus belajar pengelolaan basis data dengan menggunakan CLI dari dasar hingga tingkat lanjut. Dalam pengelolaan basis data kita tidak terlepas dari pengelolaan struktur dari basis data sampai dengan pengelolaan record dari basis data.

Record dalam basis data adalah satuan informasi yang terdiri dari sekumpulan elemen data (fields atau kolom) yang terkait satu sama lain. Record biasanya merepresentasikan satu entitas atau objek dalam dunia nyata, seperti karyawan, produk, pelanggan, atau transaksi. Dalam basis data relasional, record disebut juga dengan baris (row) dalam tabel.

Sebuah record berisi data yang terstruktur sesuai dengan skema tabel yang mendefinisikan kolom kolomnya. Setiap kolom dalam tabel mewakili atribut atau properti tertentu dari entitas yang direpresentasikan oleh tabel tersebut. Misalnya, dalam tabel pelanggan, kolom-kolomnya bisa mencakup ID pelanggan, nama, alamat, dan nomor telepon.

Record dapat berinteraksi dengan basis data melalui operasi seperti CRUD (Create, Read, Update, Delete). Pengguna dapat membuat record baru dengan memasukkan data ke dalam tabel, membaca record untuk melihat data yang ada, memperbarui record untuk mengubah data yang sudah ada, atau menghapus record yang tidak diperlukan.

Dalam basis data relasional, setiap record biasanya memiliki kunci primer (primary key) yang unik untuk membedakannya dari record lainnya. Kunci primer memastikan integritas data dan memfasilitasi akses yang efisien ke record tertentu. Record juga dapat memiliki hubungan dengan record di tabel lain melalui kunci asing (foreign key), memungkinkan representasi hubungan antar entitas dalam basis data.

Dalam praktiknya, record adalah elemen dasar yang menjadi bagian penting dalam pengelolaan dan penggunaan data dalam suatu aplikasi atau sistem informasi. Record memungkinkan pengguna dan sistem untuk mengakses, memanipulasi, dan menganalisis data dengan cara yang terstruktur dan efisien.

## Metode Praktikum

## 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)

## Langkah kerja

1. Insert

INSERT Perintah INSERT pernah kita bahas pada pertemuan sebelumnya. Perintah ini digunakan untuk memasukkan sebuah record ke dalam sebuah tabel. Perintah INSERT dapat kita tuliskan dengan dua cara yaitu :

1. Menentukan nama kolom dan nilai yang akan kita masukkan ke dalam tabel. Bentuk pertama ini dapat kita gunakan dengan menentukan secara detail kolom mana saja yang hendak kita ini dan kolom mana saja yang akan kita abaikan. Secara umum bentuk perintah cara pertama adalah sebagai berikut:

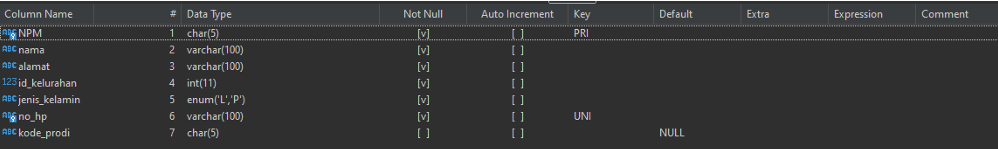
INSERT INTO table\_name (column1, column2, column3,…)

VALUES (value1, value2, value3,…);

1. Jika kita menghendaki akan mengisi semua kolom dalam sebuah tabel maka kita tidak perlu menentukan secara detail nama kolom yang akan kita isi. Kita hanya perlu menuliskan nikai apa saja yang akan kita isikan ke dalam kolom secara berurutan dari kolom sebelah kiri dari sebuah tabel. Bentuk INSERT dengan mengisi semua kolom dalam tabel secara umum adalah sebagai berikut :

INSERT INTO table\_name VALUES (value1, value2, value3,…);

Untuk lebih memahami penggunakan perintah INSERT pada basis data, kita dapat mempraktikkan pada data berikut : Buatlah sebuah basis data dan buatlah sebuah tabel dengan struktur sebagai berikut : tbl\_mahasiswa



(Gambar 1)

Tabel pada Gambar 1 dapat kita isi record dengan tanpa mengisi kode\_prodi. Perintah pengisian tabel pada kolom tertentu dapat kita ktuliskan sebagai berikut:

INSERT INTO tbl\_mahasiswa (NPM, nama, Alamat, id\_kelurahan, no\_hp, jenis\_kelamin) values (‘MHS99’,’Alexander Tukimin’, ‘Jl. Menthok Raya KM. Kemruyuk’, ‘00219’, ‘0812131415’, ‘L’);

Perintah di atas dapat kita amati bahwa kode\_prodi tidak kita masukkan sehingga secara default record pada tbl\_mahassiswa dengan NPM ‘MHS99’ pada kolom kode\_prodi akan kosong. Selain itu posisi jenis\_kelamin dan no\_hp pada perintah tersebut juga tidak kita urutkan dalam peletakkan perintah, cara pertama ini akan mengijinkan hal tersebut karena kolom yang akan kita isi kita sebutkan secara spesifik.

Perintah INSERT dengan cara kedua dapat kita lakukan jika kita ingin mengisi semua kolom pada tabel. Hal yang perlu diperhatikan jika menggunakan cara ini adalah kita harus mengisikan nilai/data dengan urutan kolom yang terdapat pada struktur tabel. Pada tabel di atas kita dapat mengurutkan nilai yang akan kita isikan mulai dari NPM sampai dengan kode\_prodi. Perintah tersebut dapat kita praktikkan pada contoh berikut :

INSERT INTO tbl\_mahasiswa values (‘MHS98’,’Nicola Arjo Utomo’, ‘Jl. Buntu Raya KM. 1/4’, ‘00219’, ‘L’, ‘0812131415’, ‘FT001’);

Jika dalam pengisian data/nilai terdapat kekurangan atau tidak urut maka akan terjadi error pada proses pengisian record.

1. Select

Perintah SELECT digunakan untuk mengambil data/record dari basis data lalu ditampilkan ke dalam tampilan monitor pada komputer. Perintah SELECT dapat diikuti dengan berbagai macam kondisi sehingga dapat menampilkan data secara spesifik. Beberapa kondisi tersebut dapat kita atur secara detail dengan menggabungkan bebebrapa kondisi sekaligus sehingga record yang ditampilkan sesuai dengan kebutuhan dari pengguna basis data. Secara umum perintah select adalah sebagai berikut :

1. SELECT column1, column2, ... FROM table\_name;
2. SELECT \* FROM table\_name;
3. SELECT \* FROM table\_name WHERE kondisi;
4. SELECT \* FROM table\_name WHERE kolom LIKE pattern;
5. SELECT \* FROM table\_name WHERE kondisi1 and kondisi2;
6. SELECT \* FROM table\_name WHERE kondisi1 or kondisi2;
7. SELECT \* FROM table\_name WHERE NOT kondisi;
8. SELECT \* FROM table\_name ORDER BY kolom1 ASC/DESC;
9. SELECT \* FROM table\_name LIMIT number;
10. SELECT MIN(colum\_name) FROM table\_name;
11. SELECT MAX(colum\_name) FROM table\_name;
12. SELECT COUNT(colum\_name) FROM table\_name;
13. SELECT AVG(colum\_name) FROM table\_name;
14. SELECT SUM(colum\_name) FROM table\_name;
15. 15. SELECT column\_name(s) FROM table\_name WHERE column\_name IN (value1, value2, ...);
16. SELECT column\_name(s) FROM table\_name WHERE column\_name BETWEEN value1 AND value2;
17. SELECT column\_name AS alias\_name FROM table\_name;
18. SELECT column\_name(s) FROM table\_name AS alias\_name;
19. SELECT column\_name(s) FROM table\_name WHERE condition GROUP BY column\_name(s);
20. SELECT column\_name(s) FROM table\_name WHERE EXISTS (SELECT column\_name FROM table\_name WHERE condition);

Berikut adala beberapa contoh penggunaan perintah SELECT untuk nemapilkan record dalam basis data :

1. SELECT
2. SELECT \* FROM tbl\_mahasiswa;
3. SELECT nama,no\_hp FROM tbl\_mahasiswa;

Perintah ini adalah perintah SELECT secara umum. Perintah ini akan menampilkan semua record yang ada dalam tabel tersebut. Perintah pertama akan menampilkan semua kolom dalam tabel, perintah kedua hanya akan menampilkan kolom secara spesifik yaitu nama dan no\_hp.

1. WHERE
2. SELECT \* FROM tbl\_mahasiswa WHERE jenis\_kelamin = ‘L’;
3. SELECT \* FROM tbl\_mahasiswa WHERE jenis\_kelamin = ‘L’ and kode\_prodi = ‘2’;
4. SELECT \* FROM tbl\_mahasiswa WHERE nama LIKE ‘i%’;

|  |  |
| --- | --- |
| LIKE Operator | Description |
| WHERE CustomerName LIKE 'a%' | Finds any values that start with "a" |
| WHERE CustomerName LIKE '%a' | Finds any values that end with "a" |
| WHERE CustomerName LIKE '%or%' | Finds any values that have "or" in any position |
| WHERE CustomerName LIKE '\_r%' | Finds any values that have "r" in the second position |
| WHERE CustomerName LIKE 'a\_%' | Finds any values that start with "a" and are at least 2 characters in length |
| WHERE CustomerName LIKE 'a\_\_%' | Finds any values that start with "a" and are at least 3 characters in length |
| WHERE ContactName LIKE 'a%o' | Finds any values that start with "a" and ends with "o" |

Perintah di atas adalah perintah SELECT dengan beberapa kondisi tertentu. Perintah pertama akan menampilkan semua record pada tabel dengan jenis kelamin L. Perintah kedua akan menampilkan semua record dengan jenis kelamin L dan kode prodi 2. Pada perintah ke dua selain dengan menggunakan logika ‘and’ kita juga dapat menggunakan logikan ‘or’ dan ‘not’. Perintah ketiga digunakan untuk mencari record dengan kondisi nama yang diawali dengan huruf ‘i’. Kondisi ‘LIKE’ dapat diisi dengan kondisi-kondisi tertentu dengan mengelola bentuk string dari nilai. Berikut adalah penjelasan kondisi LIKE :

1. ORDER BY dan LIMIT

Penrintah SELECT dengan dibarengi konsidi ORDER BY akan menampilkan record data secara berurutan sesuai dengan urutan kolom yang kita tentukan. Perintah LIMIT digunakan untuk membatasi jumlah data yang akan ditampilkan. Berikut adalah contoh perintah dengan ORDER BY dan LIMIT:

SELECT \* FROM tbl\_mahasiswa ORDER BY nama ASC LIMIT 2;

Perintah di atas akan menghasilkan tampilan data record secara urut berdasarkan nama mulai dari abjad a sampai dengan z. Coba kita ubah pada bagian ‘ASC’ menjadi ‘DESC’ lalu lihat apa perbedaan dari kedua perintah tersebut

1. NULL Values

Perintah NULL digunakan untuk menampilkan record dengan kolom tertentu yang bernilai NULL. Berikut adalah contoh penggunaan kondisi NULL pada perintah SELECT :

SELECT \* FROM tbl\_mahasiswa WHERE kode\_prodi IS NULL;

1. MIN, MX, COUNT, AVG, dan SUM

Perintah tersebut digunakan untuk menampilkan record/data dengan menggunakan operator-operator matematika dasar. Perintah ini hanya dapat diterapkan ke dalam kolom dengan tipe data number(int,float,dll) kecuali COUNT karena perintah ini akan menampilkan jumlah data. Contoh penggunaan perintah terssebut adalah sebagai berikut :

1. SELECT MIN(harga) FROM produk;
2. SELECT MAX(harga) FROM produk;
3. SELECT COUNT(nama) FROM produk;
4. SELECT AVG(harga) FROM produk;
5. SELECT SUM(harga) FROM produk;

Perintah pertama akan menampilkan data harga terendah, perintah kedua akan menampilkan data harga tertinggi, perintah ketiga akan menampilkan jumlah data, perintah keempat akan menampilkan rata-rata harga dan perintah terakhir akan menampilkan jumlah harga dari semua data.

1. IN and BETWEEN

Perintah IN digunakan untuk mencocokkan data yang akan ditampilkan dengan list data atau data pada tabel lain. Sedangkan BETWEEN digunakan untuk mencari data berdasarkan rentang tertentu. Berikut adalah contoh penggunakan perintah SELECT dengan kondisi IN dan BETWEEN :

1. ALIAS
2. SELECT \* FROM tbl\_mahasiswa WHERE alamat IN (‘bekasi', 'medan’);
3. SELECT \* FROM tbl\_mahasiswa WHERE alamat NOT IN (‘bekasi', 'medan’);
4. SELECT \* FROM tbl\_mahasiswa WHERE alamat IN (SELECT alamat FROM wali);
5. SELECT \* FROM Produk WHERE Harga BETWEEN 10000 AND 20000;

Dalam SQL, alias digunakan untuk memberikan nama sementara kepada tabel atau kolom dalam tabel. Alias biasanya digunakan untuk membuat nama kolom lebih mudah dibaca dan dimengerti.

Alias hanya ada selama durasi query tersebut berjalan. Alias diciptakan dengan menggunakan kata kunci AS.

Ketika memberikan alias pada kolom atau tabel, kita mendefinisikan nama sementara yang dapat digunakan dalam query untuk merujuk pada kolom atau tabel tersebut. Ini sangat berguna terutama ketika kolom atau tabel memiliki nama yang panjang atau kompleks, sehingga bisa disingkat atau disederhanakan untuk meningkatkan kejelasan dan keterbacaan. Berikut adalah contoh penggunaan ALIAS :

SELECT nama AS jeneng FROM tbl\_mahasiswa;

Perintah di atas akan menampilkan record pada kolom nama dengan label ‘jeneng’.

1. Update

Perintah UPDATE dapat kita gunakan untuk mengubah record yang sudah tersimpan pada tabel data. Bentuk umum dari perintah ini adalah sebagai berikut:

UPDATE table\_name SET column1 = value1, column2 = value2, ...

WHERE condition;

Hal yang harus kita perhatikan saat melakukan perintah UPDATE adalah menentukan kondisi setelah perintah WHERE. Jika kita menghilangkan kondisi pada bagian WHERE data akan berubah secara keseluruhan, sehingga kita perlu berhati-hati dalam pembuatan perintah UPDATE. Perintah tersebut dapat kita praktikkan ke dalam tabel yang sebelumnya sudah kita buat yaitu tbl\_mahasiswa. Jika kita ingin mengubah nilai no\_hp dan kode\_prodi pada mahasiswa yang mempunyai NPM MHS98 maka kita dapat menuliskan perintah UPDATE sebagai berikut :

**UPDATE tbl\_mahasiswa SET** no\_hp = ‘08564656667’, kode\_prodi = ‘FT002’ **WHERE** NPM=’MHS98’

Perintah di atas kita menggunakan NPM sebagai kondisi karena NPM bersifat unik sehingga tidak ada record yang sama dalam kolom NPM. Jika kita menggunakan kolom lain sebagai kondisi harus kita pastikan bahwa kolom tersebut bersifat unik, jika tidak maka jika kolom tersebut memeliki beberapa record dengan nilai yang sama, maka semua record tersebut akan berubah.

1. Delete

Perintah DELETE kita gunakan untuk pengelolaan record yang terakhir yaitu menghapus record. Hal yang harus kita perhatikan dalam pembuatan perintah DELETE seperti pada perintah UPDATE yaitu pada bagian kondisi (WHERE). Kita harus berhati-hati dalam membuat perintah ini karena jika kita salah dalam memnentukan kondisi atau kita tidak menentukan kondisi maka data/record akan terhapus semua secara permanen. Bentuk perintah DELETE secara umum adalah sebagai berikut :

DELETE FROM table\_name WHERE condition;

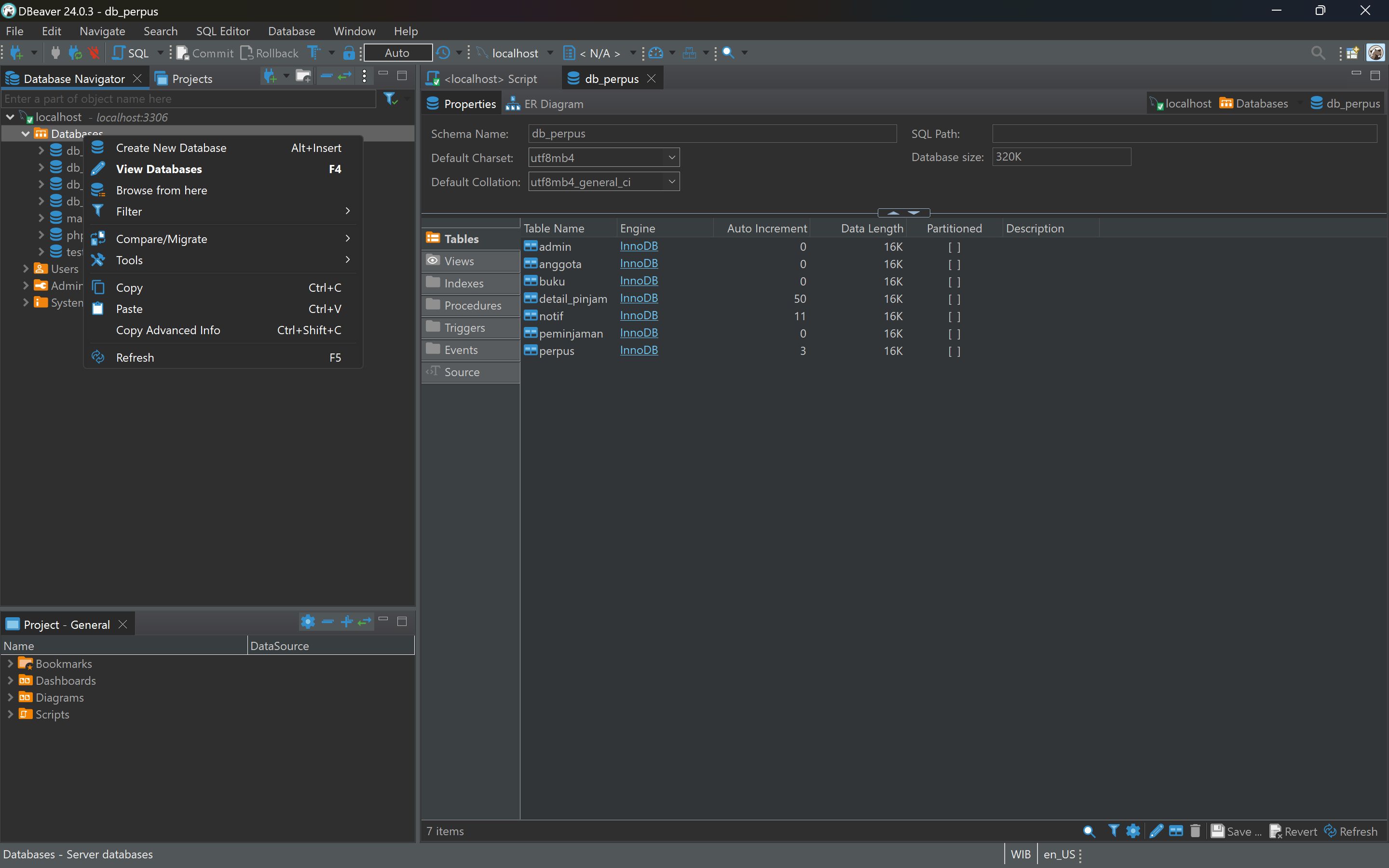
Sebagai contoh jika kita akan menghapus record pada tabel tbl\_mahasiswa dengan NPM MHS99 maka kita dapat menuliskan perintah berikut ini :

**DELETE FROM** tbl\_mahasiswa **WHERE** NPM = ‘MHS99’

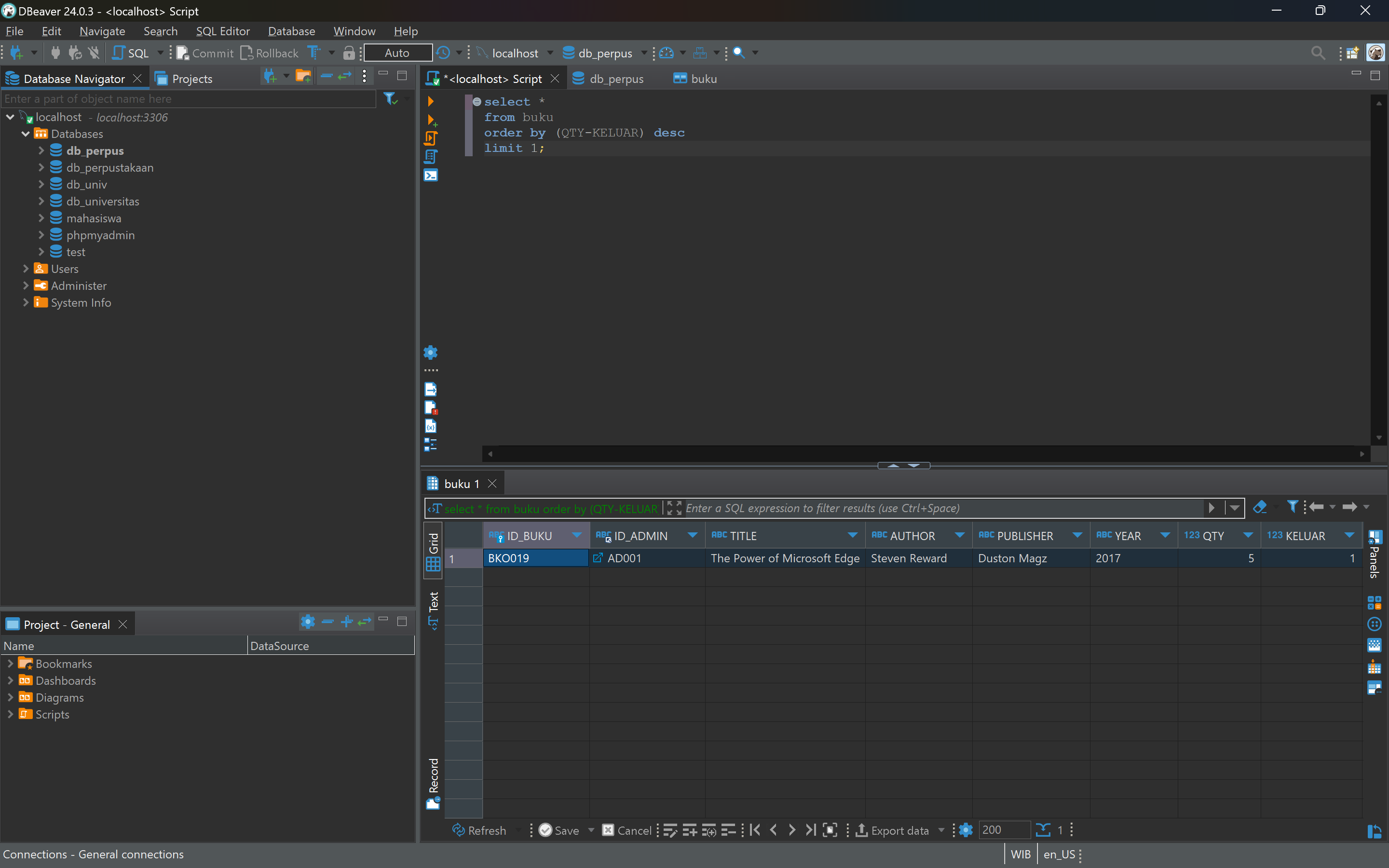
Jika kita menghilangkan bagian WHERE maka data/record dalam tabel tbl\_mahasiswa akan terhapus semua.

## Soal

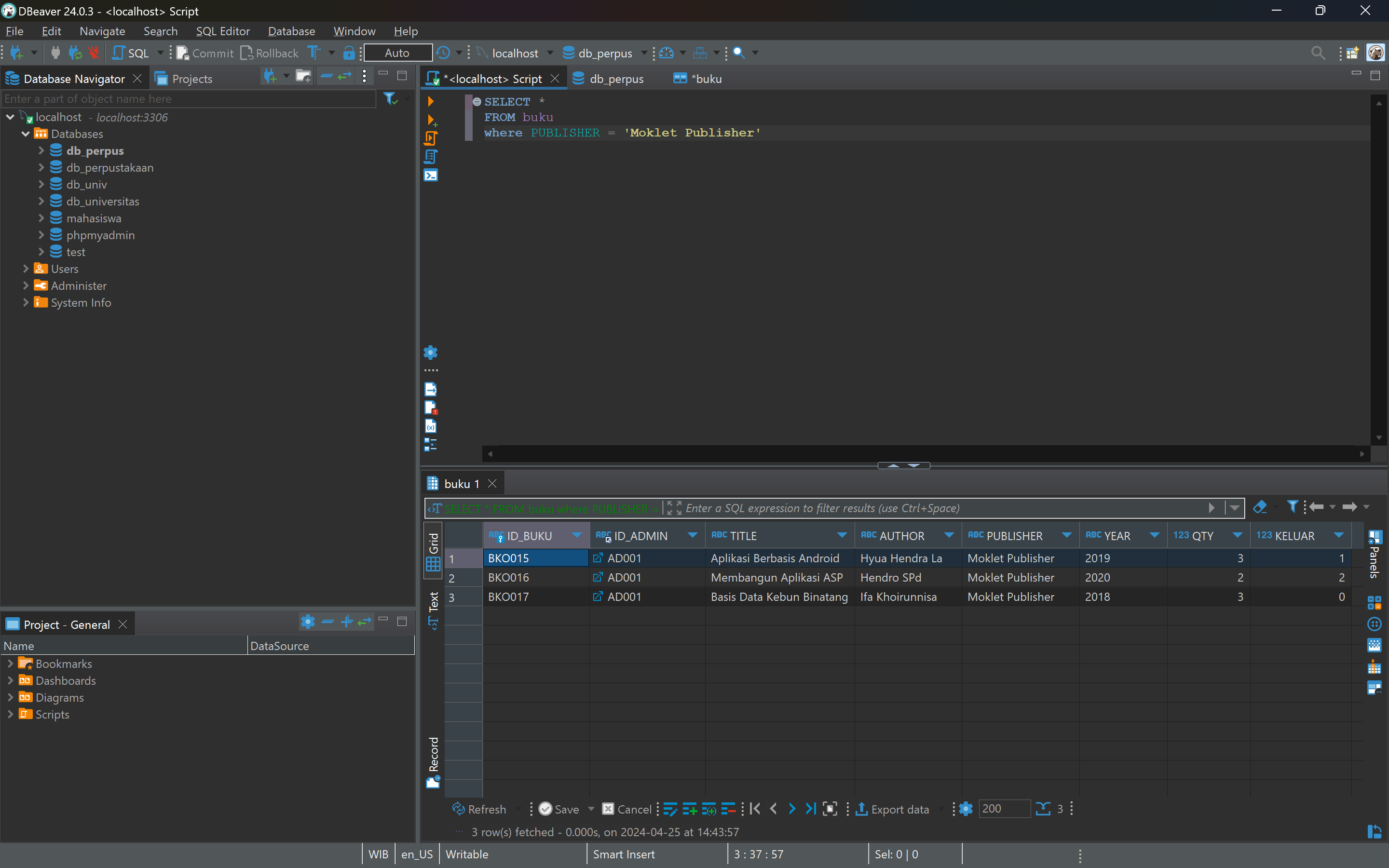
1. Restore Basis Data Pada Server



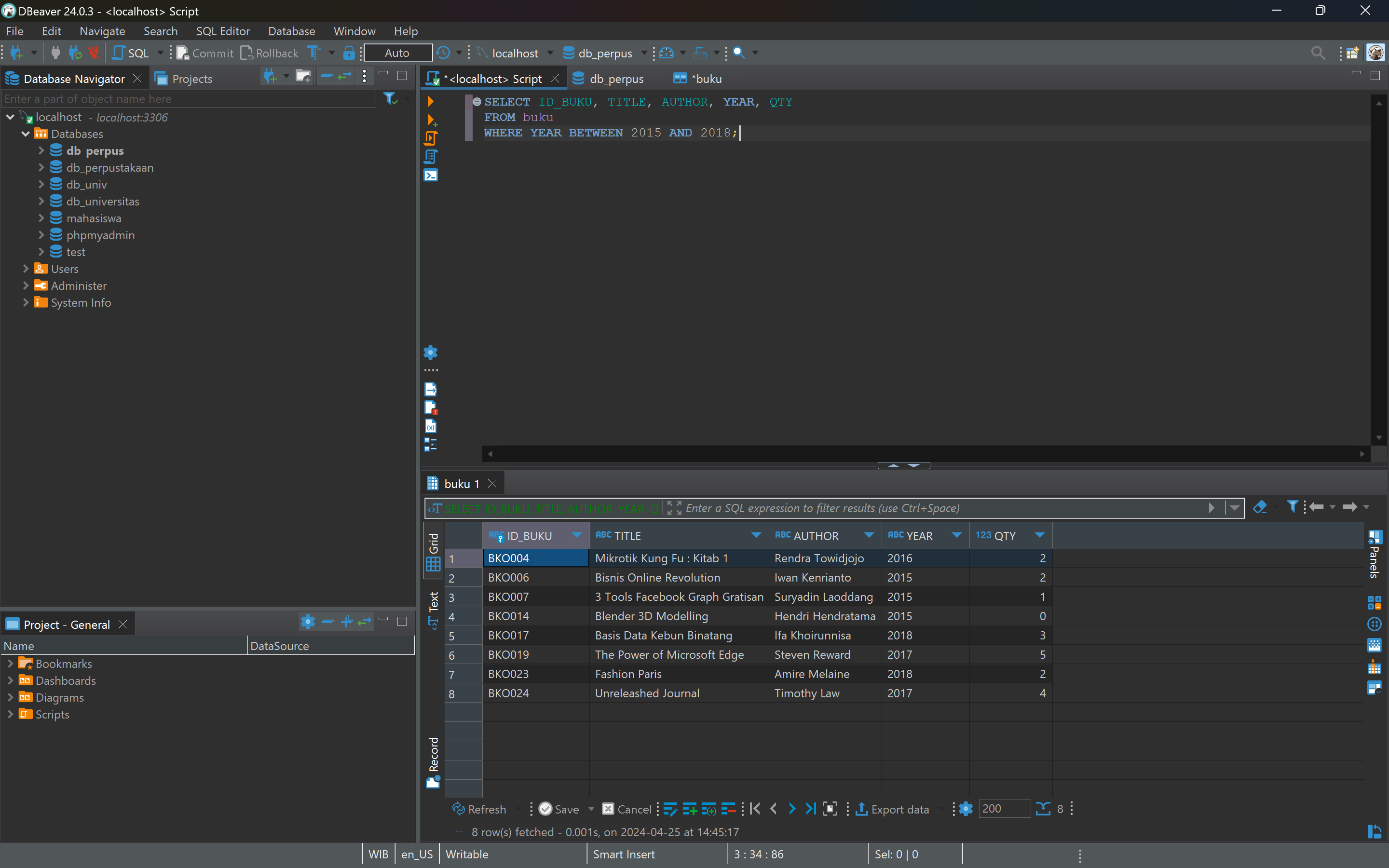
1. Perintah Menampilkan Record Jumlah Stok Buku Paling Sedikit



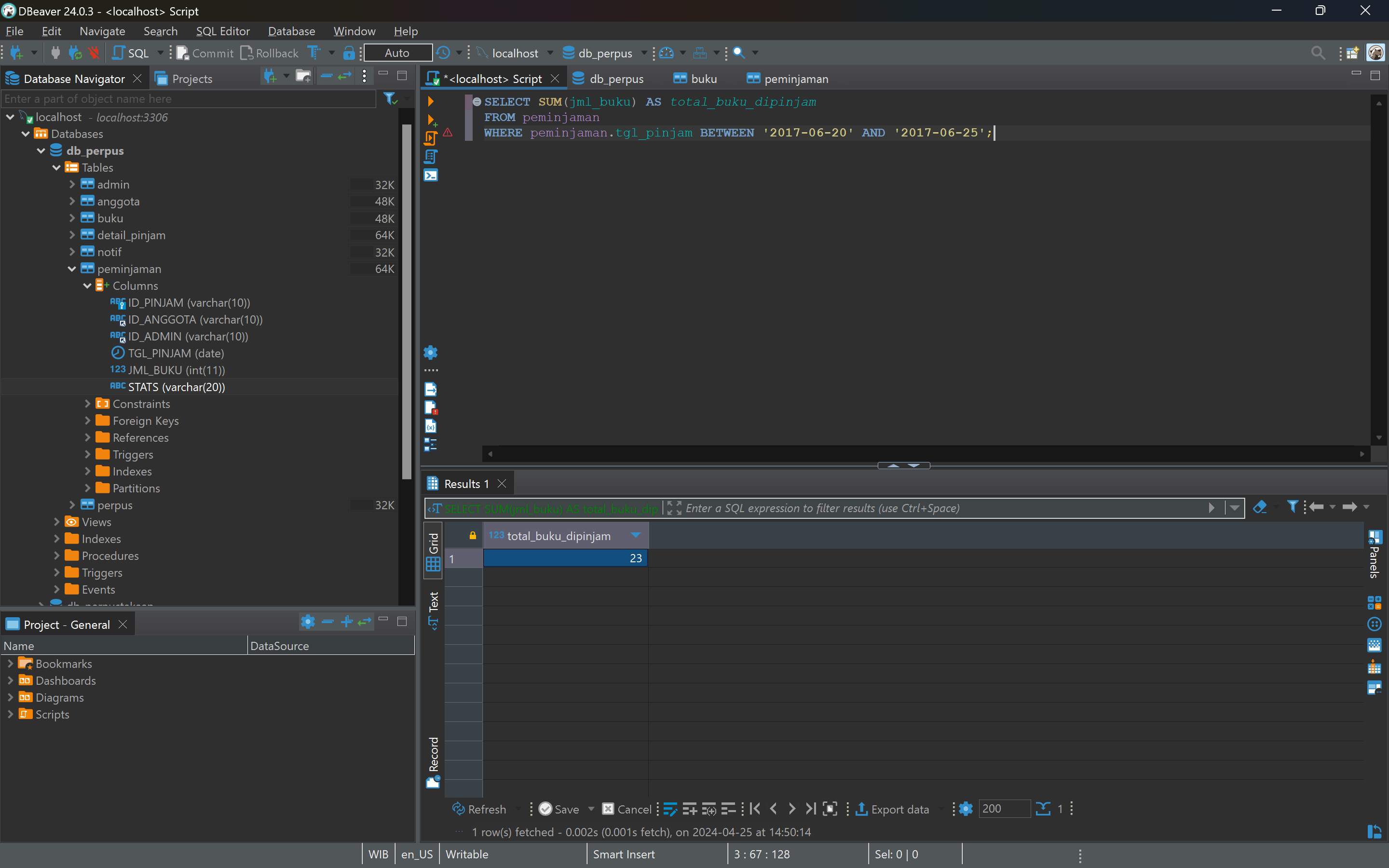
1. Perintah Menampilkan Jumlah Stok Buku dari Publisher

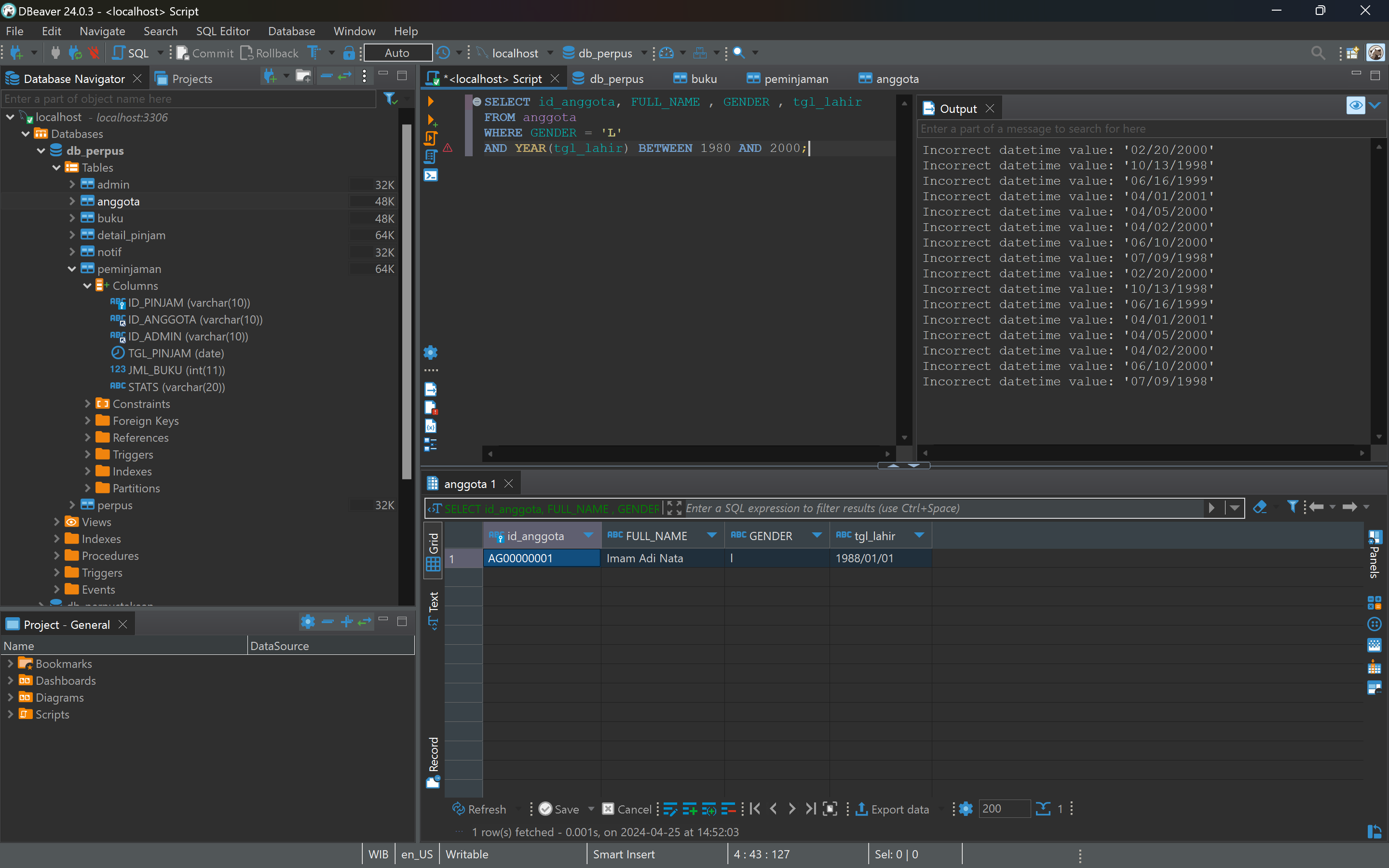


1. Menampilkan Buku yang diinginkan



1. Menampilkan Rata-Rata buku yang dipinjam pada Tahun Tertentu



1. Menampilkan Data Anggota dengan Gender Laki-Laki dan Tahun Kelahiran Tertentu  
   

## Kesimpulan

Praktikum ini menyediakan pengalaman praktis dalam mengelola basis data melalui Command Line Interface (CLI), yang merupakan keahlian penting dalam lingkungan kerja profesional. Melalui praktikum ini, kami mempelajari konsep dasar pengelolaan record dalam basis data, termasuk operasi CRUD yang melibatkan pembuatan, pembacaan, pembaruan, dan penghapusan data. Pentingnya struktur basis data relasional, di mana setiap record memiliki kunci primer unik dan dapat terhubung dengan record lain melalui kunci asing, juga ditekankan. Kami juga belajar untuk menulis perintah SQL untuk melakukan berbagai operasi, seperti pengambilan data berdasarkan kriteria tertentu, agregasi data, dan penggabungan kondisi dalam pengambilan data. Praktikum ini tidak hanya memberikan pemahaman tentang konsep teoritis pengelolaan basis data, tetapi juga mengajarkan penerapan praktis dalam kasus nyata, seperti pengelolaan perpustakaan. Dengan demikian, kami dapat mengembangkan keterampilan teknis dan konseptual yang diperlukan untuk bekerja efektif dalam pengelolaan dan penggunaan basis data dalam lingkungan profesional.

## 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*](http://atk.fam.free.fr/fichiers/stage/Python/JF/site/pytut.pdf)

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=1f2ee3831eebfc97 bfafd514ca2abb7e2c5c86bb*](https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=1f2ee3831eebfc97bfafd514ca2abb7e2c5c86bb)