LAPORAN TUGAS BESAR MANAJEMEN BASIS DATA

“ TUNING “



**Dosen Pengampu :**

Ahmad Luky Ramdani, S.Kom, M.Kom.

Disusun Oleh :

Yustika Ayu Putri Zalukhu (14117040)

Kelas RA

**PROGRAM STUDI TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI SUMATERA**

**2019**

# Daftar Isi

[Daftar Isi 2](#_Toc27982636)

[BAB I 3](#_Toc27982637)

[STUDI LITERATUR 3](#_Toc27982638)

[1.1 Tuning : Indexing 3](#_Toc27982639)

[1.2 Tuning :DBMS 3](#_Toc27982640)

[BAB II 6](#_Toc27982641)

[DESKRIPSI PERCOBAAN 6](#_Toc27982642)

[2.1 Melakukan Query tanpa Tuning 6](#_Toc27982643)

# BAB I

# STUDI LITERATUR

## Tuning : Indexing

Database Tuning adalah sejumlah aktifitas yang dilakukan untuk **memperbaiki atau meningkatkan kinerja atau performance** sebuah database. Aktifitas tuning ini meliputi banyak aspek dari software hingga hardware, antara lain I/O Tuning, DBMS Tuning, Query Tuning, dan Database Maintenance. Masing-masing memiliki tekniknya sendiri-sendiri, dan membutuhkan skill yang mumpuni. Namun kita tetap bisa mempelajari teknik-teknik dasarnya. Dalam artikel ini, kita akan mencoba melakukan Query Tuning dengan bantuan Database Index.

Umumnya programmer, pada saat mengembangkan sebuah aplikasi database, pasti akan membuat sebuah database dan memasukkan beberapa data contoh ke dalamnya. Pada saat pengembangan, tidak ada masalah dan kelihatannya semua baik-baik saja, namun seiring berjalannya waktu, aplikasi mulai terkesan lambat dan bisa jadi sampai hang. Hal ini terjadi karena biasanya programmer tidak pernah melakukan load testing menggunakan data besar, dalam artian, jumlah row yang buanyak.

## Tuning :DBMS

DBMS merupakan software yang digunakan untuk membangun suatu sistem basis data yang “sempurna”. DBMS harus dapat mengatur basis data tersebut sehingga dapat tersimpan dengan baik tanpa menimbulkan kekacauan, dapat dipakai oleh banyak user sesuai dengan kepentingan masing-masing, melindungi dari gangguan pihak-pihak yang tidak berwenang.  
Berdasarkan orientasi pemakainya, kita dapat mengelompokkan DBMS dalam 2 katagori, yaitu:  
1. DBMS yang berorientasi untuk satu atau sedikit pemakai  
MS-Access, dBase/Clipper, FoxBase, Borland-Paradox merupakan contoh-contoh DBMS yang lebih diorientasikan untuk satu pemakai dan karena itu dapat dengan mudah dipasang di komputer pribadi (Personal Computer/PC). Pada DBMS kelompok pertama ini pengembangannya terjadi pada aspek-aspek:

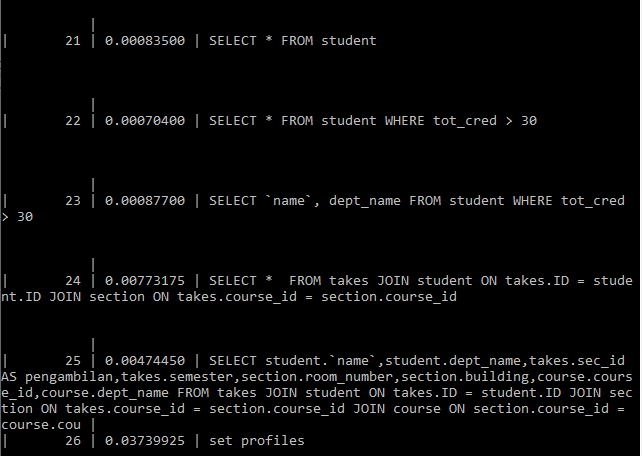
1. Jika awalnya, struktur tabel hanya mencakup pendefinisian nama field, tipe dan ukurannya, DBMS yang lebih baru juga memasukkan feature boleh tidaknya field dikosongkan, nilai awal (default), deksripsi field dan bentuk validasi (pendefinisian domain nilai) sebagai bagian dari struktur tabel.  
  
2. Tipe data yang dapat ditangani oleh DBMS terbaru sudah semakin banyak, seperti untuk mengakomodasi kebutuhan penyimpanan data teks yang panjang, teks berformat, gambar, data OLE, uang dan data autoincrement (yang nilainya bertambah otomatis)  
  
3. Bersama dengan komponen utamanya, DBMS kelompok ini juga seringkali dilengkapi dengan berbagai fasilitas tambahan untuk mempermudah pemakai dalam menggunakan DBMS seperti untuk pembuatan query, pembuatan laporan, pembuatan screen untuk berinteraksi dengan data, bahkan hingga pembuatan (men-generate) perinta makro basis data secara otomatis melalui pendifinisian menu dan tampilan layar.  
  
4. DBMS kelompok ini, karena memang lebih diorientasikan untuk pemakai tunggal, juga sering dimanfaatkan sebagai media pembangun aplikasi basis data, sehingga DBMS dan aplikasi basis data jadi menyatu bahkan aplikasi basis data jadi penyatu bahkan aplikasi basis data sendiri dianggap seabgai objek basis data sebagaimana tabel-tabel data yang kita gunakan untuk menyimpan data.  
Karena orientasi pemakai seperti itu, maka DBMS-DBMS kelompok ini, lemah dalam sejumlah aspek yang justru harus sangat diperhatikan pada kelompok DBMS yang kedua, seperti yang berkaitan dengan masalah pengamanan basis data, pemeliharaan basis data, mengendalikan persaingan pemakaian basis data dan lain-lain.  
  
2. DBMS yang berorientasi untuk banyak pemakai   
Oracle, Borland-Interface, MS-SQL Server, CA-OpenIngres, Sybase, Infomix, IBM-DB2 merupakan contoh-contoh DBMS yang lebih diorientasikan untuk banyak pemakai dan karena itu lebih ditunjukkan untuk pemakaian pada sistem jaringan komputer (LAN ataupun WAN). Tidak sebagaimana kelompok pertama, DBMS pada kelompok ini sangat tegas memisahkan fungsi pengelolaan basis data dan fungsi pembangunan aplikasi. Jika pada kelompok pertama, objek-objek yang dihasilkan oleh DBMS kelompok kedua bersifat sebaliknya. Transparansi hanya berlaku bagi DBMS yang bersangkutan, sehingga pemanfaatan objek-objek basis datanya hanya mungkin dilakukan dengan lebih dahulu mengaktifkan DBMS tersebut. Fungsi-fungsi pendukung (ultilitas) yang umumnya disatukan pada DBMS kelompok pertama, disediakan terpisah pada DBMS kelompok kedua ini, bukan saja karena fungsi-fungsi pendukung tersebut tidak relevan untuk selalu diaktifkan, tetapi juga karena fungsi pengelolaan yang ditangani DBMS kkelompok ini memang sudah sedemikian banyak dan jauh lebih penting. Pada seri-seri terbaru dari DBMS kelompok kedua ini, seperti juga d kelompok pertama, perluasan definisi struktur data dan pengkayaan tipe-tipe data baru juga diakomodasi. Perbedaan yang sangat mencolok di antara kedua kelompok DBMS terdapat pada lingkup fungsi pengelolaan basis data. Selain memiliki fungsi-fungsi standar (yang juga dimiliki oleh DBMS kelompok pertama) seperti pembentukan objek-objek basis data (tabel dan indeks), manipulasi data (penambahan, pengubahan, penghapusan data) dan pencarian data (query), fungsi pengelolaan DBMS kelompok kedua ini juga menangani aspek-aspek:  
  
1. Pengaman objek basis data terhadap akses pemakai yang tidak berhak (aspek security) dan bentuk-bentuk operasi yang tidak diperbolehkan (aspek integrity)  
  
2. Penanganan pemulihan data akibat kegagalan operasi basis data (aspek recovery), baik yang disebabkan oleh operasi-operasi basis data yang salah atau menimbulkan konflik, maupun yang disebabkan oleh fakor-faktor eksternal seperti mesin yang macet (crash), disk yang rusak atau terputusnya koneksi jaringan.  
  
3. Pembuatan data cadangan (aspek backup) yang dapat dilakukan secara incidental maupun periodic yang dapat dilakukan secara statis (dengan menonaktifkan pemakaian basis data) ataupun secara dinamis (tanpa menghalangi pemakaian basis data oleh para pemakai)  
  
4. Pengendalian persaingan pemakaian objek-objek basis data oleh banyak pemakai pada saat yang sama (aspek concurrency control) demi terjaminnya konsistensi data dan optimalisasi pemakaian setiap sumber daya mesin.  
  
5. Optimalisasi pengerjaan query (aspek query processing) yang diberikan oleh aplikasi pada server DBMS demi peningkatan performansi/ kecepatan pengerjaannya.  
Optimalisasi pemanfaatan sumber daya (aspek parallel processing/database) dengan memperhatikan optimalisasi pemakaian sumber daya mesin seperti processor, disk dan memori utama jika tersedia lebih dari satu dalam sebuah mesin.  
  
Banyak program basis data yang sudah di gunakan, misalnya : Ms Access, My SQL, Oracle, SQL Server, Postgre, Fire Bird, Paradok, Interbase. Dan contoh yang lainnya adalah Sybase, DB2, Informix, FoxPro, Clipper, dan lain-lain.  
Berikut fiktur dan standarisasi yang digunakan pada beberapa contoh DBMS:

# BAB II

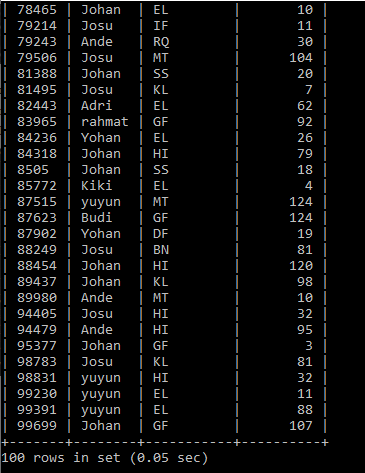
## DESKRIPSI PERCOBAAN

## 2.1 Melakukan Query tanpa Tuning

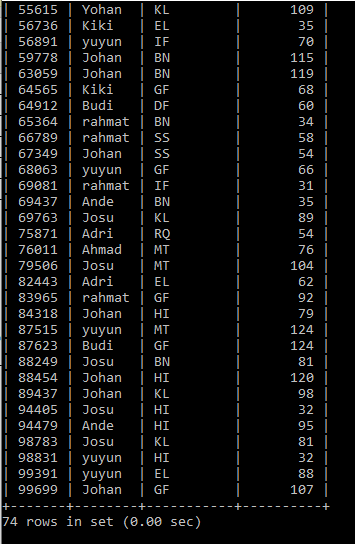
**1. Menggunakan Data 1**



* Dengan menggunakan Query SELECT \* FROM student



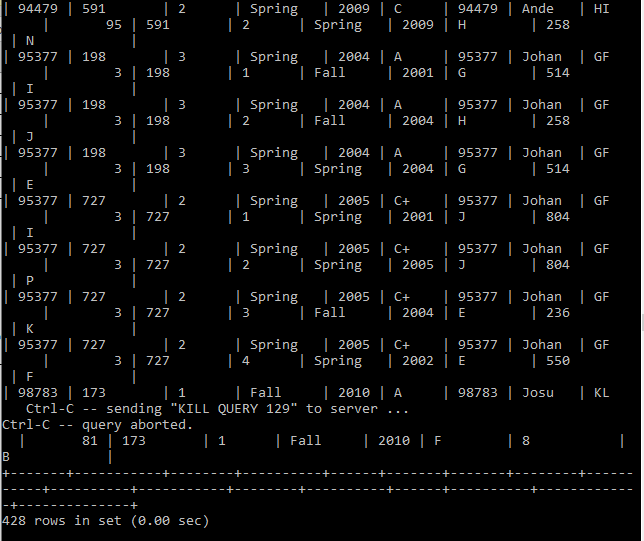
* Dengan menggunakan Query SELECT \* FROM student WHERE tot\_cred > 30;



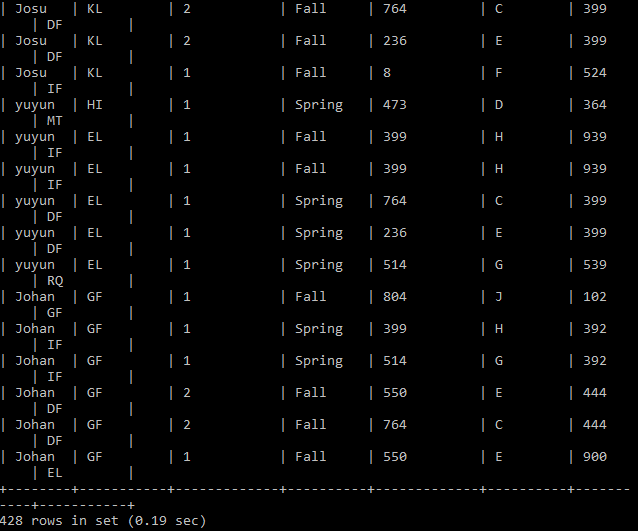
* Dengan menggunakan Query SELECT `name`, department FROM student WHERE tot\_cred > 30;



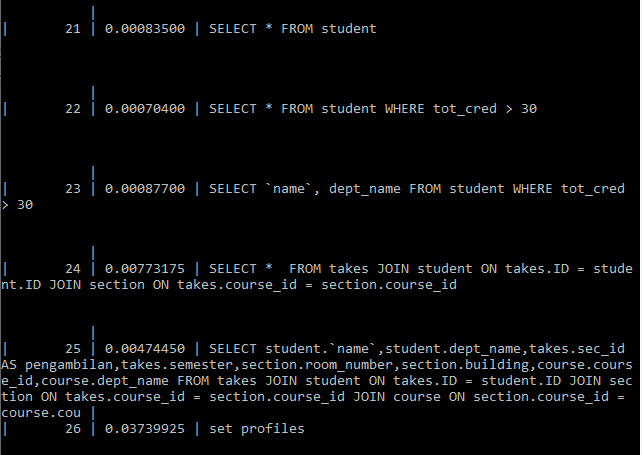
* Dengan menggunakan Query SELECT \* FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course\_id = section.course\_id



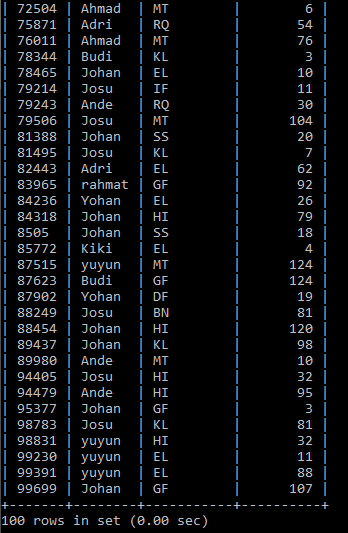
* Dengan menggunakan Query SELECT student. `name`,student. dept\_name,takes.sec\_id AS, pengambilan, takes.semester ,section. room\_number, section.building, course. course\_id, course.dept\_name FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course\_id = section.course\_id JOIN course ON section.course\_id = course.course\_id



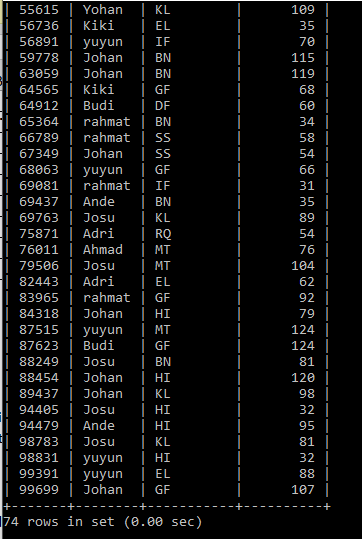
**2. Menggunakan Data 2**



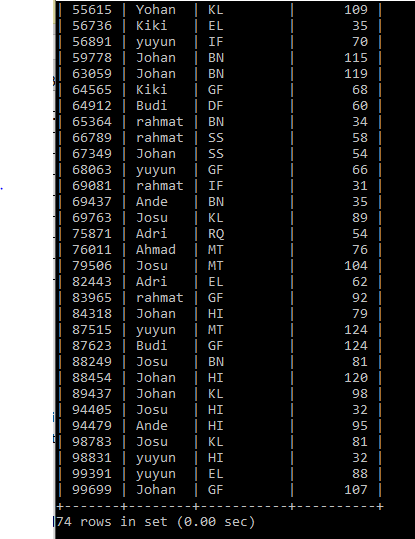
* Dengan menggunakan Query SELECT \* FROM student



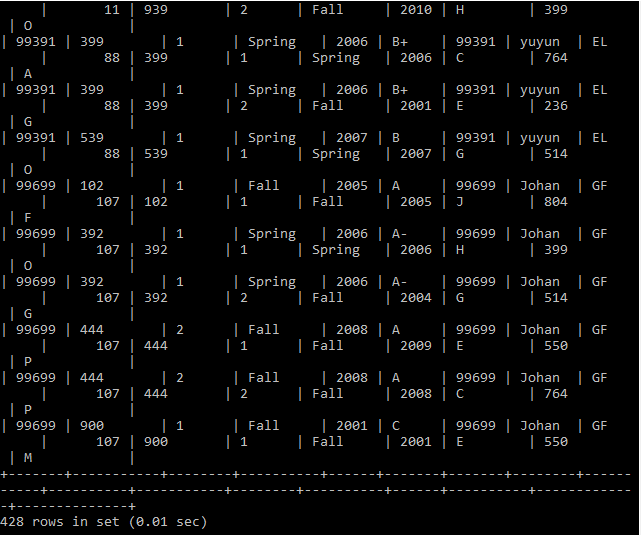
* Dengan menggunakan Query SELECT \* FROM student WHERE tot\_cred > 30;



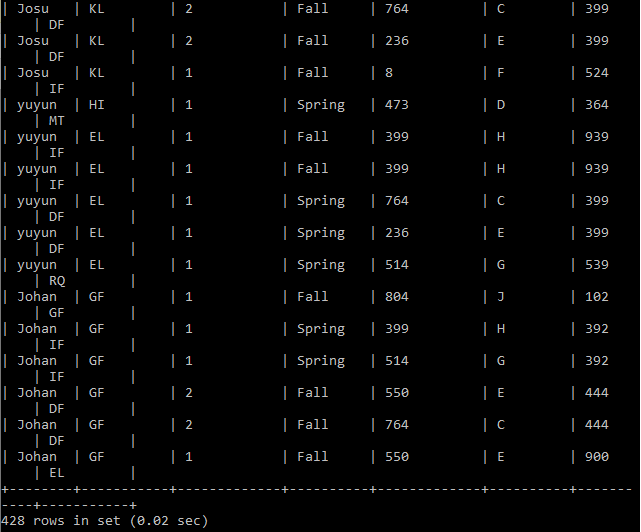
* Dengan menggunakan Query SELECT `name`, dept\_name FROM student WHERE tot\_cred > 30;



* Dengan menggunakan Query SELECT \* FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course\_id = section.course\_id



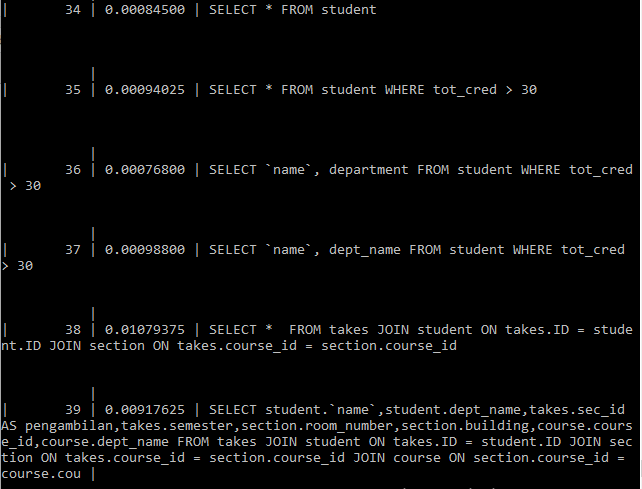
* Dengan menggunakan Query SELECT student. `name`,student. dept\_name,takes.sec\_id AS, pengambilan, takes.semester ,section. room\_number, section.building, course. course\_id, course.dept\_name FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course\_id = section.course\_id JOIN course ON section.course\_id = course.course\_id



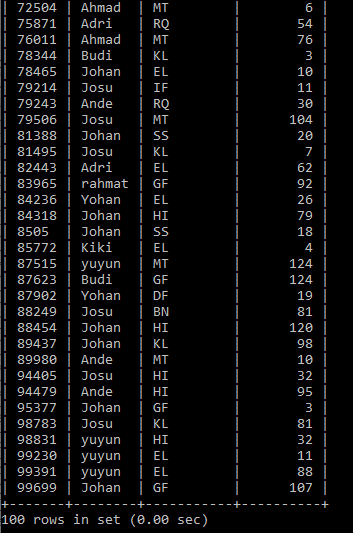
## 2.2 Tuning : Indexing

Percobaan yang dilakukan dengan mengitung waktu ekseskusi setiap query sebelum dilakukan tuning dan membandingkannya dengan setiap query yang telah di tuning.

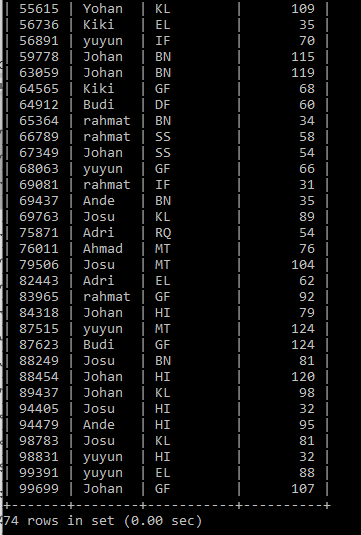
1. Menggunakan Data 1



* Dengan menggunakan Query SELECT \* FROM student



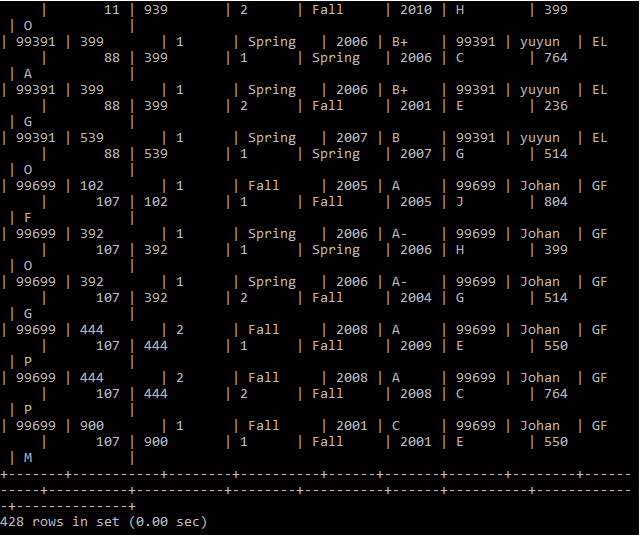
* Dengan menggunakan Query SELECT \* FROM student WHERE tot\_cred > 30;



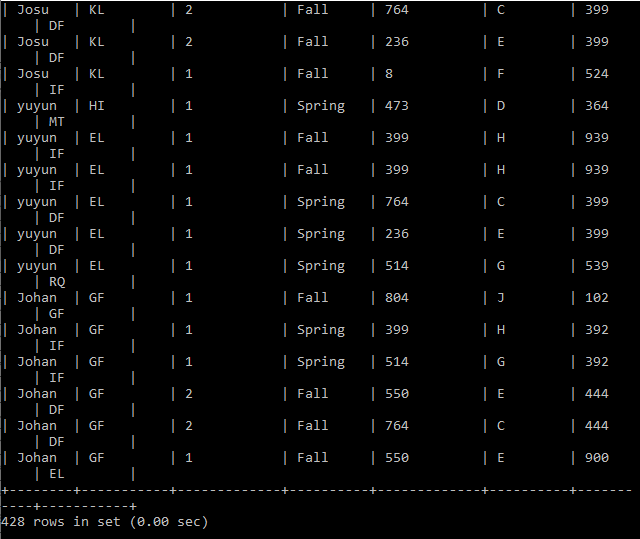
* Dengan menggunakan Query SELECT `name`, department FROM student WHERE tot\_cred > 30;



* Dengan menggunakan Query SELECT \* FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course\_id = section.course\_id

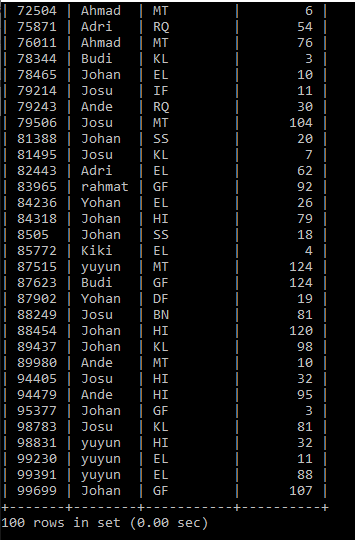


* Dengan menggunakan Query SELECT student. `name`, student.dept\_name, takes.sec\_id AS pengambilan, takes.semester, section.room\_number, section.building, course.course\_id,course.dept\_name FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course\_id = section.course\_id JOIN course ON section.course\_id = course.course\_id



1. **Menggunakan Data 2**

* Dengan menggunakan Query SELECT \* FROM student



* Dengan menggunakan Query SELECT \* FROM student WHERE tot\_cred > 30;



* Dengan menggunakan Query SELECT `name`, department FROM student WHERE tot\_cred > 30;



* Dengan menggunakan Query SELECT \* FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course\_id = section.course\_id



* Dengan menggunakan Query SELECT student.`name`,student.dept\_name,takes.sec\_id AS pengambilan,takes.semester,section.room\_number,section.building,course.course\_id,course.dept\_name FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course\_id = section.course\_id JOIN course ON section.course\_id = course.course\_id



## Daftar Pustaka

<https://prince-mienu.blogspot.com/2010/01/dbms-data-base-manajemen-sistem.html>

Cecilia, C., Mihai, G. (2011). Increasing Database Performance using Indexes, Database Systems Journal.