DigitalSkola
Uncover The World of Digital Skills with Us

Intermediate SQL

Advanced SQL

Version Control System

Learning Progress Review Week 4

By:
MARVEL TEAM
Fikrie | Natalia | Satria

1. *Intermediate SQL*

Limit | Distinct | Where | Order by | Aggregate Functions | Group by | Other SQL Functions

Apa itu LIMIT ?

LIMIT dalam SQL adalah untuk membatasi pengambilan jumlah Row Data. Biasanya digunakan pada awal pengambilan data dan ingin mengecek kolom apa saja yang terdapat pada tabel tersebut.

Mengecek 5 data pada tabel

SELECT …
FROM …
LIMIT n;

```
-- Limit
select nama_kolom
from nama_tabel
limit n;
```

{
untuk limit n bisa di ganti ke angka berapapun sesuai kebutuhan.
}

Contoh pengaplikasikan syntax :

```
select "order_id" , profit, quantity, total_profit
from batch_4.order_details_csv odc
limit 5;
```

Apa itu DISTINCT ?

DISTINCT dalam SQL adalah untuk eliminasi data yang sama tau duplikat dan akan menampilkan data yang unik saja.

Mencari nilai Unique

SELECT …
DISTINCT …
FROM …

```
-- Distinct
select
    distinct nama_kolom1, nama_kolom2
from nama_tabel;
```

{
bila di data suatu kolom terdapat nama atau angka yang sama bisa menggunakan fungsi ini.}

Contoh pengaplikasikan syntax :

```
select
      distinct category , sub_category
  from batch_4.order_details_csv odc ;
```

Apa itu WHERE ?

WHERE dalam SQL adalah digunakan untuk memfilter data berdasarkan kondisi tertentu.

Memfilter data

SELECT …
FROM …
WHERE …

```
-- Where
select nama_kolom
from nama_tabel
where konisi;
```

{

where digunakan Ketika ingin memilih spesifik data.

}

Contoh pengaplikasikan syntax :

```
select distinct category , sub_category
from batch_4.order_details_csv odc
where category = 'Furniture';
```

Apa itu Order By ?

ORDER BY dalam SQL adalah untuk mengurutkan resul-set dalam pengurutan ‘Ascending’ atau ‘Descending’.

Mengurutkan data

SELECT …
FROM …
GROUP BY …
ORDER BY …

```
-- order by
select nama_kolom1, nama_kolom2, nama_kolom3
from nama_tabel
group by nama_kolom1/2/3
order by nama_kolom1/2/3;
```

{
Kita dapat mengurutkan dari angka terbesar atau terkecil dengan fungsi ini.
}

Contoh pengaplikasikan syntax :

```
select
      sub_category
      , sum(quantity) as total_quantity_per_sub_category
  from batch_4.order_details_csv odc
  where category = 'Furniture'
  group by sub_category
  order by 2 desc ;
```

Apa itu Aggregate ?

Aggregate dalam SQL digunakan untuk melakukan perhitungan pada sekelompok nilai dan kemudian mengembalikkan nilai tunggal.

Perhitungan

SELECT …
SUM() …
FROM …

```
-- aggregate functions
select sum(nama_kolom)
from nama_tabel;
```

{
Fungsi ini untuk perhitungan
dari sum, avg, max, min, dll
}

Contoh pengaplikasikan syntax :

```
select sum(total_profit) as total_profit_all
from batch_4.order_details_csv odc ;
```

Apa itu GROUP BY ?

GROUP BY dalam SQL adalah untuk mengelompokkan data dalam sebuah kolom yang di tunjuk.

Penggabungan

SELECT …
FROM …
GROUP BY …

```
-- group by
select nama_kolom1, nama_kolom2, nama_kolom3
from nama_tabel
group by nama_kolom1/2/3
```

{
Fungsi ini untuk menggabungkan beberpa kolom menjadi satu tabel.
}

Contoh pengaplikasikan syntax :

```
select
      sub_category
      , sum(quantity) as total_quantity_per_sub_category
  from batch_4.order_details_csv odc
  group by sub_category ;
```

Fungsi SQL lain yang sering digunakan?

AND dalam SQL adalah untuk mengeluarkan output yang diinginkan. Semua kondisi harus bernilai untuk menghasilkan nilai benar.

OR dalam SQL jika salah satu kondisi pertama dan kedua benar, maka hasilnya pun akan muncul, jadi tidak mesti keduanya benar.

BETWEEN dalam SQL merupakan pernyataan kejadian di antara dua kriteria.

DigitalSkola
Uncover The World of Digital Skills with Us

2. *Advanced SQL*

Date Function | Join Tables | Sub Query

Date Function

DigitalSkola
Uncover The World of Digital Skills with Us

MEMANGGIL TANGGAL DAN WAKTU SAAT INI

*memanggil tanggal hari ini

```
select current_date
```

*memanggil tanggal & waktu saat ini (Jam:menit:detik)

```
select now()
```

*memanggil waktu saat ini

```
select current_time
```

*memanggil tanggal hari ini, waktu saat ini dan waktu serta tanggal saat ini

```
select current_date, now(), current_time
```

MENGAMBIL HANYA JAM, MENIT ATAU DETIK

*mengambil detik

```
select date_part('second', timestamp '2021-06-01, 07:19:35')
```

Date Function

DigitalSkola
Uncover The World of Digital Skills with Us

MENGGANTI ZONA WAKTU

*mengganti zona waktu sesuai tempat yang dipilih

```
select now() at time zone 'Asia/Jakarta'
```

*menyandingkan zona waktu yang berbeda

```
select now(), now() at time zone 'Asia/Jakarta'
```

Date Function

DigitalSkola
Uncover The World of Digital Skills with Us

MEMBUAT WAKTU DAN TANGGAL

*membuat tanggal

```
select make_date(2021, 5, 28)
```

*membuat waktu

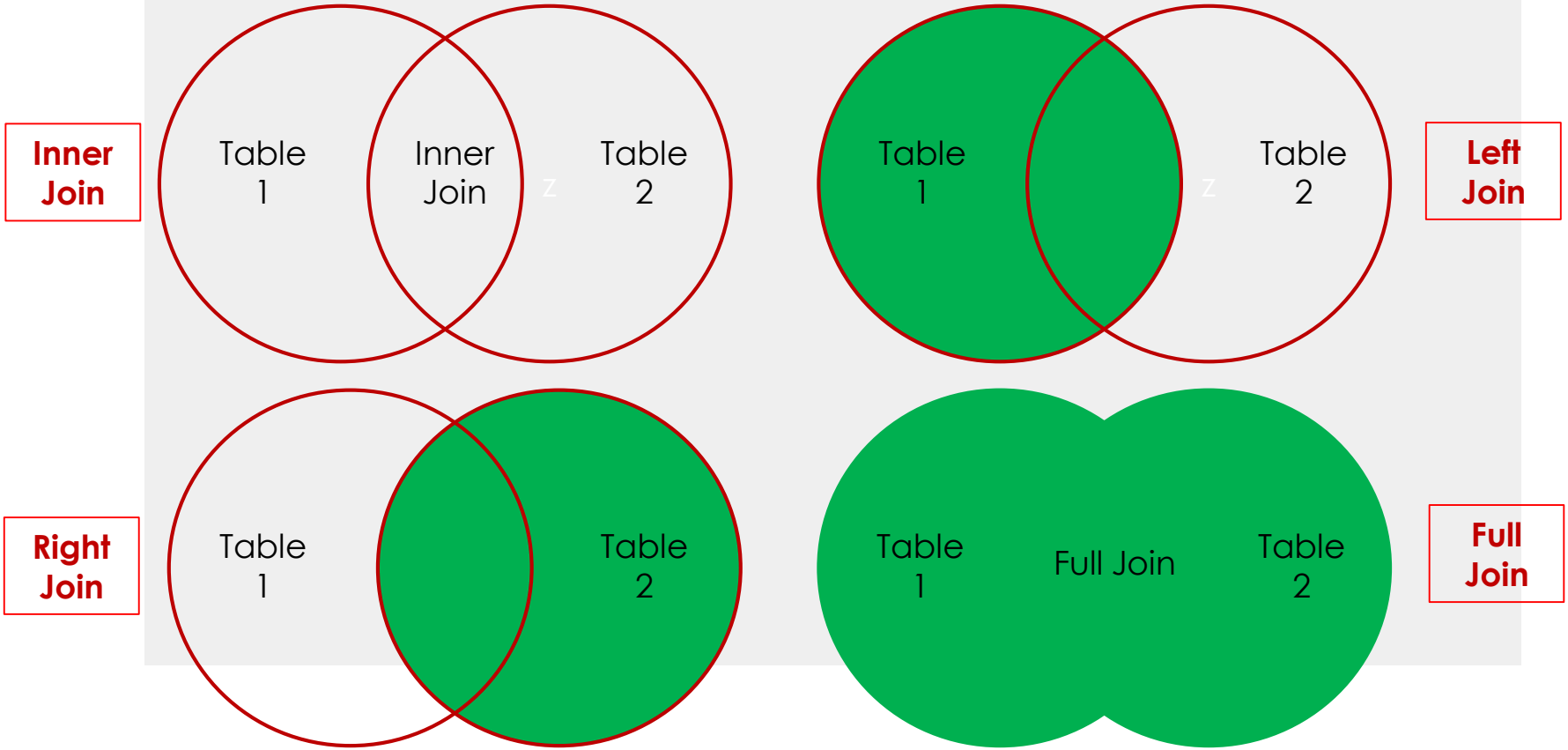
```
select make_time(15, 30, 2)
```

*membuat tanggal dan waktu

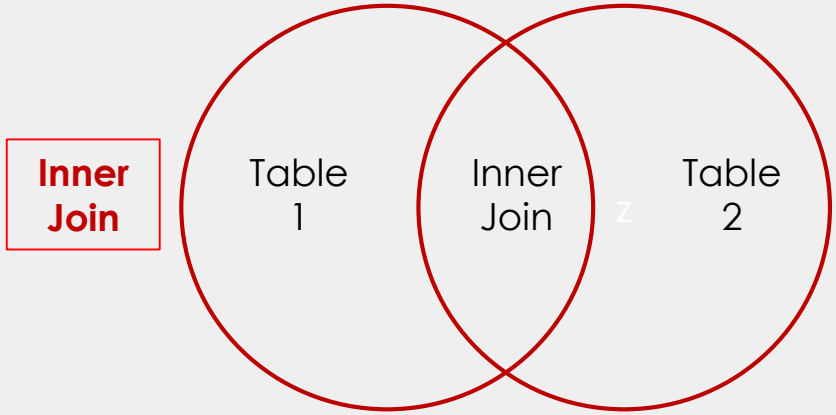
```
select make_timestamp(2021, 5, 28, 15, 30, 02)
```

Join Tables

DigitalSkola
Uncover The World of Digital Skills with Us



Join Tables

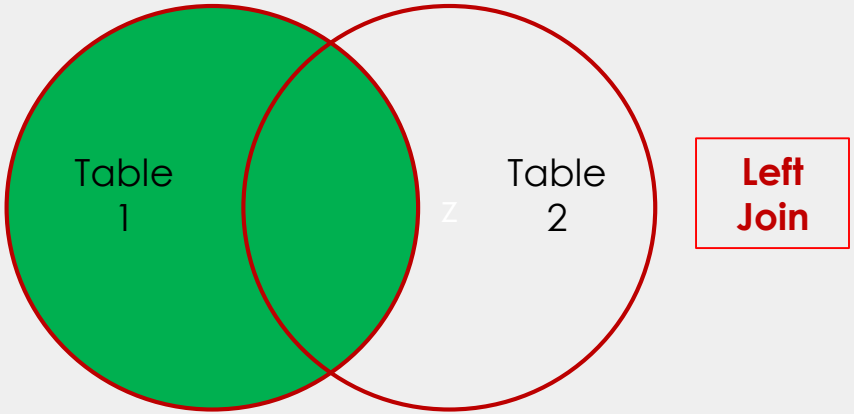


Mengambil data yang sama diantara dua table.

Contoh :

```
select
*
from datasource.employees a
inner join datasource.departments b on b.department_id = a.department_id
```

Join Tables

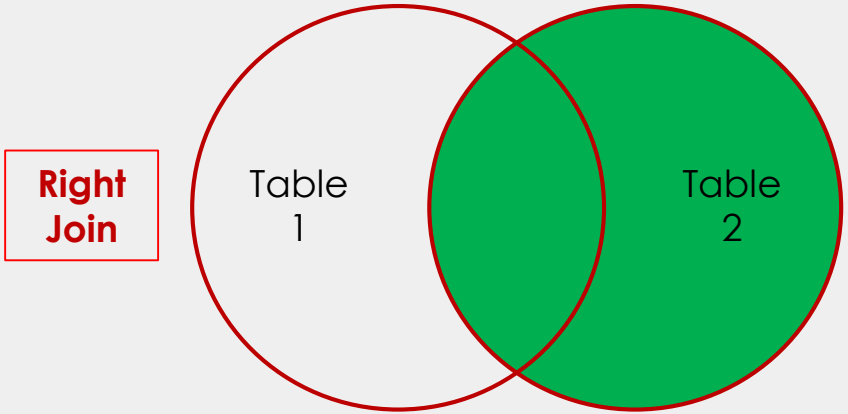


Mengambil keseluruhan data dari table pertama dan data yang sama dari table kedua.

Contoh :

```
select
*
from datasource.employees a
left join datasource.departments b on b.department_id = a.department_id
```

Join Tables

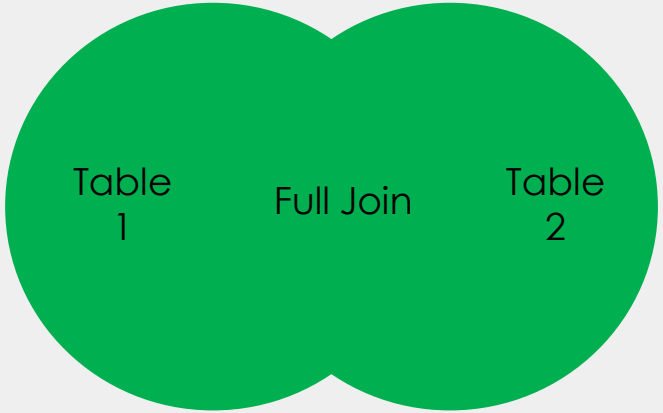


Mengambil keseluruhan data dari table kedua dan data yang sama dari table pertama.

Contoh :

```
select
*
from datasource.employees a
right join datasource.departments b on b.department_id = a.department_id
```

Join Tables



Full Join

Mengambil keseluruhan data dari table kedua dan table pertama serta data yang sama dari table keduanya.

Contoh :

```
select
*
from datasource.employees a
full join datasource.departments b on b.department_id = a.department_id
```

Sub Query

Mengolah data dari data yang sudah diolah.
Query di dalam **Query**.

Outside Query

Outside Query

```
select
    first_name
    , avg (salary) as total_salary
    from (
    select
    *
    from datasource.employees a
    full join datasource.departments b on b.department_id = a.department_id
    ) tabel1
group by 1
order by 2 asc
```

Inside Query

2. *Version Control System*

Database spesial untuk menyimpan *code*

Mengapa *Version Control Sistem* itu penting?

Version Control System berfungsi untuk :

- Kolaborasi dalam tim
- Memahami apa yang sedang terjadi pada perubahan *code*
- Teknologi *powerful* yang bisa mengerjakan proyek apapun seperti python, SQL, Java, Javascript
- Jika melakukan kesalahan, bisa kembali ke langkah sebelumnya
- Bisa membuat version
- Menjadi *back-up* atau tempat penyimpanan proyek

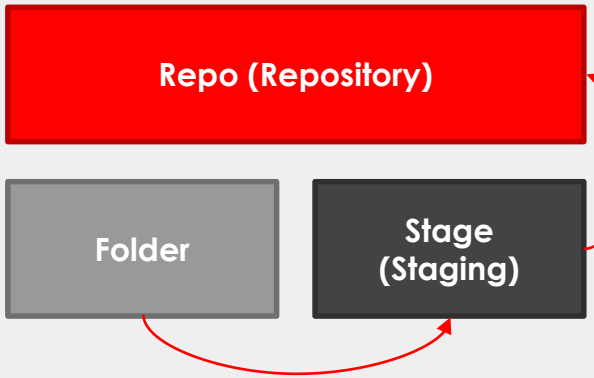
Apa itu Git?

Git adalah software distributed version control system gratis dan open-source yang dirancang untuk dapat menyimpan skala proyek besar dan kecil supaya kerjanya lebih efisien. Git dibuat oleh Linux.

Fitur-fitur Git yaitu :

- Dapat mencatat sejarah dari tiap file. Pada Git, kita membuat suatu langkah antara *Non-linear history* dan *Linear history*.
- Dapat di-*trace* (siapa yang mengerjakan, menghapus, mengubah)
- *Distributed system* adalah sistem dimana tiap orang memiliki repository yang sama dengan server
- Merging (menggabungkan pekerjaan tim) = menghubungkan cabang ke jalur utama
- *Branch workflow*

Bagaimana cara kerja Git?



git commit –m <message> (memasukkan dari stage ke dalam repo)

Pertama, folder yang sudah dibuat dimasukkan ke dalam stage

git add <file> (memasukkan satu file ke stage)
git add * (memasukkan semua file ke stage)

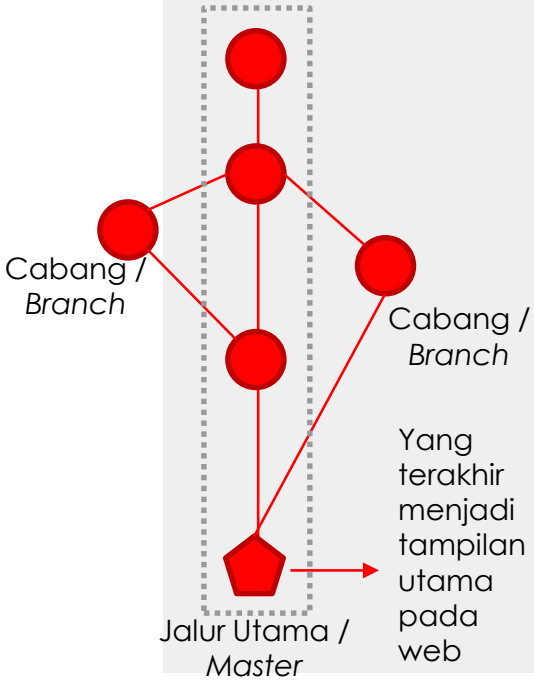
Git Configuration and Repo Commands

Perintah (*Commands*)	**Fungsi**
git –version	Mengecek versi Git
git config –global user.name “Name Name”	Konfigurasi *username* Git
git config –global user.email “email@email.com”	Konfigurasi *email* Git
git config user.name / user.email	Mengecek *user name* atau *email*
git init	Membuat *repository* (sebelumnya membuat folder dahulu). Ketika ada tulisan master, folder yang dibuat resmi menjadi *repository*.
git status	Mengecek status
git log	Melihat *history* yang dibuat
git log –p	Melihat *history* lebih detail. Tekan “q” untuk keluar.

Git Execution Commands

Perintah (*Commands*)	**Fungsi**
clear	Menghapus tulisan sebelumnya. Untuk melihat *command history* menggunakan tanda panah atas bawah di keyboard.
git add “filename”	Menyimpan satu file ke dalam *staging*
git add *	Menyimpan seluruh file ke dalam *staging*
git rm –cached “filename”	Menghapus satu file di dalam *staging*
git rm –cached *	Menghapus seluruh file di dalam *staging*
git commit –m “message”	Memasukkan atau meng-*commit* file ke dalam Repo. Cara menulis *message* yang baik adalah menggunakan kata kerja dahulu dan tidak terlalu panjang. Jika sudah di-*commit*, tidak bisa diubah lagi, hanya bisa dihapus.
git commit –a –m “message”	Memasukkan atau meng-*commit* seluruh file ke dalam Repo.

Git Branching Commands



Perintah (*Commands*)	**Fungsi**
git checkout –b "new_branch"	Membuat cabang. Isi *Master* dan *Branch* berbeda.
git branch	Mengecek seluruh *branch* dan yang aktif
git switch "new_branch"	Pindah ke *branch* lain
git merge "new_branch"	Menghubungkan cabang / *branch* ke jalur utama. Sebelumnya *switch* dulu ke *Master*, kemudian *Merge* cabang yang sudah dibuat.
git reset –hard HEAD~1	Jika terlanjur sampai *commit*, pilih *hard* dan *mixed* untuk kembali ke awal. Posisi paling depan Namanya HEAD ~1, seterusnya ke belakang.
git reset –mixed	
git reset –soft	Jika masih dalam tahap *staging*, gunakan *command soft* ini untuk kembali ke awal.
git branch –d "branch name"	Menghapus cabang

Apa itu Git Hub?

Github adalah *repository* yang disimpan dalam *server*.

Github itu penting karena dapat mengakses berbagai data yang disimpan dari bermacam laptop dan berkontribusi ke *community*.

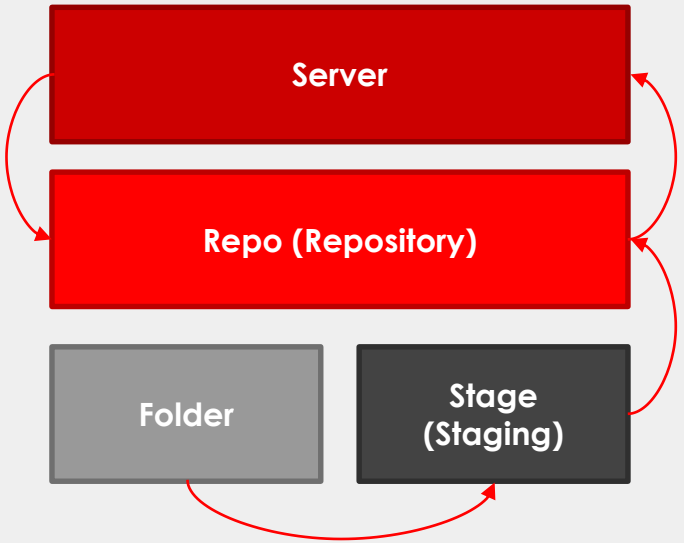
Github ada yang gratis (*public*) dan berbayar (*private*).

Github dibuat oleh suatu perusahaan yang menyediakan server untuk menyimpan *repository*.

Skema Kerja Github

DigitalSkola
Uncover The World of Digital Skills with Us

git pull
Digunakan andaikan yang di server harus di-upgrade tapi yang di repo lokal telat, maka kita harus menyesuaikannya dengan menariknya.



git push
(memasukkan repo lokal ke dalam server)

git commit –m <message>
(memasukkan dari stage ke dalam repo)

Pertama, folder yang sudah dibuat dimasukkan ke dalam stage

git add <file> (memasukkan satu file ke stage)
git add * (memasukkan semua file ke stage)

Create Github Account and .gitignore

- Masuk ke *website github.com, sign up*. Setelah buat, masuk (*log in*) ke dalam *github*. Lalu, membuat repo di *server* lalu masukkan ke lokal.
- Ketik nama repo yang diinginkan. Pilih Public atau Private.
- Centang Add a README file dan Add.gitignore (gitignore itu adalah file dan jangan lupa pilih file yang ingin di-*ignore* / diabaikan).
- Untuk meng-*exclude* file yang tidak diinginkan, dapat menggunakan file ***gitignore*** ini. Contoh: jika ingin abaikan file gambar, musik dan video di dalam file gitignore ketik:

  # Ignore images

  *.jpeg

  *.jpg

  *.png

- Cara membuat file gitignore sendiri:

  1. Buat notepad, lalu Save As ke dalam folder Repo yang dari server
  2. Simpan dengan nama .gitignore
  3. Ketik nama-nama file yang ingin diabaikan

Github Commands

Perintah (*Commands*)	**Fungsi**
git clone <link>	*Download repository* yang di *server* ke lokal (komputer sendiri)
git remote set-url origin <link>	Menghubungkan repo yang di lokal dengan yang di *server*
git push	Memasukkan repo lokal ke dalam *server*. Lalu, *sign in* ke dalam *github*.
git pull	Menarik repo yang di *server* ke repo lokal

Special Thanks to :

Slide template by SlideCarnival