

Basis Data dan Penelusuran Data

Bakti Siregar, M.Sc

2023-08-26

Contents

Kata Pengantar	5
Ringkasan Materi	5
Penulis	6
Asisten Lab	6
Ucapan Terima Kasih	7
Masukan & Saran	7
1 Pendahuluan	9
1.1 Apa itu SBD?	10
1.2 Mengapa R & SQL?	14
1.3 MySQL vs PostgreSQL	15
1.4 Instalasi MySQL (XAMPP)	18
1.5 Instalasi PostgreSQL	24
1.6 Praktikum	36
2 Connecting R to SQL	41
2.1 Introduction	41
2.2 Connecting R to SQL	41
2.3 Import Data	44
2.4 Write Dataframe to Database	45
2.5 Basic Queries	46
2.6 Your Job	48
3 Referensi	49

Kata Pengantar

Selamat datang dalam modul praktikum mengenai basis data dan penelusuran data. Dalam era digital yang semakin maju, pengelolaan informasi dan akses terhadap data sangatlah penting. Basis data merupakan fondasi utama dalam pengelolaan data yang efisien dan terstruktur, sedangkan penelusuran data memungkinkan kita untuk menggali wawasan berharga dari kumpulan informasi yang tersedia. Dalam modul ini, kita akan menjelajahi konsep-konsep dasar dalam basis data, termasuk jenis-jenis basis data, model data, bahasa kueri, dan praktik terbaik dalam merancang basis data yang optimal. Secara khusus, modul ini

Selain itu, penelusuran basis data yang menjadi fokus penting adalah menggunakan R Programming dan SQL dalam membuat data analytics system. Penelusuran data melibatkan teknik-teknik dan alat-alat untuk menggali informasi yang berharga dari kumpulan data yang besar dan kompleks. Dengan adanya kemajuan dalam analisis data dan kecerdasan buatan, penelusuran data telah menjadi aspek penting dalam pengambilan keputusan dan inovasi. Penulis berharap bimbingan ini akan memberikan pemahaman yang kokoh tentang basis data dan penelusuran data, serta memberi Anda wawasan yang berguna dalam mengelola data dan mengambil informasi berharga dari sumber daya yang ada. Selamat belajar!

Ringkasan Materi

Adapun isi pembelajaran dalam modul ini adalah sebagai berikut:

- Bab 1
- Bab 2
- Bab 3
- Dst

Penulis

- **Bakti Siregar, M.Sc** adalah Ketua Program Studi di Jurusan Statistika Universitas Matana. Lulusan Magister Matematika Terapan dari National Sun Yat Sen University, Taiwan. Beliau juga merupakan dosen dan konsultan Data Scientist di perusahaan-perusahaan ternama seperti JNE, Samora Group, Pertamina, dan lainnya. Beliau memiliki antusiasme khusus dalam mengajar Big Data Analytics, Machine Learning, Optimisasi, dan Analisis Time Series di bidang keuangan dan investasi. Keahliannya juga terlihat dalam penggunaan bahasa pemrograman Statistik seperti R Studio dan Python. Beliau mengaplikasikan sistem basis data MySQL/NoSQL dalam pembelajaran manajemen data, serta mahir dalam menggunakan tools Big Data seperti Spark dan Hadoop. Beberapa project beliau dapat dilihat di link berikut: Rpubs, Github, Website, dan Kaggle.

Asisten Lab

- **Yonathan Anggraiwan, S.Stat** adalah seorang alumni Statistika yang bersemangat dalam dunia pemrograman dan analisis data. Lahir di Tangerang, minatnya terhadap teknologi dan komputer muncul sejak usia dini. Ia tumbuh dengan rasa ingin tahu yang kuat terhadap bahasa pemrograman, dan ini membawanya menuju dunia analisis data menggunakan bahasa pemrograman R dan Python. Selama menjalankan tugas sebagai asisten lab, Yonathan Anggraiwan berperan dalam membantu mahasiswa dalam memahami konsep-konsep dasar dan kompleks dalam pemrograman R dan Python. Ia memberikan penjelasan yang jelas dan dukungan kepada mahasiswa yang mengalami kesulitan. Selain itu, ia juga terlibat dalam merancang tugas dan ujian praktikum, serta memberikan umpan balik konstruktif kepada para mahasiswa. Dalam perjalanan waktu, Yonathan Anggraiwan mulai mengambil tanggung jawab lebih besar dalam laboratorium. Ia membantu mengembangkan materi pembelajaran tambahan, seperti tutorial online tentang analisis data menggunakan R dan Python. Ia juga aktif dalam berbagai proyek penelitian di bawah bimbingan dosen, yang melibatkan pengolahan data besar untuk analisis statistik dan visualisasi. Dengan semangat yang tinggi, dedikasi, dan keterampilan yang dimilikinya, Yonathan Anggraiwan adalah contoh nyata dari seorang mahasiswa yang berhasil menggabungkan minatnya dalam pemrograman R dan Python dengan peran yang produktif sebagai asisten laboratorium dan kontributor dalam dunia analisis data.

Ucapan Terima Kasih

Saya ingin mengucapkan terima kasih yang tulus kepada semua yang telah mendukung dan berkontribusi dalam perjalanan pembuatan modul “Basis Data dan Penelusuran Data”. Modul ini tidak akan mungkin menjadi kenyataan tanpa kerja keras, semangat, dan dukungan yang luar biasa dari berbagai pihak. Terima kasih juga kepada rekan-rekan dan kolega yang telah memberikan masukan, saran, dan diskusi berharga sepanjang perjalanan penulisan modul ini. Kontribusi kalian telah membantu memperkaya isi modul dan menghadirkan sudut pandang yang beragam. Tentu saja, modul ini tidak akan lengkap tanpa rasa terima kasih kepada para peneliti dan praktisi di bidang basis data dan penelusuran data yang telah menciptakan landasan pengetahuan yang menjadi dasar dari modul ini. Pengalaman dan pengetahuan yang kalian bagikan sangat berharga. Saya juga ingin mengucapkan terima kasih kepada keluarga dan teman-teman saya atas dukungan, pengertian, dan dorongan yang tak henti-hentinya. Tanpa dukungan kalian, perjalanan menulis modul ini pastinya tidak akan semudah ini.

Akhir kata, semoga modul ini dapat memberikan manfaat dan wawasan baru kepada para pembaca yang ingin mendalami dunia basis data dan penelusuran data. Ucapan terima kasih terakhir saya tujuhan untuk semua yang telah berkontribusi, baik secara langsung maupun tidak langsung, dalam menghadirkan modul ini kepada para pembaca.

Masukan & Saran

Semua masukan dan tanggapan Anda sangat berarti bagi kami untuk memperbaiki template ini kedepannya. Bagi para pembaca/pengguna yang ingin menyampaikan masukan dan tanggapan, dipersilahkan melalui kontak dibawah ini!

Email: dscienclabs@outlook.com

Chapter 1

Pendahuluan

Sejak tahun 1970, **Structured Query Language (SQL)** telah digunakan oleh para programmer untuk membangun dan mengakses **Sistem Basis Data (SBD)**. Banyak sekali perdebatan mengenai cara penyebutan SQL ini, namun pada kenyataannya, kita dapat melafalkannya sebagai “sequel” ataupun “S.Q.L”. Mempelajari bahasa pemrograman umum seperti R adalah penting dan akan lebih baik jika memiliki kemampuan SQL dalam bidang pengolahan data.

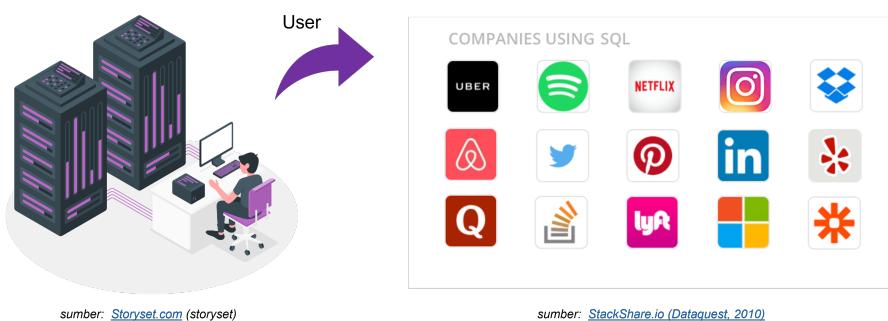


Figure 1.1: Beberapa Perusahaan Besar Pengguna SQL

Banyak perusahaan besar di bidang teknologi menggunakan SQL seperti Uber, Netflix, dan Airbnb. Bahkan dalam perusahaan seperti Facebook, Google dan Amazon, yang telah membuat sendiri **SBD** berkemampuan tinggi, tetap menggunakan SQL untuk melakukan query dan analisis data.

1.1 Apa itu SBD?

Secara umum **SBD** dapat didefinisikan sebagai berikut:

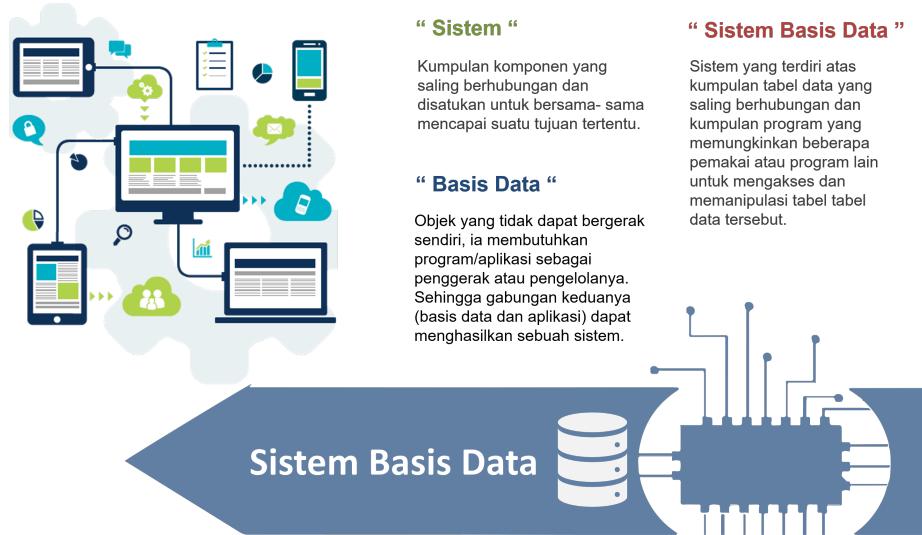


Figure 1.2: Definisi Sistem Basis Data

1.1.1 Komponen SBD

Adapun beberapa komponen dasar yang diperlukan dalam SBD adalah:

1.1.2 Manfaat SBD

Manfaat atau kegunaan penerapan SBD cukup banyak dan cakupannya pun luas dalam mendukung keberadaan lembaga atau organisasi maupun perusahaan, diantaranya:

1.1.3 Definisi SQL vs NoSQL

Sebenarnya perbedaan antara SQL dan NoSQL secara mendasar sudah dapat dijelaskan dari akronimnya.

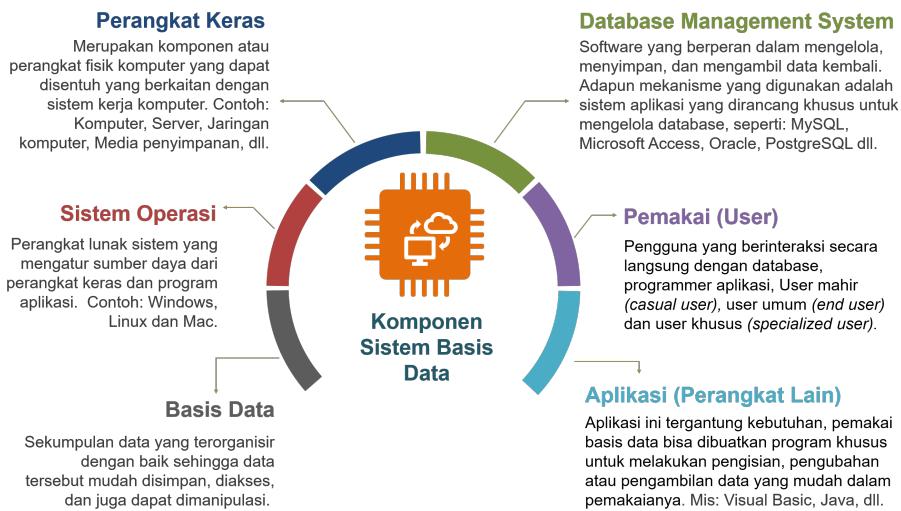


Figure 1.3: Komponen SBD



Figure 1.4: Manfaat SBD

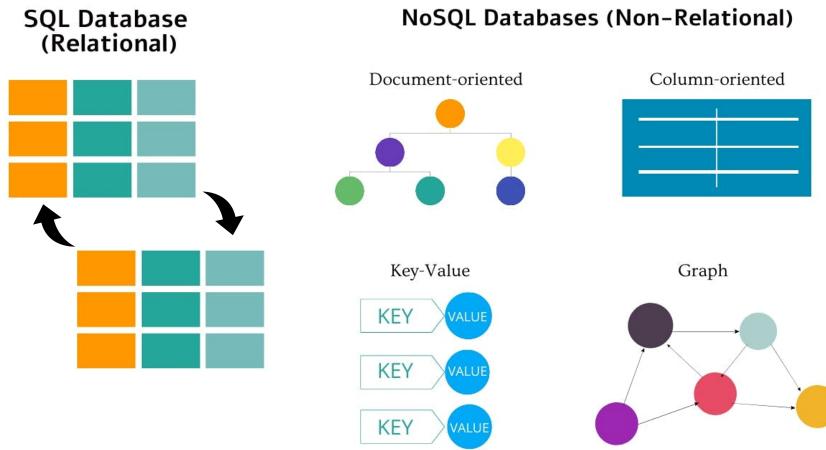


Figure 1.5: SQL vs NoSQL

SQL basis data relasional yang menggunakan ‘relasi’ (yang biasanya disebut tabel) untuk menyimpan data dan mencocokkan data tersebut dengan memakai karakteristik umum di setiap dataset. Sedangkan, *NoSQL* adalah database yang menggunakan format JSON untuk setiap dokumennya sehingga mudah dibaca dan dimengerti. NoSQL banyak diminati karena memiliki performa yang tinggi dan bersifat non-relasional sehingga dapat memakai berbagai model data.

1.1.4 Perbedaan SQL vs NoSQL

Sebenarnya banyak perbedaan yang dimiliki di antara dua database tersebut tapi inilah perbedaan yang paling mencolok antara SQL dan NoSQL:

1.1.5 Top 7 SQL

Tercatat sampai bulan Februari 2020 ada 334 jenis database menurut db-engines.com. Berikut ini saya merangkum daftar 7 database terpopuler yang menggunakan SQL (Relasional):

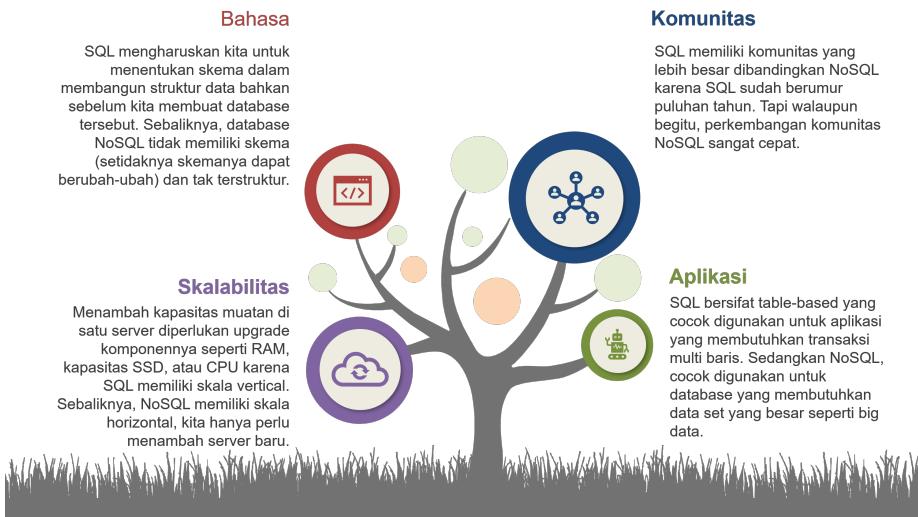


Figure 1.6: Perbedaan SQL vs NoSQL

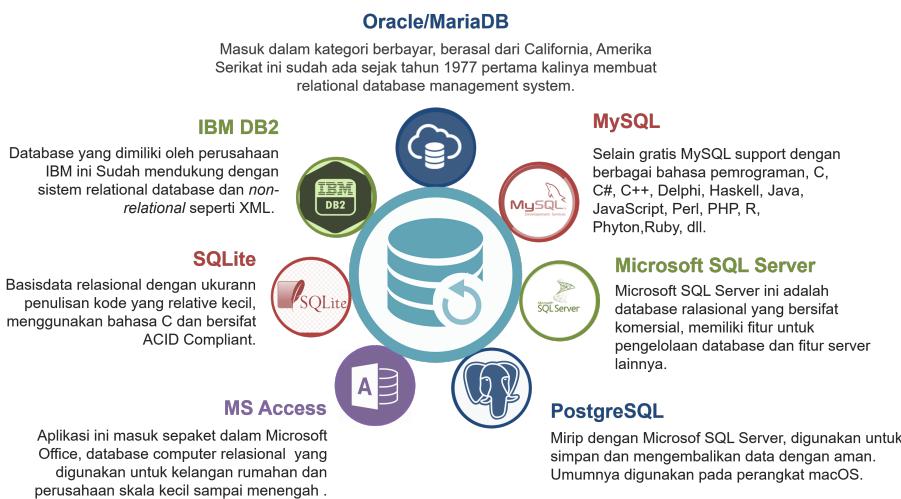


Figure 1.7: Top 7 Perangkat Lunak SQL

1.1.6 Top 8 NoSQL

Kebanyakan basis data NoSQL digunakan dalam dunia aplikasi web waktu nyata (real-time web app). Berikut ini adalah ulasan 8 jenis basis data NoSQL yang paling populer digunakan diseluruh dunia:



Figure 1.8: Top 8 Perangkat Lunak NoSQL

1.2 Mengapa R & SQL?

Menggunakan R dan SQL merupakan kombinasi yang kuat untuk analisis data dan pengelolaan basis data. Keduanya memiliki peran yang berbeda dalam proses analisis dan pengelolaan data. Berikut adalah beberapa alasan mengapa menggunakan R dan SQL bersama:

- **Kekuatan Analisis R**

R adalah bahasa pemrograman yang khusus dirancang untuk analisis statistik dan visualisasi data. R memiliki berbagai paket (packages) yang menawarkan fungsi statistik dan analisis yang kuat, termasuk regresi, pengelompokan, analisis deret waktu, dan banyak lagi. Visualisasi yang dapat dihasilkan dengan R sangat bervariasi, dari grafik sederhana hingga visualisasi interaktif yang kompleks.

- **Manipulasi dan Pengelolaan Data dengan SQL**

SQL digunakan untuk mengelola dan mengambil data dari basis data terstruktur. SQL menyediakan cara efisien untuk membuat, mengubah, menghapus, dan memanipulasi data dalam basis data. SQL memiliki fitur untuk menggabungkan data dari berbagai tabel, melakukan agregasi, dan menyaring data.

- **Integrasi Antara R dan SQL**

Banyak perpustakaan R yang mendukung koneksi ke basis data menggunakan SQL. Anda dapat menggunakan perintah SQL dalam skrip R untuk mengambil data dari basis data, memanipulasi data di dalam R, dan kemudian menerapkan analisis statistik menggunakan paket R. Integrasi ini memungkinkan Anda menggabungkan kekuatan analisis statistik R dengan kemampuan pengelolaan data SQL.

- **Skalabilitas dan Efisiensi**

Menggunakan SQL untuk mengambil dan memanipulasi data dalam basis data bisa lebih efisien daripada melakukannya dalam R, terutama untuk dataset besar. SQL memungkinkan query yang dioptimalkan dan penggunaan indeks untuk kinerja yang lebih baik.

- **Data Preprocessing**

Sebelum menerapkan analisis di R, Anda mungkin perlu melakukan prapemrosesan pada data, seperti membersihkan data, menggabungkan tabel, dan mengisi data yang hilang. SQL dapat membantu dalam melakukan tugas-tugas ini.

Jadi, menggunakan R dan SQL bersama memungkinkan Anda menggabungkan kekuatan analisis statistik R dengan kemampuan pengelolaan data SQL. Ini bisa sangat berguna ketika Anda ingin melakukan analisis data yang luas dan kompleks dari berbagai sumber data yang berbeda.

1.3 MySQL vs PostgreSQL

MySQL adalah sistem manajemen basis data relasional yang memungkinkan Anda untuk menyimpan data sebagai tabel dengan baris dan kolom. Sistem ini populer sehingga digunakan di banyak aplikasi web, situs web dinamis, dan sistem tertanam. PostgreSQL adalah sistem manajemen basis data relasional-objek yang menawarkan lebih banyak fitur daripada MySQL. Sistem ini memberi Anda lebih banyak fleksibilitas dalam tipe data, skalabilitas, konkurensi, dan integrasi data.

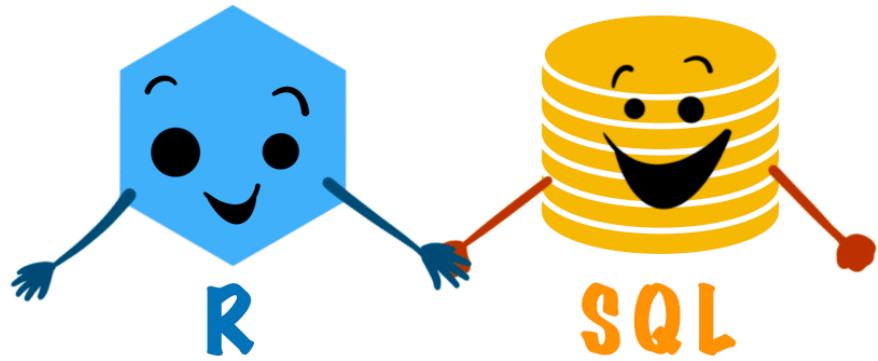


Figure 1.9: R dan SQL [<https://irene.rbind.io/>](<https://irene.rbind.io/post/using-sql-in-rstudio/>)

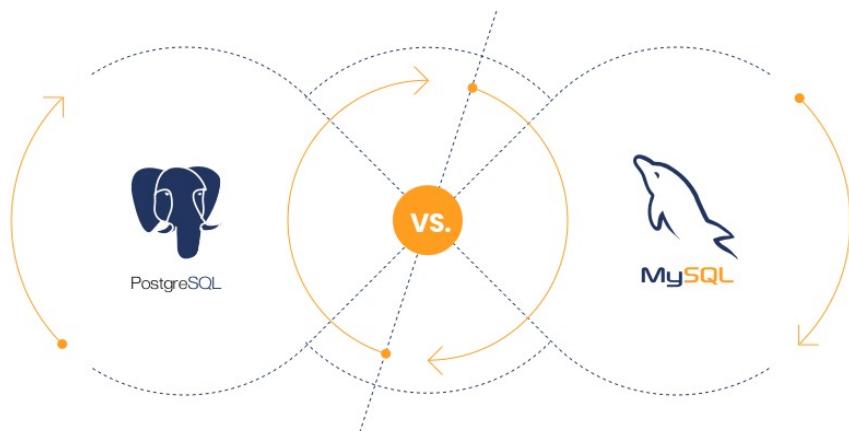


Figure 1.10: MySQL vs PostgreSQL [<https://integrio.net/>](<https://integrio.net/blog/postgresql-vs-mysql>)

MySQL dan PostgreSQL, Keduanya menyimpan data di dalam tabel yang terkait satu sama lain melalui nilai kolom umum. Namun keduanya sering dibandingkan karena terdapat beberapa perbedaan. Ingin mengenal lebih dalam? Simak penjelasan di bawah.

1.3.1 Kelebihan

MySQL	PostgreSQL
Integrasi bahasa pemrograman sangat luas;	Support framework website modern seperti Node.js dan Django; Support framework website modern seperti Node.js dan Django;
Aplikasi ringan, tidak membutuhkan spesifikasi hardware yang tinggi;	Dirilis dengan lisensi PostgreSQL sendiri;
Struktur tabel dengan fleksibilitas tinggi;	Bersifat open source dan gratis;
Dibekali banyak administrative tools;	Skala besar, mampu memuat hingga ribuan transaksi data;
Bersifat open source dan gratis (versi basic);	Memiliki banyak fitur yang mumpuni;
Meski open source, MySQL menjamin keamanan tingkat tinggi;	Memiliki banyak fitur yang mumpuni;
Mendukung berbagai variasi Data Type;	Performa sangat baik meski menuntut query yang lebih kompleks;
Dapat digunakan banyak pengguna karena mendukung multi user.	Kecepatan analisis (read-write) sangat cepat; Keamanan yang lebih ketat.

1.3.2 Kekurangan

MySQL	PostgreSQL
Sistem manajemen database kurang cocok untuk aplikasi mobile dan game;	PostgreSQL tidak mendukung semua stack development;
Technical support MySQL dinilai kurang baik;	Meski memiliki integrasi dan skalabilitas tinggi, kecepatan PostgreSQL kalah unggul dibandingkan RDBMS lain;

MySQL	PostgreSQL
Sulit diaplikasikan untuk manajemen database berskala besar.	Sistem kompatibilitas PostgreSQL menuntut pengguna untuk bekerja lebih keras dalam perbaikan dan perawatan.

1.4 Instalasi MySQL (XAMPP)

1.4.1 Download Aplikasi XAMPP

Silakan klik disini untuk mengunduh aplikasi XAMPP, pilih salah satu saja sesuai Operating System pada Komputer anda.

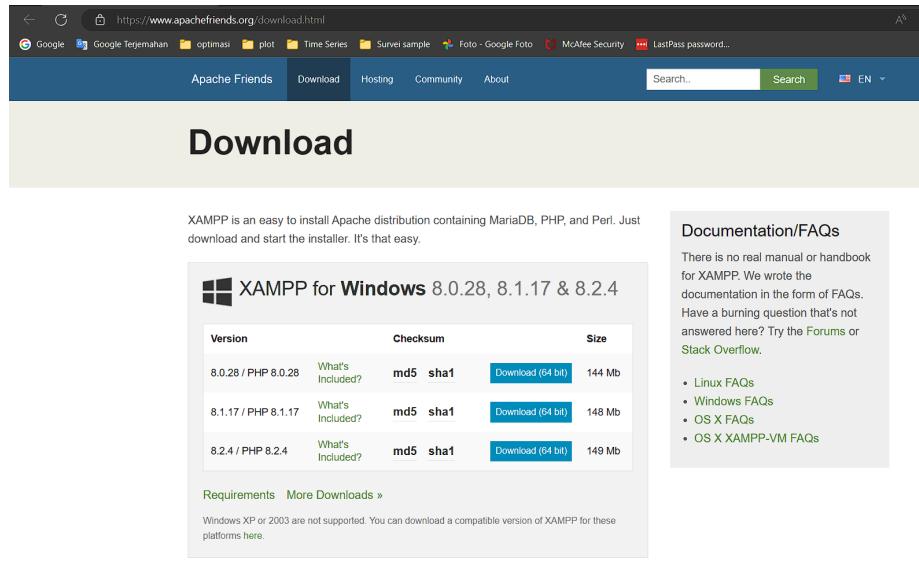


Figure 1.11: Langkah 1, Download XAMPP)

1.4.2 Install Aplikasi

Temukan file XAMPP.exe yang telah anda download, secara default biasanya disimpan di;

Selanjutnya, akan muncul Warning di klik **OK**

selanjutnya klik next

Klik next lagi, karena sudah dipilih secara default oleh XAMPP

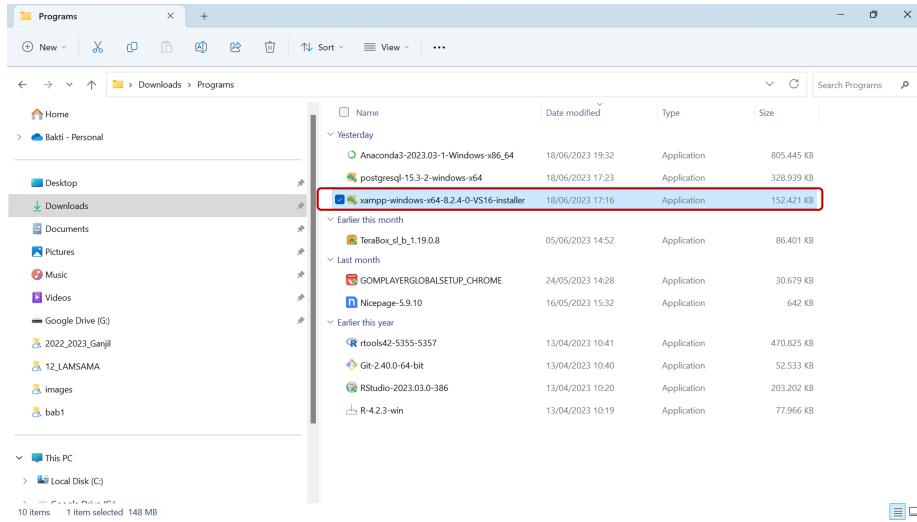


Figure 1.12: Langkah 2, Instalasi XAMPP)

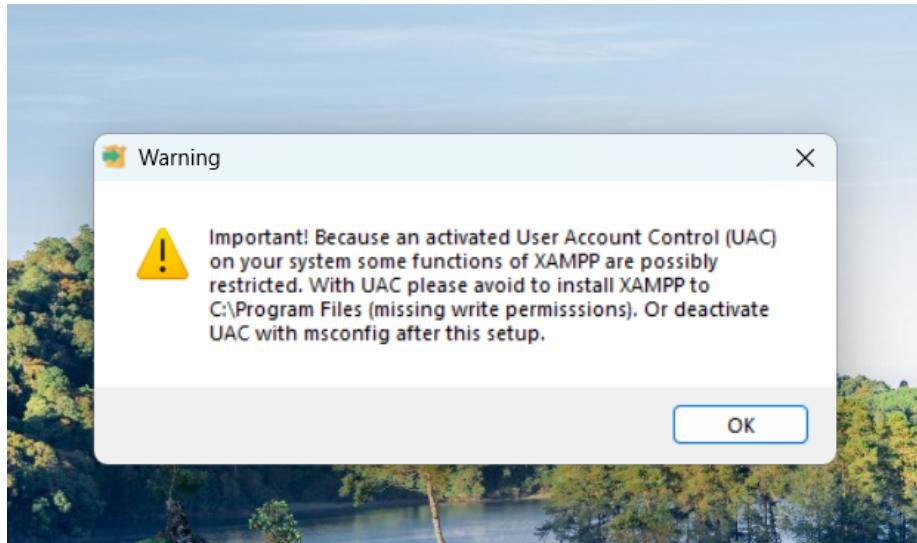


Figure 1.13: Langkah 3, Instalasi XAMPP)

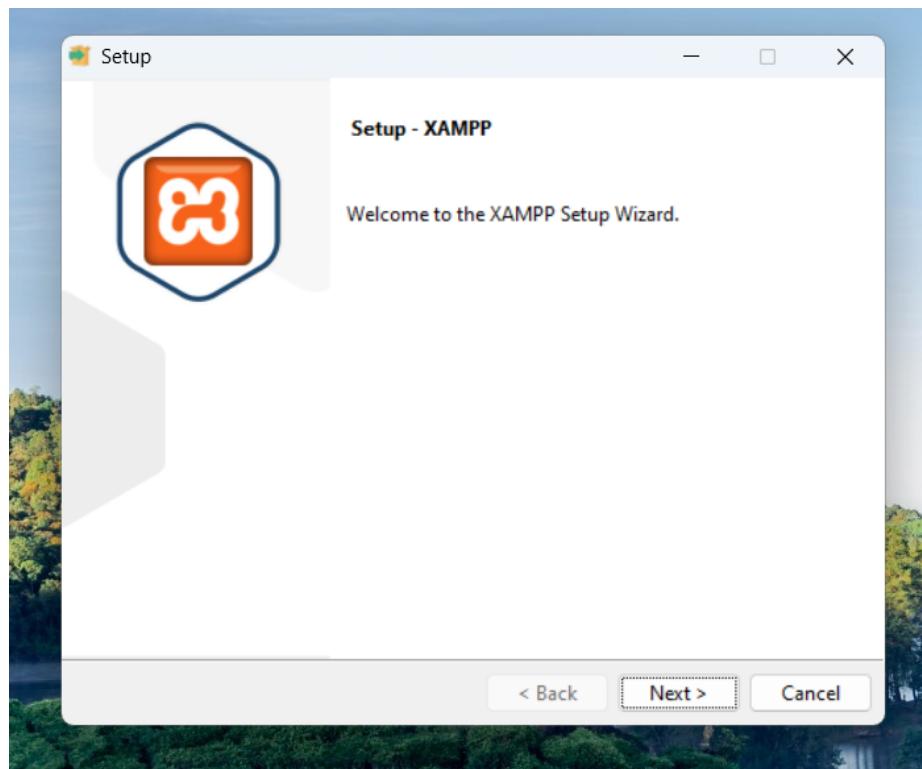


Figure 1.14: Langkah 4: Instalasi XAMPP)

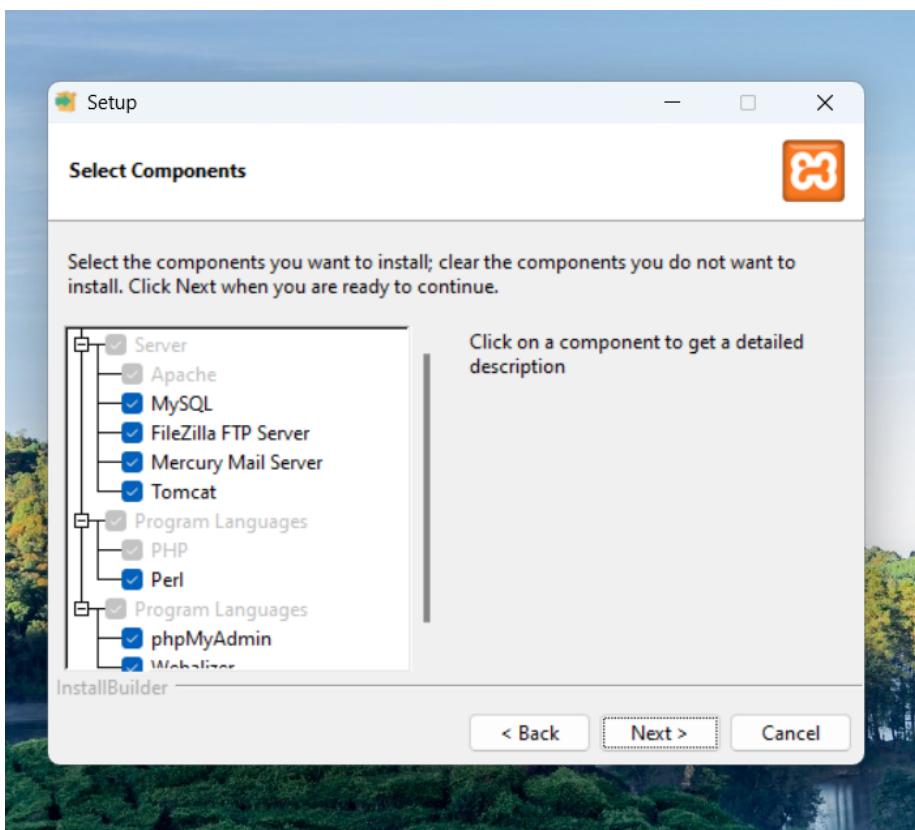


Figure 1.15: Langkah 5, Instalasi XAMPP)

1.4.3 Pilih Folder

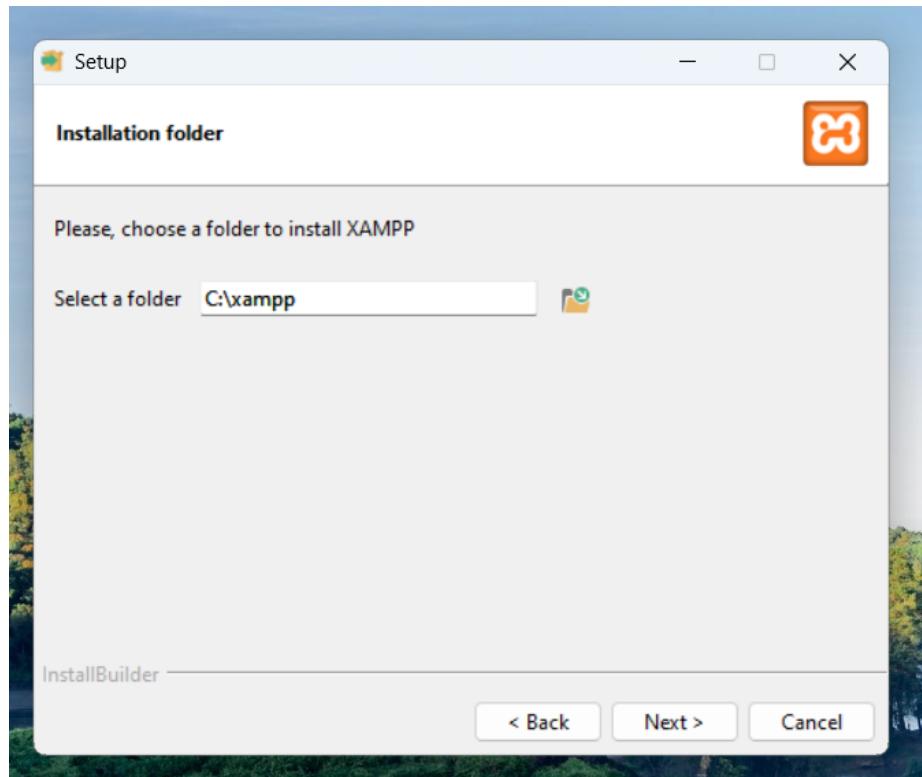


Figure 1.16: Langkah 6, Instalasi XAMPP)

Secara default akan membuat folder baru **C:\xampp**, lalu pilih next.

note: jika anda sudah pernah mendownload aplikasi xampp, perlu di hapus terlebih dahulu file xampp yang lama di file **C:\xampp**

1.4.4 Jalankan proses Instalasi

Tunggu proses instalasi selesai **Biasanya 5-10 menit, tergantung kecepatan komputer anda.**

1.4.5 XAMPP sudah terinstall

Setelah aplikasi terinstall, sudah bisa digunakan.

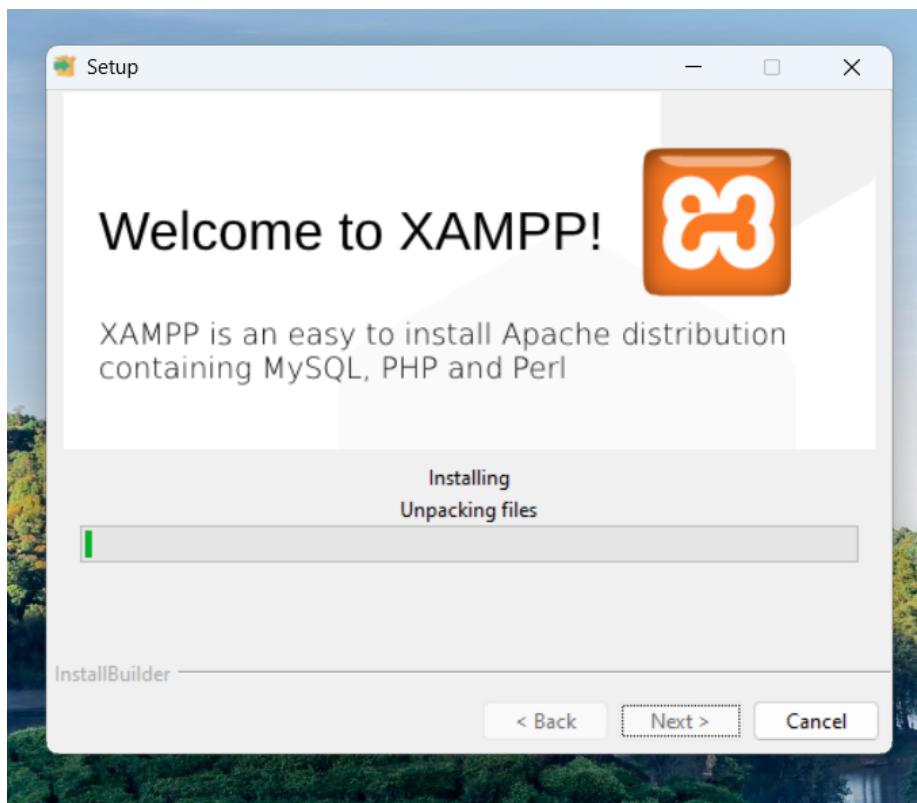


Figure 1.17: Langkah 7, Instalasi XAMPP)

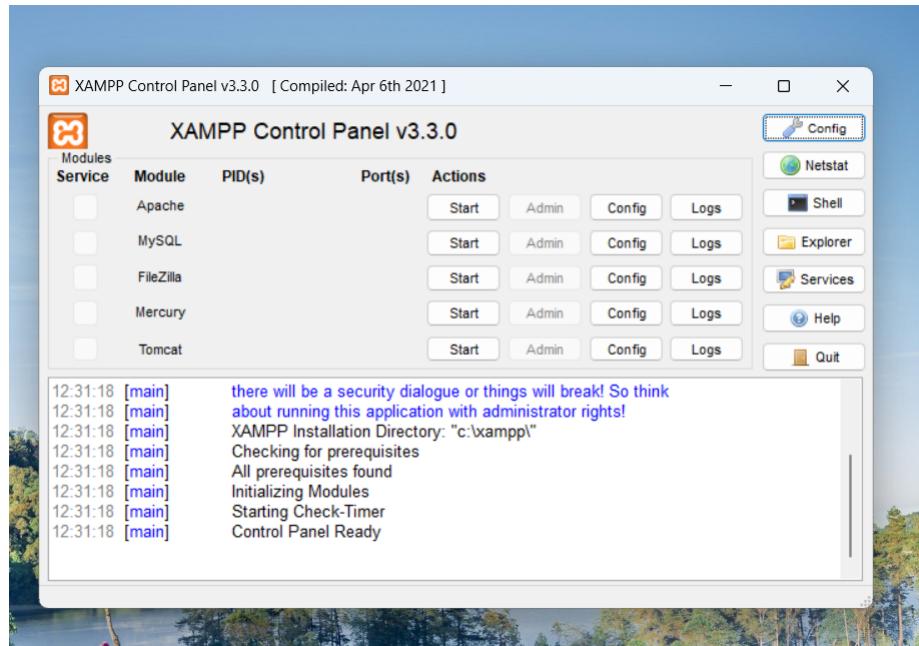


Figure 1.18: Langkah 8, Instalasi XAMPP)

1.4.6 Video Instalasi XAMPP

1.5 Instalasi PostgreSQL

Berikut ini adalah proses langkah demi langkah tentang Cara Menginstal PostgreSQL di Windows:

1.5.1 Buka Browser

Klik <https://www.postgresql.org/download> and pilih Windows

1.5.2 Cek Option

Klik Download the installer Interactive Installer by EnterpriseDB

1.5.3 Pilih PostgreSQL version

Anda akan diminta untuk memilih versi PostgreSQL dan sistem operasi yang diinginkan. Pilih versi PostgreSQL terbaru dan OS sesuai dengan environment

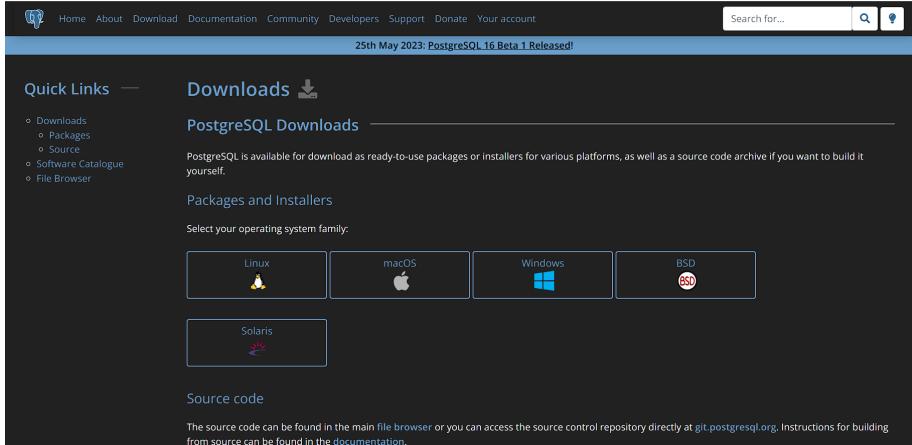


Figure 1.19: Langkah 1, Instalasi PostgreeSQL)

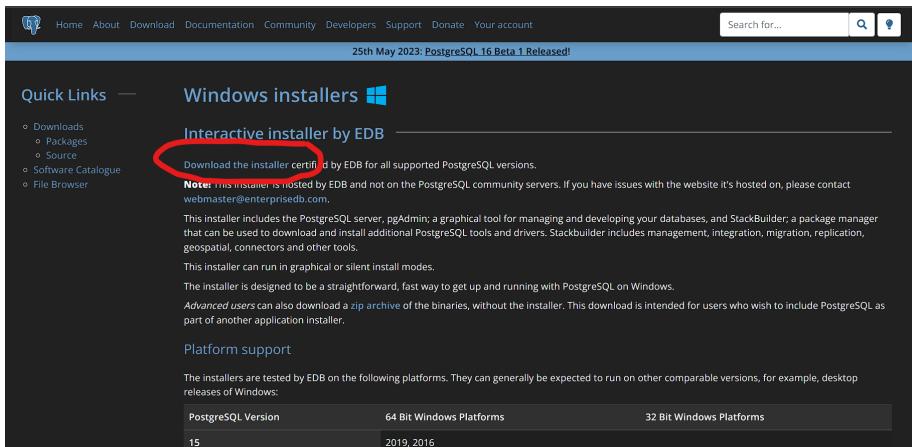


Figure 1.20: Langkah 2, Instalasi PostgreeSQL)

Anda, **klik tombol download.**

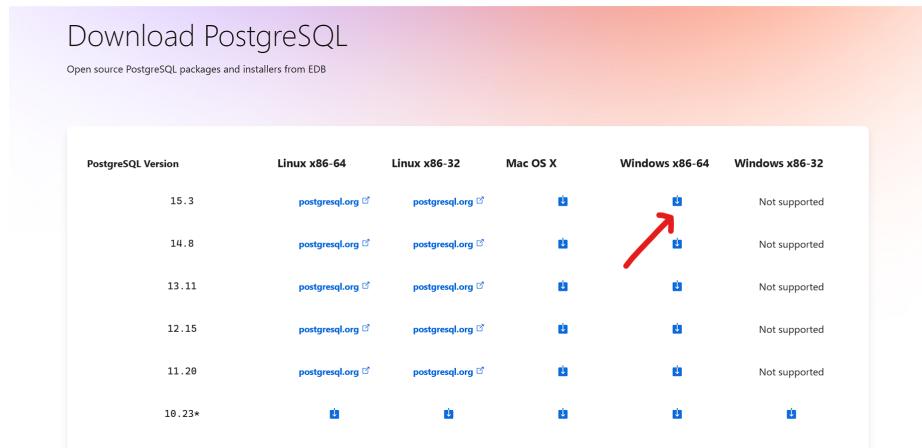


Figure 1.21: Langkah 3, Instalasi PostgreeSQL)

1.5.4 Open exe file

Setelah Anda mengunduh PostgreSQL, buka exe yang telah diunduh dan Klik berikutnya pada layar install welcome screen.

1.5.5 Pilih folder

Ubah direktori Instalasi jika diperlukan, jika tidak, biarkan default, **klik Next.**

1.5.6 Select components

Anda dapat memilih komponen yang ingin Anda instal di sistem Anda. Anda dapat menghapus centang pada Stack Builder (*disarankan ikuti secara default*), **klik Next.**

1.5.7 Check data location

Anda dapat mengubah lokasi data, **Klik Next.**

1.5.8 Masukan Password

Masukkan kata sandi superuser. Catat kata sandi tersebut, **Klik Next.**

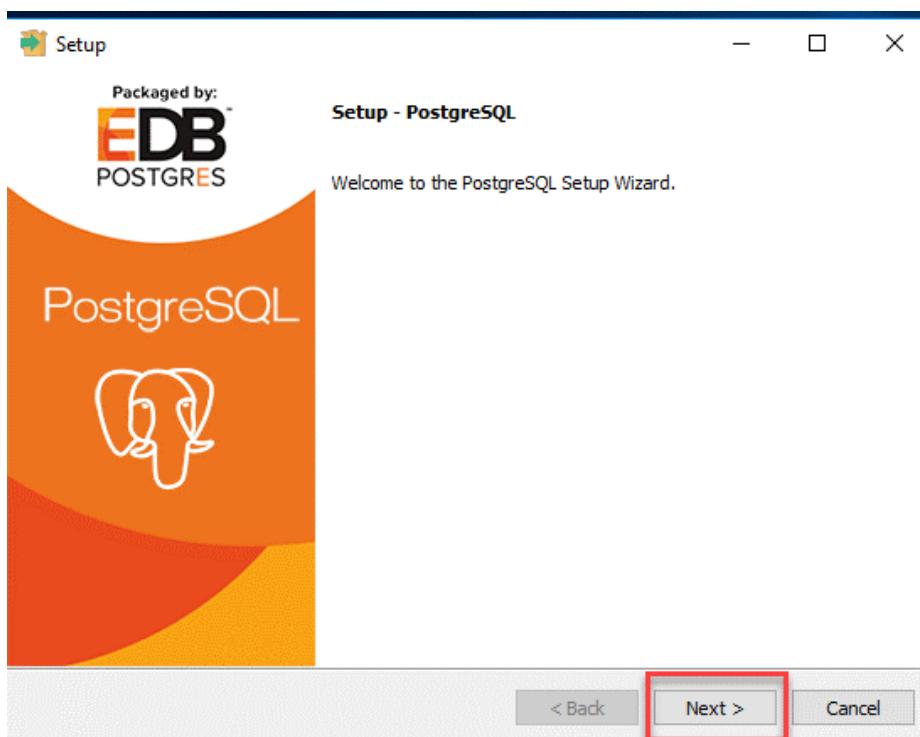


Figure 1.22: Langkah 4, Instalasi PostgreeSQL)

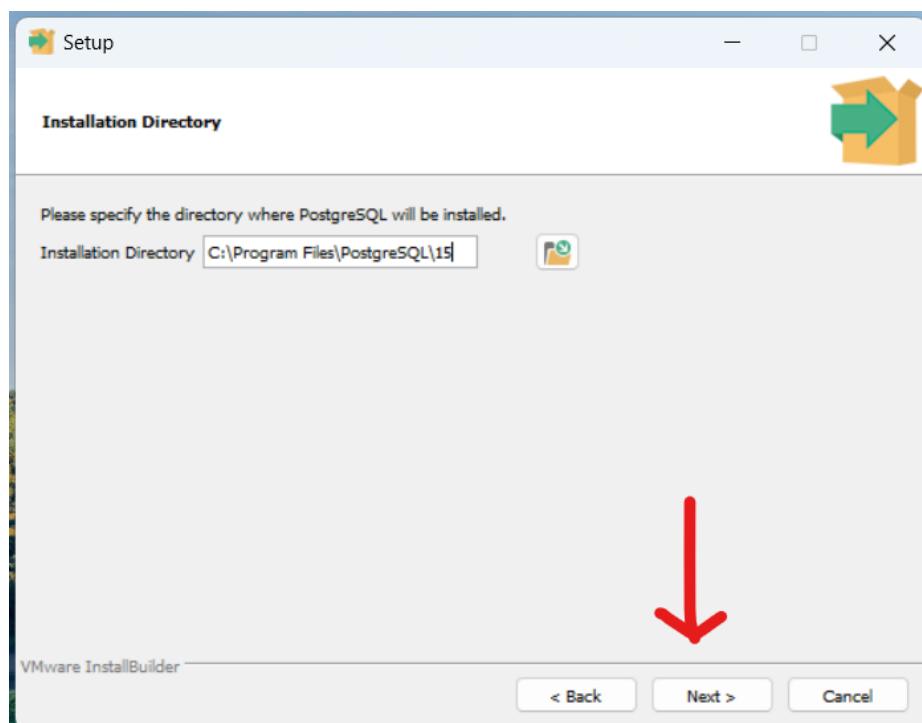


Figure 1.23: Langkah 5, Instalasi PostgreeSQL)

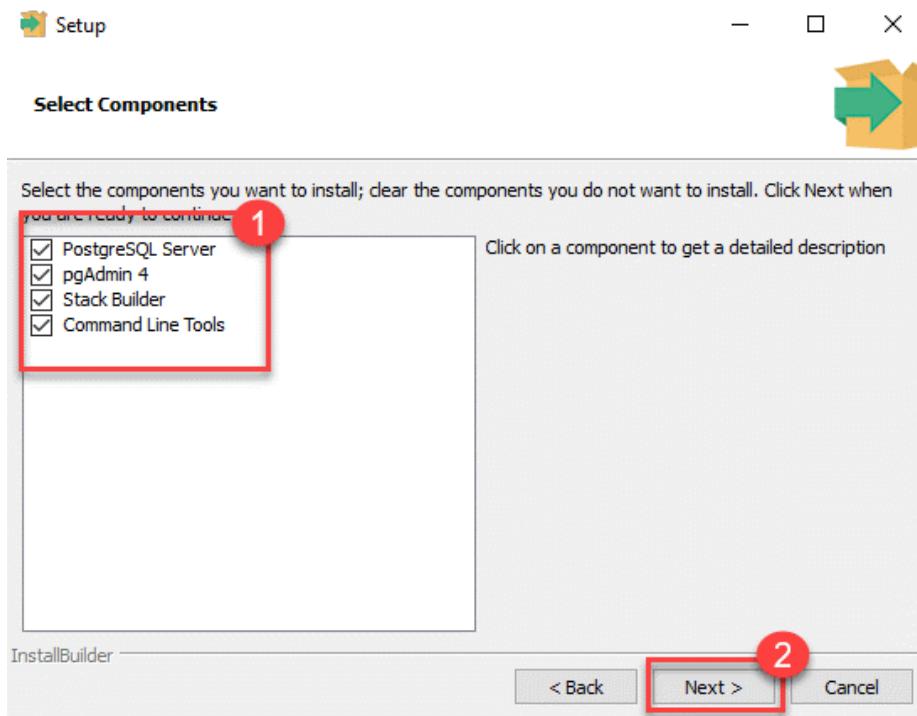


Figure 1.24: Langkah 6, Instalasi PostgreeSQL)

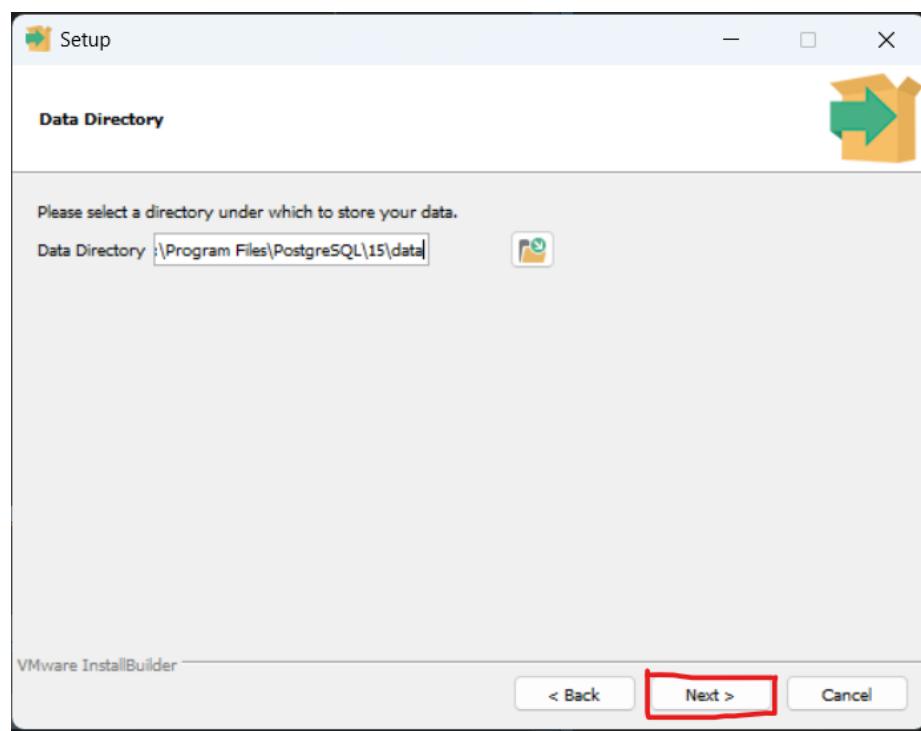


Figure 1.25: Langkah 7, Instalasi PostgreeSQL)

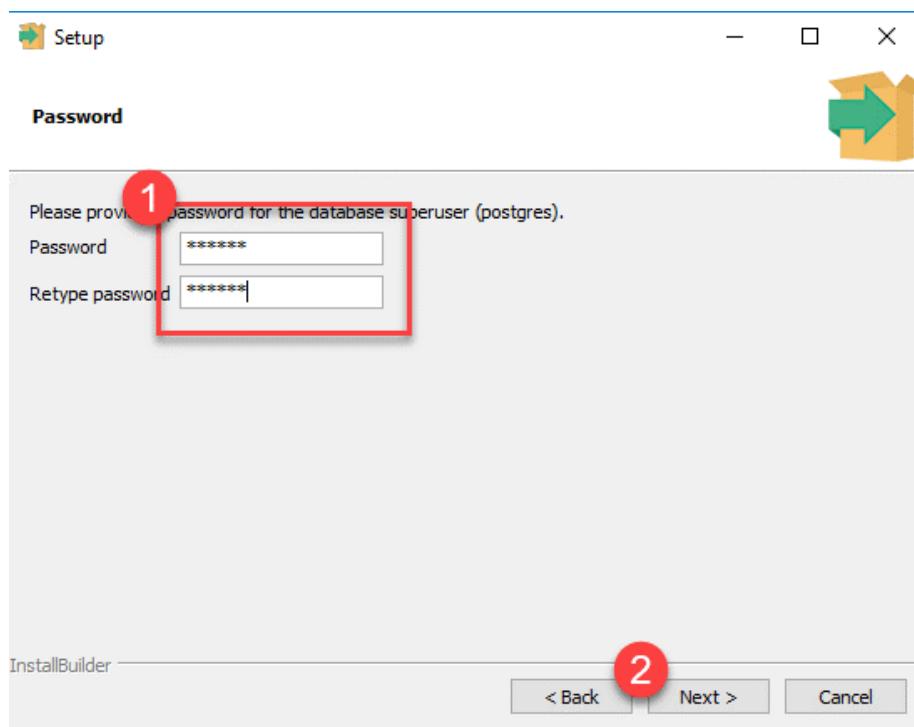


Figure 1.26: Langkah 8, Instalasi PostgreeSQL)

1.5.9 Cek opsi port

Biarkan nomor port menjadi default, **Klik Next.**

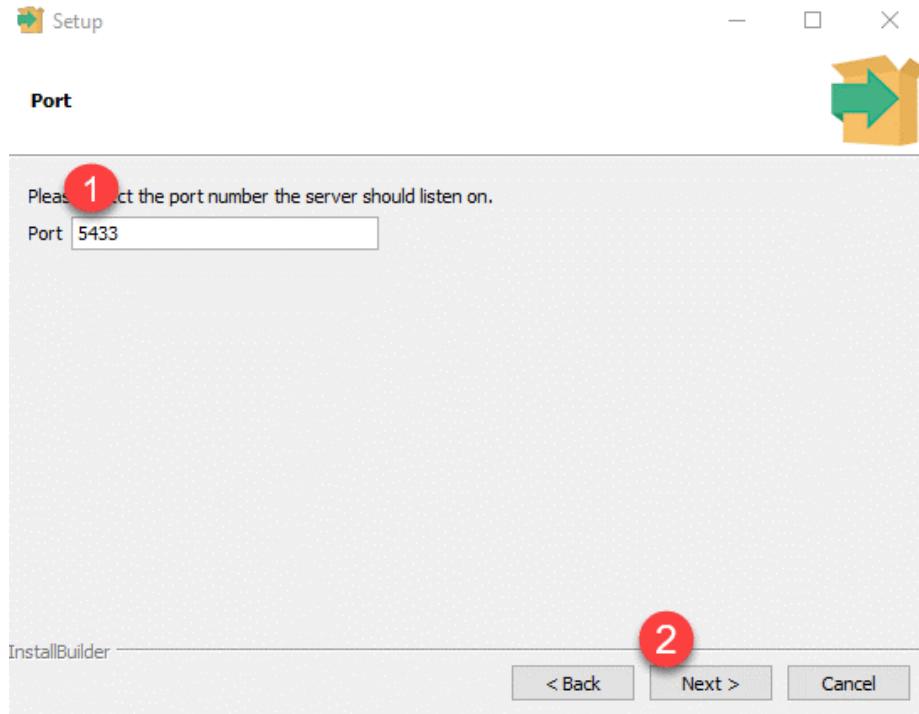


Figure 1.27: Langkah 9, Instalasi PostgreeSQL)

1.5.10 Cek Summary

Periksa pra-penginstalan summary, **Klik Next**

1.5.11 Ready to Install

Klik tombol Next

1.5.12 Check stack builder prompt

Setelah instalasi selesai, Anda akan melihat prompt Stack Builder. Hapus centang pada opsi tersebut. Kita akan menggunakan Stack Builder dalam tutorial selanjutnya, **Klik Finish.**

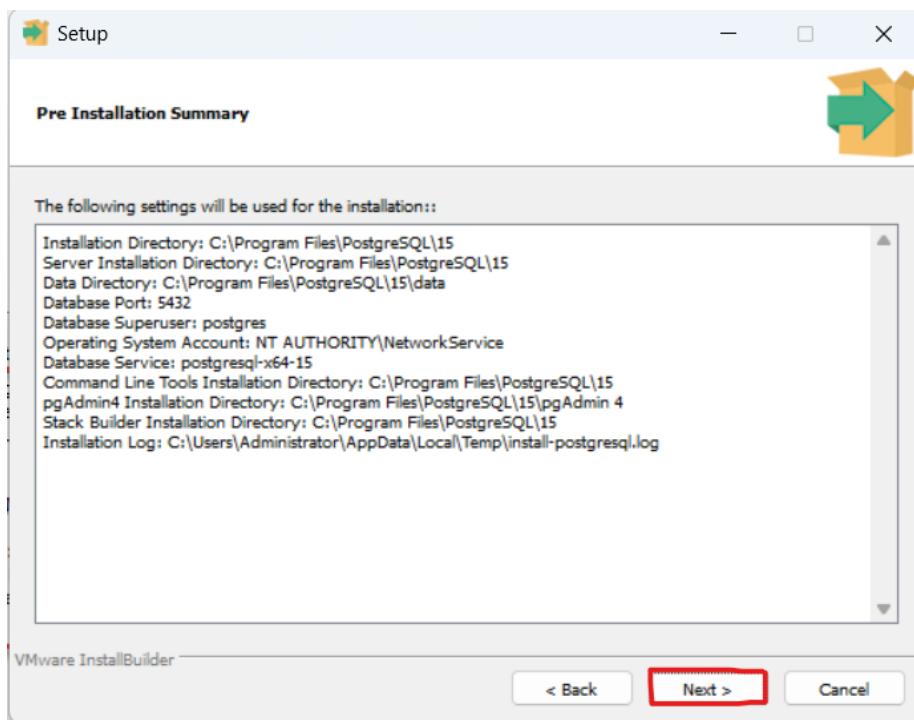


Figure 1.28: Langkah 10, Instalasi PostgreeSQL)

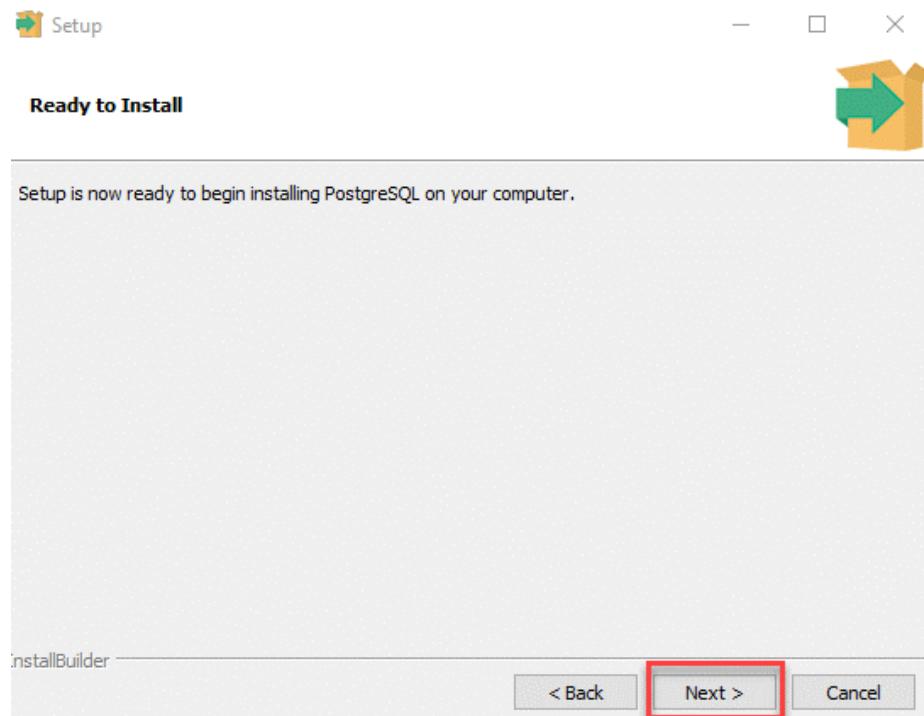


Figure 1.29: Langkah 11, Instalasi PostgreeSQL)



Figure 1.30: Langkah 12, Instalasi PostgreeSQL)

1.5.13 Launch PostgreSQL

Untuk launch PostgreSQL, buka Start Menu dan cari pgAdmin 4

1.5.14 Check pgAdmin

Anda akan melihat beranda pgAdmin

1.5.15 Cari PostgreSQL 15

Klik pada Servers > PostgreSQL 15 di sub sebelah kiri

1.5.16 Enter password

Masukkan kata sandi superuser yang ditetapkan selama instalasi, **Klik OK**

1.5.17 Cek Dashboard

Anda akan melihat Dashboard

1.5.18 Video Instalasi PostgreSQL

1.6 Praktikum

- Tutorial di MySQL (CREATE & Drop Database, Create & Drop Tabel)
- Tutorial di PostgereeSQL (CREATE & Drop Database, Create & Drop Tabel)

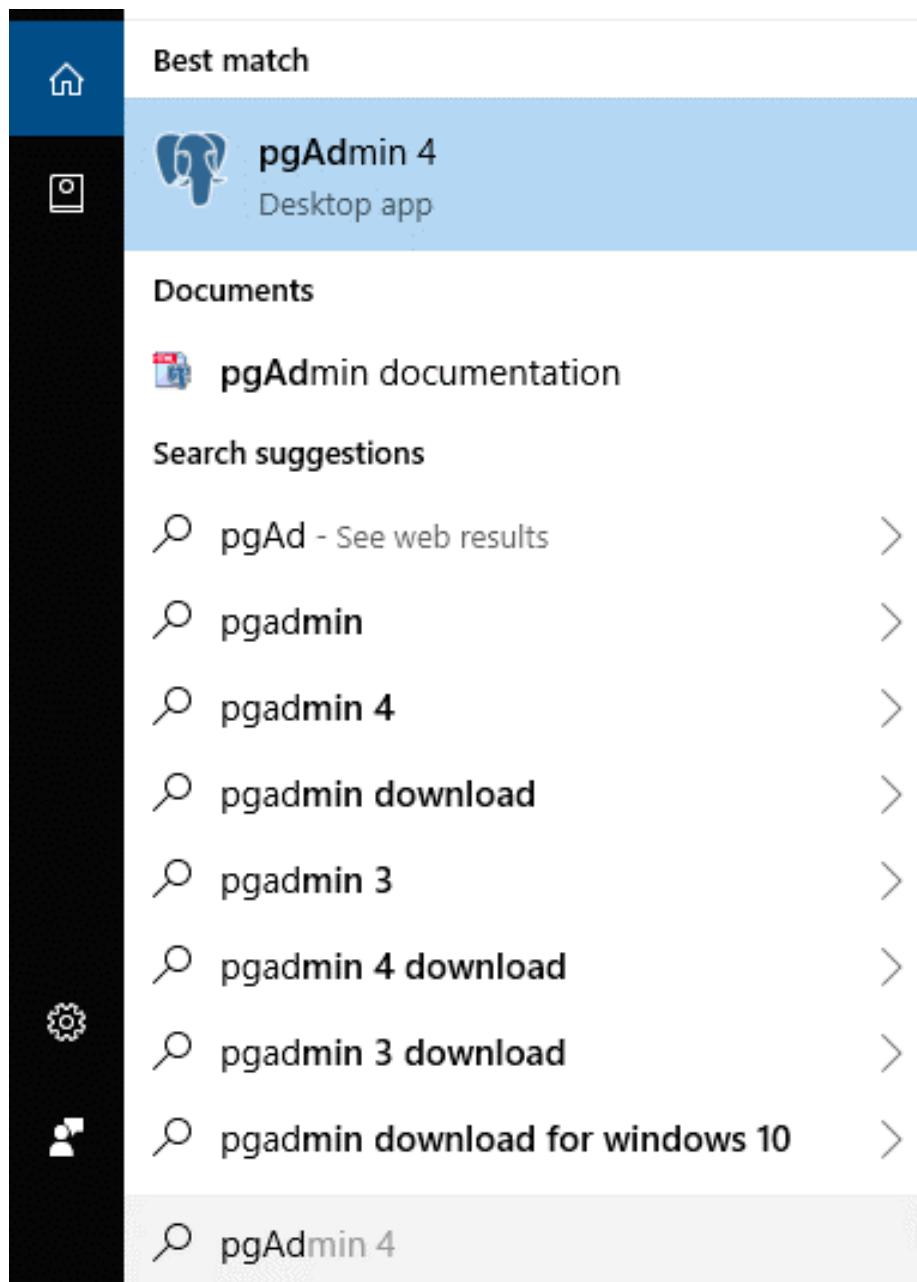


Figure 1.31: Langkah 13, Instalasi PostgreeSQL)

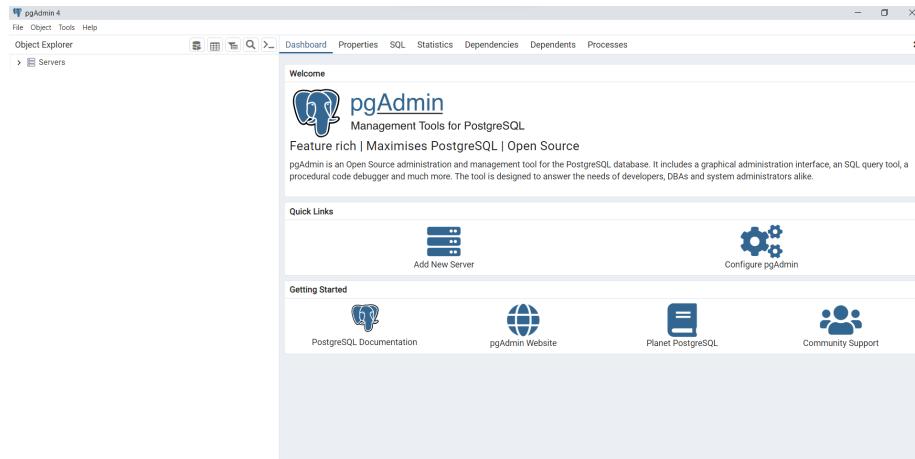


Figure 1.32: Langkah 14, Instalasi PostgreeSQL

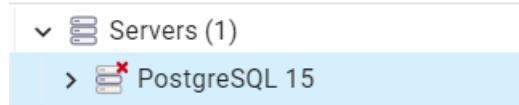


Figure 1.33: Langkah 15, Instalasi PostgreeSQL



Figure 1.34: Langkah 16, Instalasi PostgreeSQL)

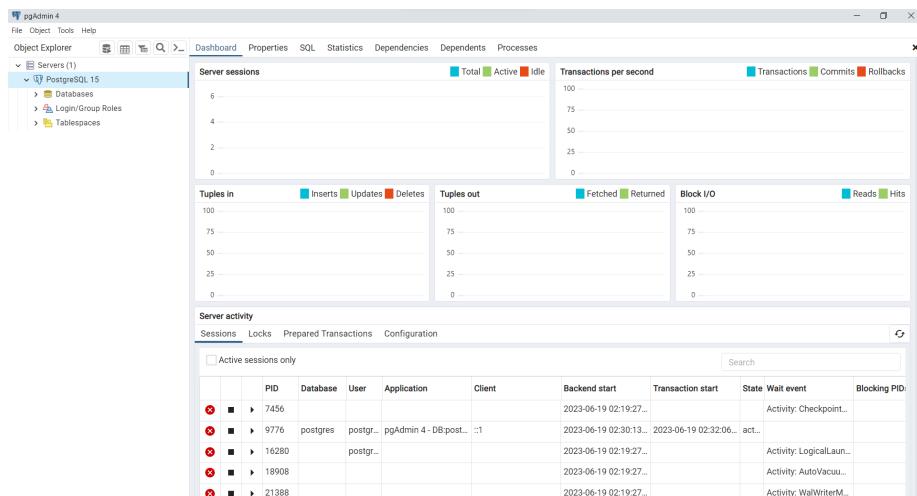


Figure 1.35: Langkah 17, Instalasi PostgreeSQL)

Chapter 2

Connecting R to SQL

2.1 Introduction

A database is a structured set of data. Terminology is a little bit different when working with a database management system compared to working with data in R.

- **field:** variable or quantity
- **record:** collection of fields
- **table:** collection of records with all the same fields
- **database:** collection of tables

The relationship between R terminology and database terminology is explained below.

R terminology	Database terminology
column	field
row	record
data frame	table
types of columns	table schema
collection of data	frames database

2.2 Connecting R to SQL

Connecting R to SQL databases allows you to leverage the power of R for data analysis while directly interacting with and querying data stored in relational

databases. This connection enables you to retrieve, manipulate, and analyze data using SQL queries within your R environment.

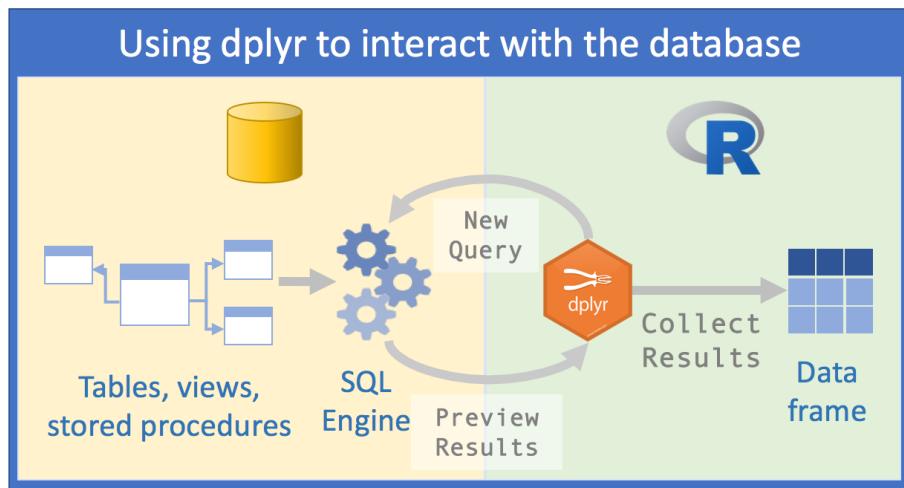


Figure 2.1: Connecting R to SQL [https://rvviews.rstudio.com](https://rvviews.rstudio.com/2017/05/17/using-r/)

Here's a step-by-step introduction to connecting R to an SQL database:

2.2.1 Install Required Packages

First, you need to install R packages that facilitate database connections and SQL interactions. The DBI package provides a common interface for database connections, and you'll also need a database-specific package like RMySQL for MySQL, RPostgreSQL for PostgreSQL, or RODBC for ODBC connections.

You can install these packages using the following commands:

```
install.packages(c(
  "RMariaDB",
  "RMySQL",
  "RPostgres"
  "RSQLite",
))
```

2.2.2 Load Packages

Then, load all these requirement packages:

```
# install.packages("pacman")
pacman::p_load(
  RMariaDB,                      # Database Interface and 'MariaDB' Driver
  RMySQL,                         # Database Interface and 'RMySQL' Driver
  RPostgres,                       # Database Interface and 'RPostgres' Driver
  RSQLite,                         # Database Interface and 'RSQLite' Driver
)
```

2.2.3 Establish a Connection

There are many ways to connect your database with R. This article shows you three of the most common ways:

MariaDB

```
MariaDB <- dbConnect(RMariaDB::MariaDB(),
  user='root',
  password='',
  dbname='bakti',
  host='localhost')
dbListTables(MariaDB)                      # table list on your database
dbExecute(MariaDB, "CREATE DATABASE new_MariaDB") # Create a new Database
dbExecute(MariaDB, "DROP DATABASE new_MariaDB") # Drop a Database
```

MySQL

```
MySQL <- dbConnect(MySQL(),
  user='root',
  password='',
  dbname='bakti',
  host='localhost')
dbListTables(MySQL)                      # table list on your database
dbExecute(MySQL, "CREATE DATABASE new_MySQL") # Create a new Database
dbExecute(MySQL, "DROP DATABASE new_MySQL") # Drop a Database
```

Postgres

```

postgres <- dbConnect(RPostgres::Postgres(),
                      user='postgres',
                      password='1234',
                      dbname='postgres',
                      host='localhost')
dbListTables(postgres)                                # table list on your database
dbExecute(postgres, "CREATE DATABASE new_SQL")
dbExecute(postgres, "DROP DATABASE new_SQL")          # Create a new Database
                                                       # Drop a Database

```

SQLite

```

RSQLite <- dbConnect(RSQLite::SQLite(), "folder_db/mydb3.sqlite")
dbListTables(RSQLite)                                 # table list on your database

```

- *Notes:* RSQLite will store the database you created in your current working directory.

2.3 Import Data

This section can be ignored if the data (table) that you need is already registered in your database. If not, then it is necessary to import data set according to your available files, download it below:

- Customers.csv
- Categories.csv
- Employees.csv
- OrderDetails.csv
- Orders.csv
- Products.csv
- Shippers.csv
- Suppliers.csv
- RawDatabase.xlsx

2.3.1 CSV Files

When you're working with files in R, such as reading data from a CSV file or saving plots as image files, R needs to know the location of these files. By setting the working directory, you provide a starting point for R to look for and save files.

In R, the `setwd()` function is used to set the working directory for your R session. The working directory is the folder on your computer where R will look for files and where it will save files unless you specify a different location. Here's why the `setwd()` function is important and when you might use it:

```
# Set the working directory
setwd("/path/to/your/folder")

# Now you can read CSV files without specifying the full path
data <- read.csv("file.csv")
```

Replace “/path/to/your/folder” with the actual path to the folder containing your CSV files. Then, you can run the following code!.

```
Customers <-read.csv("data/Customers.csv")
Categories <-read.csv("data/Categories.csv")
Employees <-read.csv("data/Employees.csv")
OrderDetails<-read.csv("data/OrderDetails.csv")
Orders <-read.csv("data/Orders.csv")
Products <-read.csv("data/Products.csv")
Shippers <-read.csv("data/Shippers.csv")
Suppliers <-read.csv("data/Suppliers.csv")
```

2.3.2 XLSX Files

```
library("readxl")
Customers <-read_excel("data/RawDatabase.xlsx",sheet=1)
Categories <-read_excel("data/RawDatabase.xlsx",sheet=2)
Employees <-read_excel("data/RawDatabase.xlsx",sheet=3)
OrderDetails<-read_excel("data/RawDatabase.xlsx",sheet=4)
Orders <-read_excel("data/RawDatabase.xlsx",sheet=5)
Products <-read_excel("data/RawDatabase.xlsx",sheet=6)
Shippers <-read_excel("data/RawDatabase.xlsx",sheet=7)
Suppliers <-read_excel("data/RawDatabase.xlsx",sheet=8)
```

2.4 Write Dataframe to Database

The key here is the `dbWriteTable` function which allows us to write an R data frame directly to a database table. The data frame's column names will be used as the database table's fields. In the following example I use `RMariaDB` connection, you can apply another driver as you like.

```

new_con <- dbConnect(MariaDB(),
                      user='root',
                      password='',
                      dbname='new_MariaDB',
                      host='localhost')

dbWriteTable(new_con, "Customers", Customers, append=T)
dbWriteTable(new_con, "Categories", Categories, append=T)
dbWriteTable(new_con, "Employees", Employees, append=T)
dbWriteTable(new_con, "OrderDetails", OrderDetails, append=T)
dbWriteTable(new_con, "Orders", Orders, append=T)
dbWriteTable(new_con, "Products", Products, append=T)
dbWriteTable(new_con, "Shippers", Shippers, append=T)
dbWriteTable(new_con, "Suppliers", Suppliers, append=T)

```

Note: Some important things that must be considered when storing table data are as follows:

- Data Structure adjustments
- Changes Data type (especially, Date and Time)

In this case, we have a problem with the data table `Employees` and `Orders`. When you consider these Table (Employees and Orders), you will find there is no date are written correctly in the database. In order to handle this problem, just type the following code in your R console:

```

dbRemoveTable(new_con, "Orders")

Orders["OrderDate"] <- as.Date(Orders$OrderDate, format = "%Y-%m-%d")

dbWriteTable(new_con, "Orders", Orders, append=T)

```

Your Exercise: Do the same thing to update data table `Employees`

2.5 Basic Queries

2.5.1 SELECT

The SELECT statement is used to select data from a database.

```
library(DT)
df1<-dbGetQuery(new_con, 'SELECT city
                           FROM Customers')
datatable(df1)
```

2.5.2 WHERE

The WHERE clause is used to filter records, extract only those records that fulfill a specified condition.

```
df2<-dbGetQuery(new_con, "SELECT *
                           FROM Customers
                           WHERE Country='Germany' ")
datatable(df2)
```

2.5.3 INSERT INTO

If you are adding values for all the columns of the table, you do not need to specify the column names in the SQL query. However, make sure the order of the values is in the same order as the columns in the table. The INSERT INTO syntax would be as follows:

```
dbExecute(new_con, "INSERT INTO Customers(CustomerName,ContactName,Address,City,PostalCode, Country)
                           VALUES('Bakti','Siregar','Jl.Bahagia Selalu','Tangerang','081369','Indonesia')")
```

2.5.4 DELETE

The DELETE statement is used to delete existing records in a table.

```
dbExecute(new_con, "DELETE FROM Customers
                           WHERE CustomerName ='Bakti' ")
```

2.5.5 UPDATE

The UPDATE statement is used to modify the existing records in a table.

```
dbExecute(new_con, "UPDATE Customers
                           SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
                           WHERE CustomerID = 1")
```

2.5.6 Disconnect Database

If you are done with the query process and you don't want to use it anymore, you should disconnect the connection from your database.

```
dbDisconnect(new_con) # disconnect from your database
```

2.6 Your Job

1. Write Dataframe to your Database directory, using the following Engine:

- RMySQL
- RPostgres
- RSQLite

2. Create a Basic Queries tutorial using all engine that you already use at task no 1, (Such as: `SELECT`, `WHERE`, `INSERT INTO`, `DELETE`, and `UPDATE`)!

Chapter 3

Referensi