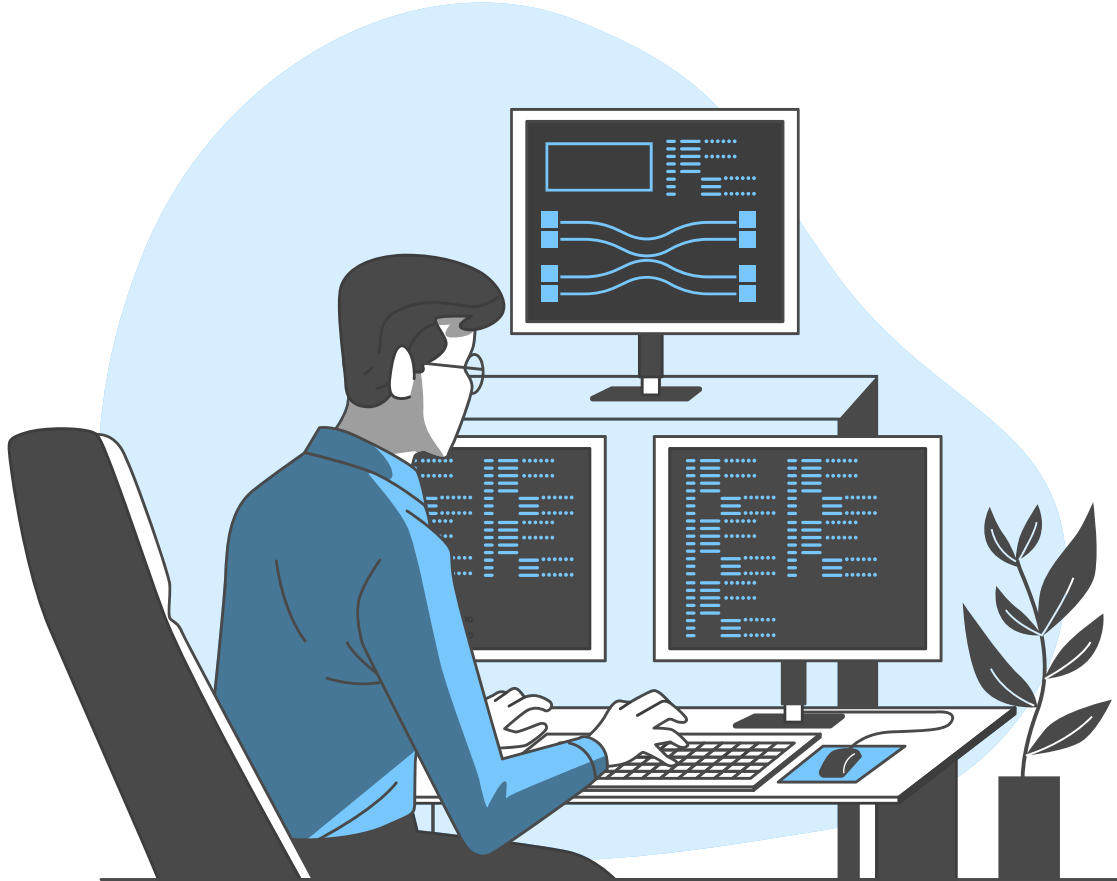
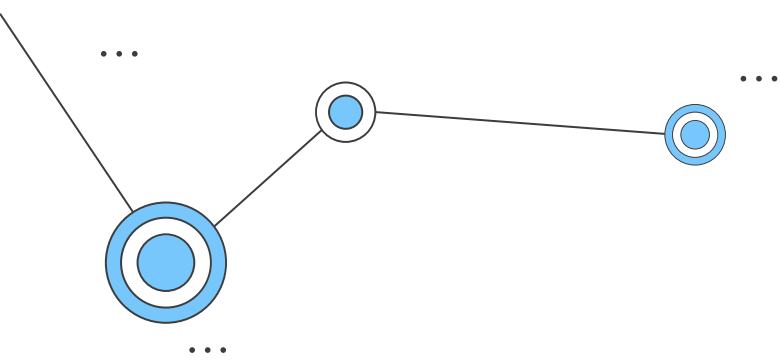


BASIS DATA LANJUT

Pertemuan 10

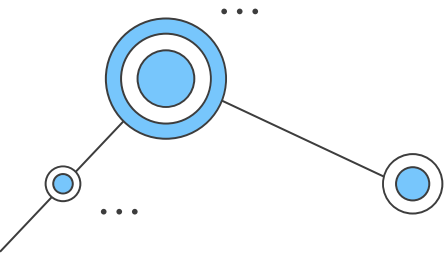
Backup-Restore dan Migrasi
MySQL → PostgreSQL





OUTLINE

- Backup dan Restore
- Migrasi

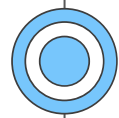




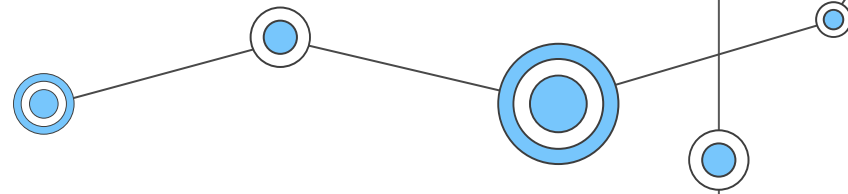
Latar Belakang

Semua perusahaan komersial dan korporasi tidak pernah 100% bebas dari ancaman data. Kewaspadaan terhadap skenario seperti kerusakan data, kegagalan host atau jaringan, atau bahkan penghancuran fasilitas penyimpanan yang disengaja sangatlah penting.

Oleh karena itu, pencadangan data merupakan aktivitas penting yang diandalkan setiap perusahaan untuk memastikan eksistensinya di pasar. Gagasan di balik pencadangan server lebih dari sekadar kemampuan untuk bersaing di pasar.

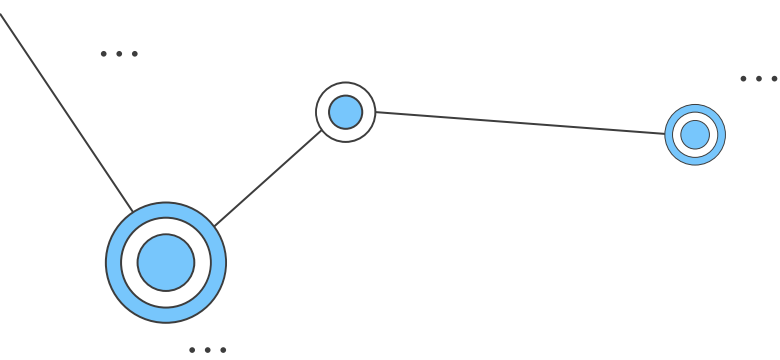


Tujuan Pembelajaran

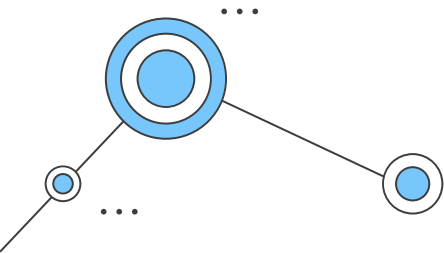


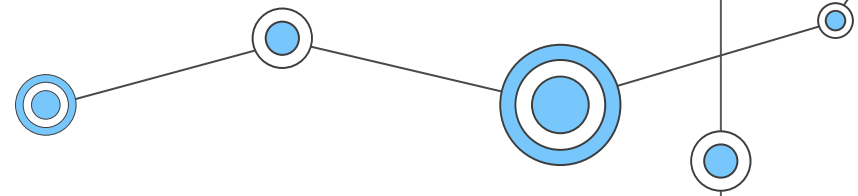
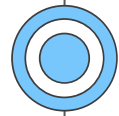
- Menjelaskan konsep dasar backup dan restore di PostgreSQL
- Melakukan proses backup dan restore menggunakan pg_dump dan pg_restore
- Memahami dan menggunakan tools migrasi dari mySQL ke PostgreSQL (misalnya pdloader)
- Melakukan konversi query secara manual antara MyDQL dan PostgreSQL





Backup dan Restore





Konsep Dasar Backup dan restore

Backup (Percadangan Basis Data) adalah proses menyalin data dari database untuk tujuan keamanan dan pemulihan jika terjadi kerusakan.

Restore (Pemulihan Basis Data) adalah proses mengembalikan data dari hasil backup ke database baru atau lama.

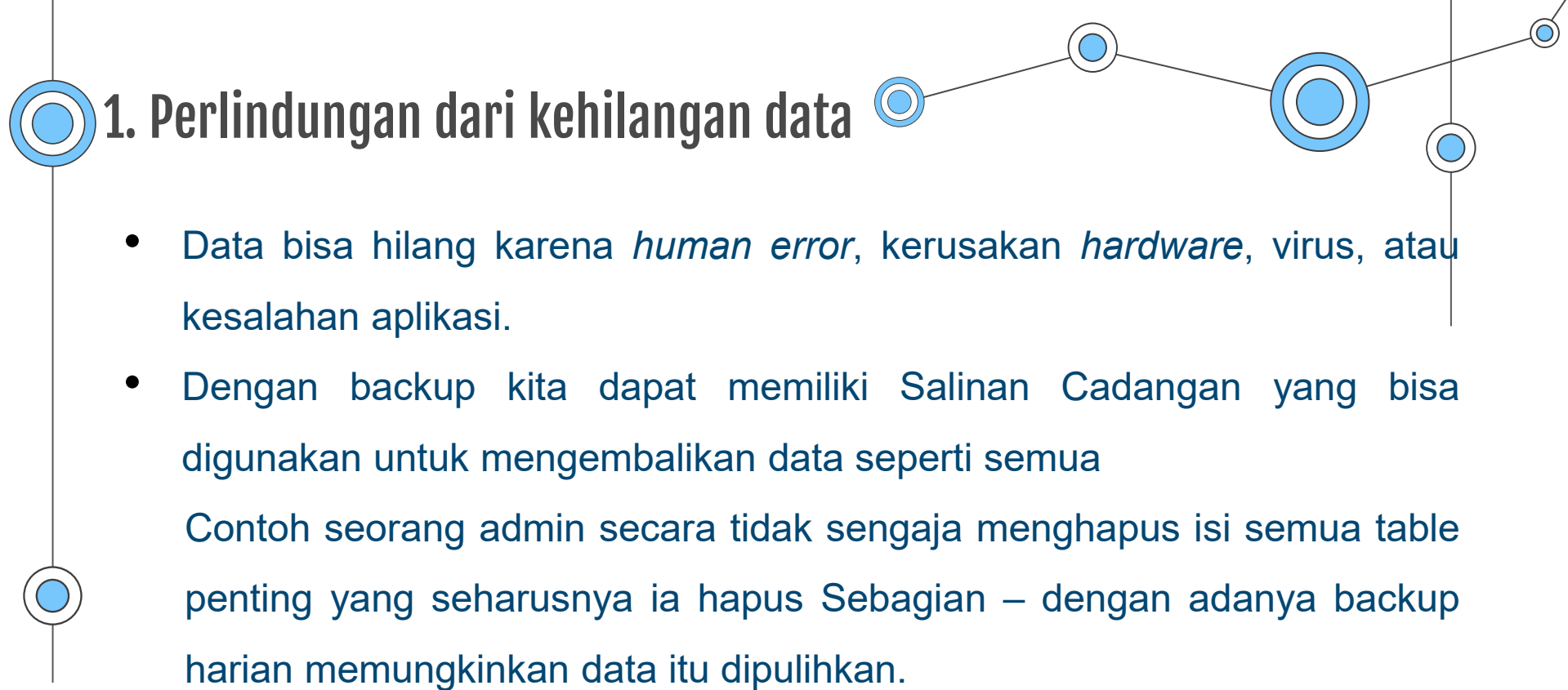


Mengapa Backup dan Restore itu Penting?

Backup dan restore **sangat penting** dalam pengelolaan database karena berkaitan langsung dengan **keamanan, keberlanjutan, dan keandalan data**.

Alasannya:

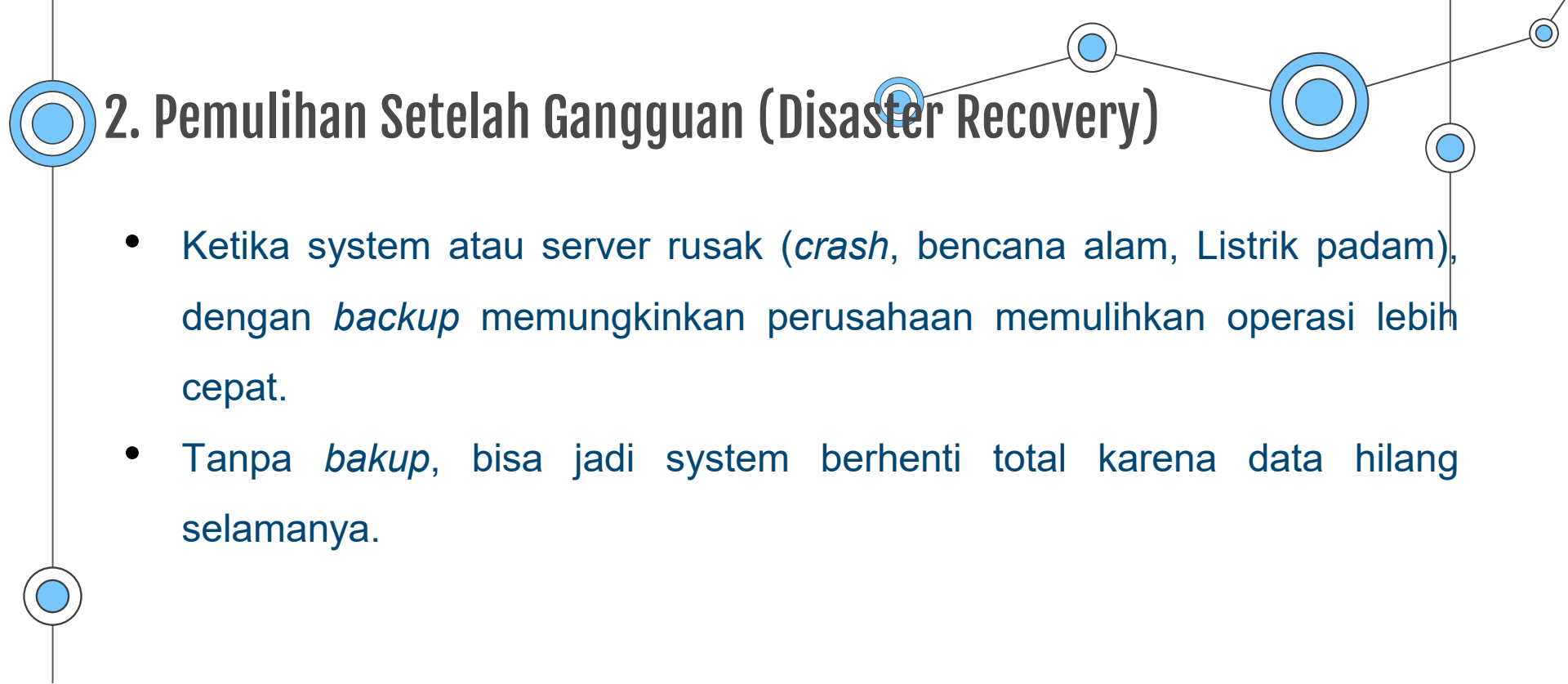
1. Perlindungan dari kehilangan data
2. Pemulihan setelah gangguan (Disaster Recovery)
3. Mendukung pengujian dan migrasi
4. Kepatuhan dan audit
5. Efisiensi pemeliharaan



1. Perlindungan dari kehilangan data

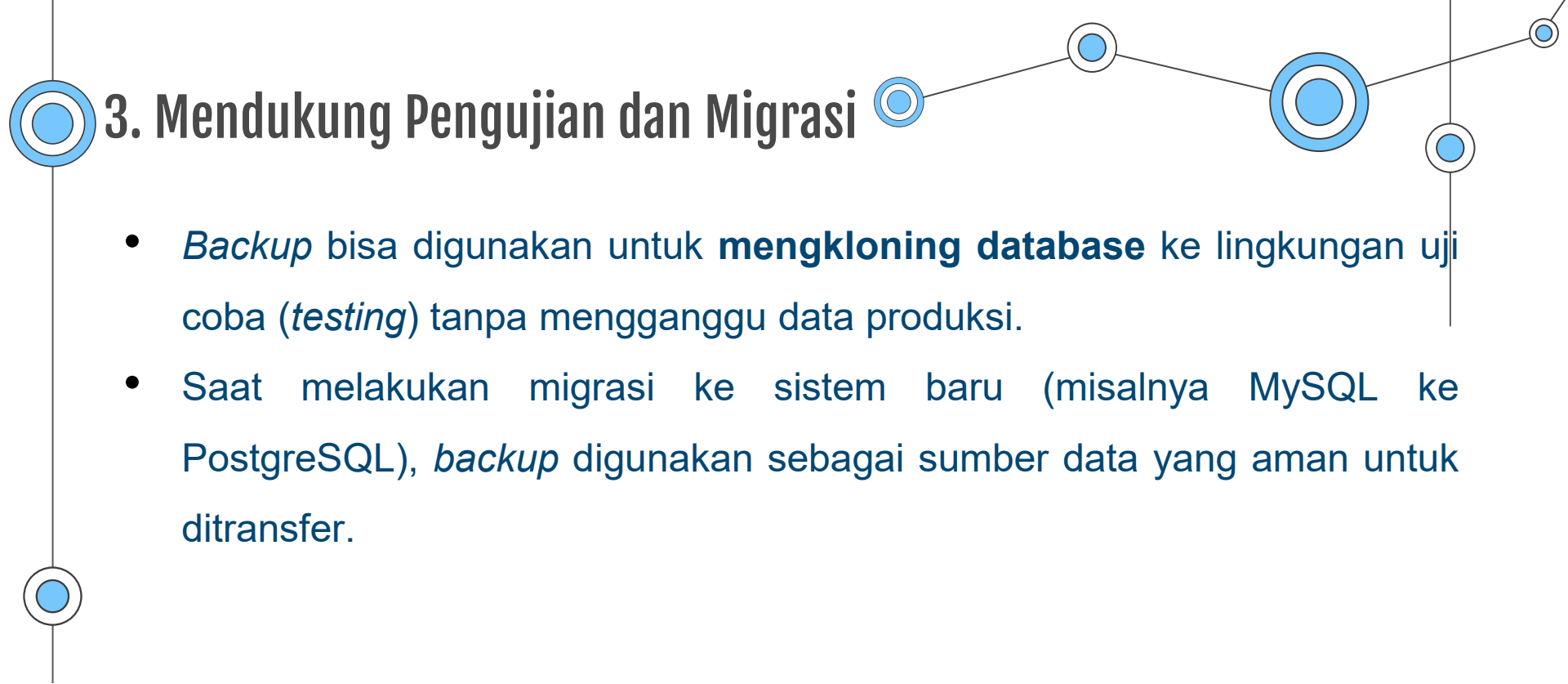
- Data bisa hilang karena *human error*, kerusakan *hardware*, virus, atau kesalahan aplikasi.
- Dengan backup kita dapat memiliki Salinan Cadangan yang bisa digunakan untuk mengembalikan data seperti semua

Contoh seorang admin secara tidak sengaja menghapus isi semua table penting yang seharusnya ia hapus Sebagian – dengan adanya backup harian memungkinkan data itu dipulihkan.



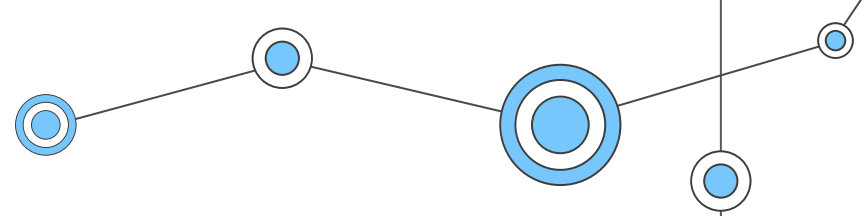
2. Pemulihan Setelah Gangguan (Disaster Recovery)

- Ketika system atau server rusak (*crash*, bencana alam, Listrik padam), dengan *backup* memungkinkan perusahaan memulihkan operasi lebih cepat.
- Tanpa *bakup*, bisa jadi system berhenti total karena data hilang selamanya.



3. Mendukung Pengujian dan Migrasi

- *Backup* bisa digunakan untuk **mengkloning database** ke lingkungan uji coba (*testing*) tanpa mengganggu data produksi.
- Saat melakukan migrasi ke sistem baru (misalnya MySQL ke PostgreSQL), *backup* digunakan sebagai sumber data yang aman untuk ditransfer.



4. Kepatuhan dan Audit

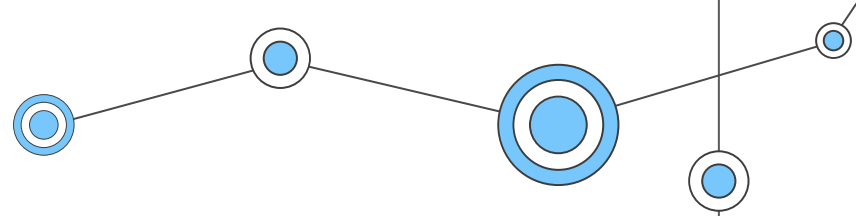
- Beberapa Perusahaan wajib melakukan *backup* rutin untuk memenuhi standar keamanan data atau aturan hukum (misalnya GDPR, ISO 27001).
- Backup juga menjadi bukti dokumentasi data yang dapat diperiksa saat audit.





5. Efisiensi Pemeliharaan

- Dengan memiliki strategi *backup-restore* yang baik, admin database bisa melakukan eksperimen, upgrade, atau pemeliharaan system tanda takut kehilangan data penting.





Ada Dua Jenis Backup dalam PostgreSQL

PostgreSQL memiliki pangsa pasar yang signifikan, mengalahkan MySQL sebagai pemimpin yang tak terbantahkan. Sebagaimana lazimnya Sistem Manajemen Basis Data (DBMS) dengan standar setinggi itu, PostgreSQL menyediakan serangkaian cadangan logis dan fisik.

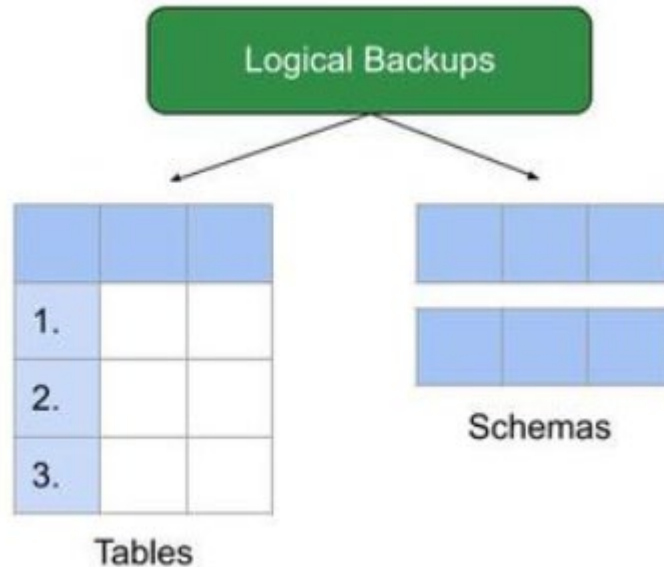
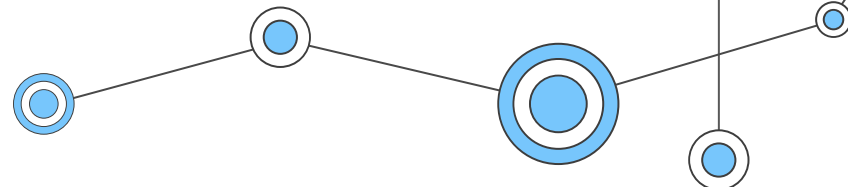
Berikut ini jenis backup dalam PostgreSQL:

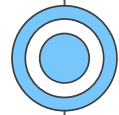
- Logical Backup
- Physical Backup



1. Logical Backup

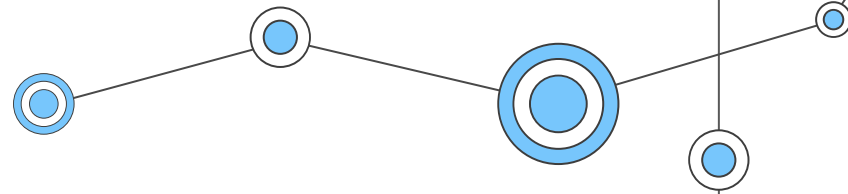
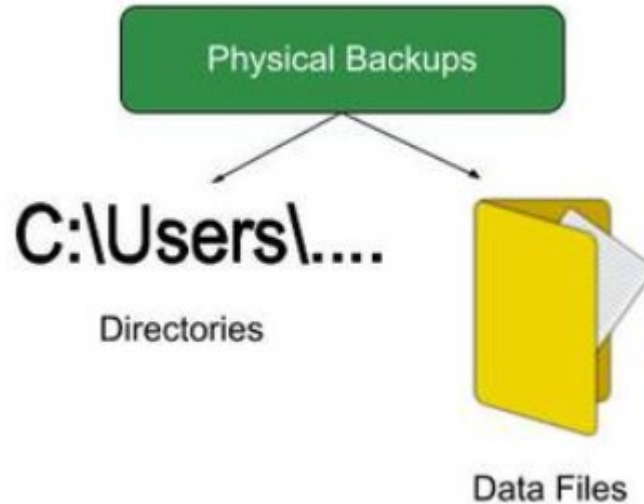
Cadangan logis berisi info mengenai basis data, seperti tabel dan skema, sedangkan cadangan fisik berisi file dan direktori informasi.





2. Physical Backup

Pencadangan logis berukuran lebih kecil dibandingkan dengan pencadangan fisik.





Fokus Tiga Pendekatan PostgreSQL dalam Backup:

1. File System Level Backup (Pencadangan Sistem Berkas)
2. SQL Dumps (Pembuangan SQL)
3. Continuous or Repetitive Archive (Pengarsipan Berkelanjutan atau Berulang)

1. File System Level Backup (1/9)

- File System Level Backup (Pencadangan tingkat sistem berkas) adalah pencadangan fisik.
- Tujuannya adalah membuat salinan semua berkas yang digunakan untuk membuat basis data. Ini berarti kita melihat kluster dan direktori basis data yang digunakan oleh PostgreSQL untuk menulis data ke dalam basis data.
- Statement:

```
tar -cf backup.tar /%p $PATH TO DATA FILE/%f $FILENAME  
OR  
rsync -v /var/lib/pgsql/13/backups/ /Directory 2/
```

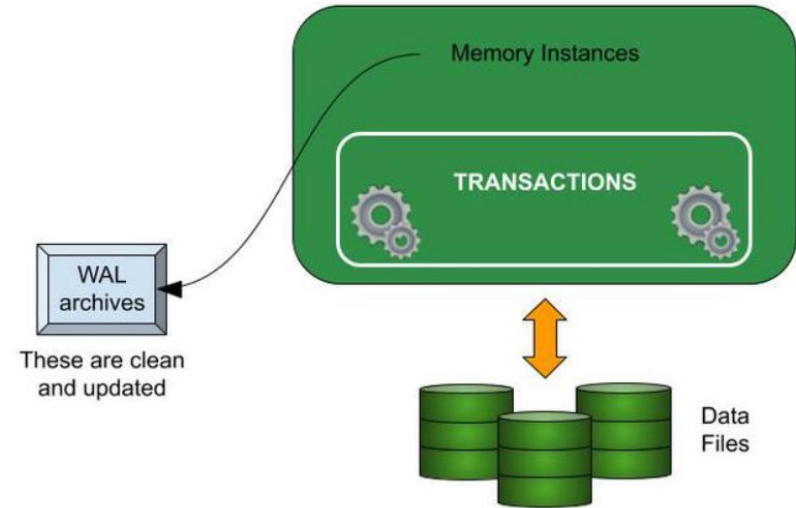


1. File System Level Backup (2/9)

- Pada statemen tersebut perhatikan bahwa backup.tar dibuat, yaitu pita arsip.
- Sementara itu, rsync adalah perintah yang MENYINKRONKAN dua direktori. Jadi, jika Direktori 2 tidak memiliki konten yang sama dengan direktori pertama (/var/lib/....), maka setelah dieksekusi, isinya akan sama - sebuah bentuk penyalinan yang cerdas.
- Namun, kita harus mengaktifkan pengarsipan sebelum eksekusi. Tentu saja, menyalin data saja dianggap ceroboh; transaksi yang sedang berlangsung akan memengaruhi status file yang sedang disalin, sehingga menyebabkan status *backup* dan keandalannya dipertanyakan

1. File System Level Backup (3/9)

- File WAL Adalah file 'Write-Ahead Log'.
- Semua transaksi yang dilakukan akan ditulis terlebih dahulu ke berkas WAL sebelum perubahan tersebut diterapkan ke ruang disk yang sebenarnya.
- Arsip WAL merupakan Salinan dari berkas WAL dan berada di direktori terpisah untuk menghindari potensi kerusakan.



1. File System Level Backup (4/9)

- **Langkah 1:** Mulai pencadangan dengan terlebih dahulu menghubungkan ke basis data sebagai pengguna istimewa, lalu jalankan perintah berikut.

```
SELECT pg_start_backup('label',false,false);  
tar -cf -z backup_name.tar /db $the name of the database you wish to  
backup
```

- /db adalah nama basis data yang ingin Anda cadangkan.
- -cf menunjukkan 'format khusus. Dengan menyertakan ini, pengguna menyatakan bahwa berkas arsip akan memiliki format khusus. Ini memberikan fleksibilitas saat mengimpor ulang berkas.
- -z Adalah atribut yang menunjukkan pada PostgreSQL untuk mengompres file keluaran.



1. File System Level Backup (5/9)

- **Langkah 2:** Beritahukan pangkaan data untuk mengakhiri mode pencadangan.

```
SELECT * FROM pg_stop_backup(false, true);
```

- Perintah ini juga membuat berkas .backup yang membantu mengidentifikasi log arsip pertama dan terakhir. Hal ini penting saat memulihkan basis data.

1. File System Level Backup (6/9)

- Dalam kasus yang sangat mungkin terjadi bahwa pengguna lupa mengaktifkan pengarsipan WAL, ini akan ditampilkan di shell:

```
Tollgate=# SELECT * FROM pg_stop_backup(false,true);
NOTICE: WAL archiving is not enabled; you must ensure that all required WAL segments are copied through other means to
complete the backup
```

lsn	labelfile	spcmapfile
0/2000170	START WAL LOCATION: 0/2000060 (file 00000001000000000000000002)+	+
	CHECKPOINT LOCATION: 0/2000098	+
	BACKUP METHOD: streamed	+
	BACKUP FROM: master	+
	START TIME: 2021-06-17 13:35:33 IST	+
	LABEL: label	+
	START TIMELINE: 1	+

```
(1 row)
```

1. File System Level Backup (7/9)

- Tetapkan yang berikut ini dalam file postgresql.conf:

```
-wal_level = replica  
-archive_mode = on  
-archive_command = 'cp %p /%f'
```

- %p Adalah nama jalur
- %f Adalah nama file
- Pengaturan parameter dalam berkas konfigurasi hanya dapat dilakukan melalui baris perintah



1. File System Level Backup (8/9)

Kekurangan File System Level Backup:

- Mengharuskan seluruh basis data dicadangkan. Mencadangkan skema atau hanya tabel tertentu bukanlah pilihan. Hal yang sama berlaku untuk proses pemulihan, yaitu seluruh basis data harus dipulihkan.
- Akibatnya, mereka memakan lebih banyak ruang penyimpanan secara default.
- Server harus dihentikan agar cadangan dapat digunakan. Hal ini mengakibatkan overhead yang tidak perlu dan mengganggu kelangsungan transaksi bisnis.

A decorative line with several circles of varying sizes and colors (blue and white) connected by thin lines, running across the top and down the left side of the slide.

1. File System Level Backup (9/9)

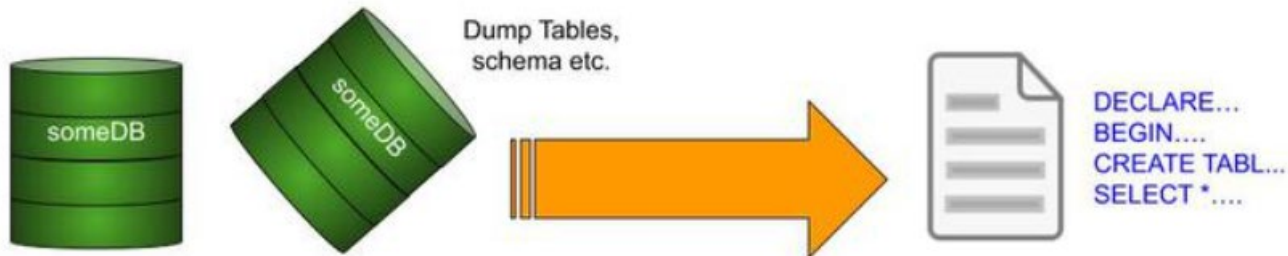
Akternatif Terdekat:

Metode lain untuk menerbitkan cadangan adalah dengan menggunakan `pg_basebackup` karena alasan berikut:

- Pemulihan dari cadangan lebih cepat dan aman
- Ini adalah versi instalasi khusus
- Pencadangan dilakukan melalui protokol replikasi

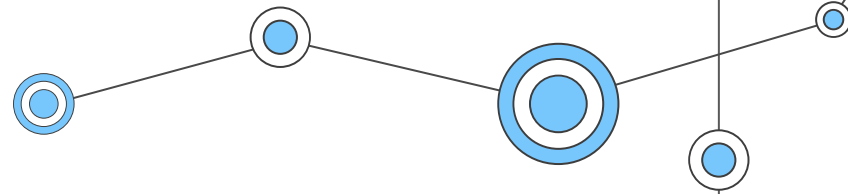
2. SQL Dump (1/7)

- Untuk basis data yang lebih kecil, jauh lebih mudah untuk menjalankan SQL Dump.
- SQL Dump adalah pencadangan logis. Ini berarti metode ini akan mencadangkan semua instruksi yang digunakan untuk membuat skema dan tabel.
- Oleh karena itu, berkas yang diekspor setelah pencadangan secara harfiah adalah berkas yang penuh dengan perintah DDL dan DML.





2. SQL Dump (2/7)



Metode 1: Menggunakan 'pg_dump'

pg_dump Adalah perintah sederhana yang membuat Salinan salah satu basis data di server. Bayangkan seperti 'menumpuk berkas objek dari SATU basis data ke dalam berkas baru'.

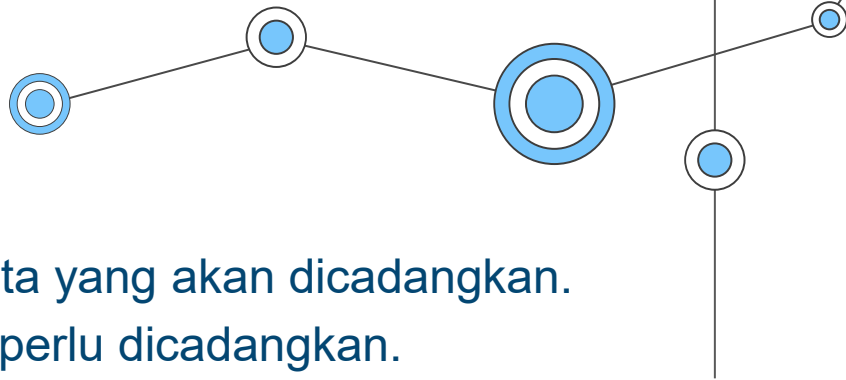
pg_dump berfungsi dengan opsi-opsi berikut:

- -Fc: mengekspor file dalam format khusus - dijelaskan di atas untuk pencadangan Tingkat Sistem File
- -Z : Perintah ini digunakan bersama perintah untuk mengompresi data. Seperti yang diharapkan, perintah ini digunakan pada basis data besar.
- -j : digunakan untuk mengaktifkan kembali pencadangan paralel menggunakan pg_dump.





2. SQL Dump (3/7)



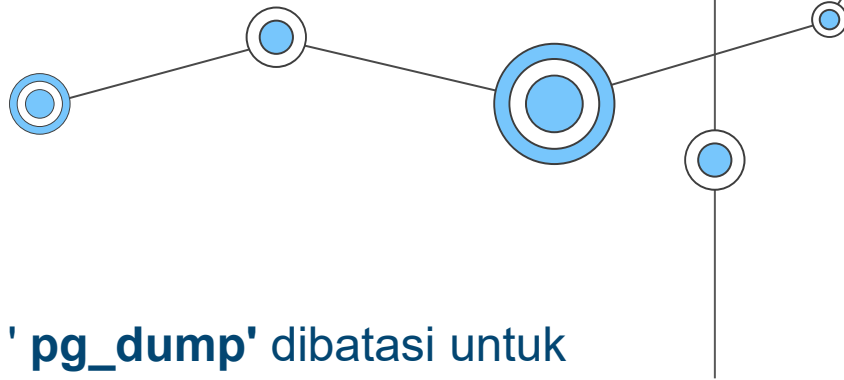
- **-a** : opsi ini menentukan bahwa hanya data yang akan dicadangkan.
- **-s** : digunakan ketika hanya skema yang perlu dicadangkan.
- **-t** : digunakan jika hanya tabel basis data yang perlu dicadangkan.

Berikut ini adalah contoh format yang dapat digunakan pengguna untuk membuat dump SQL menggunakan pg_dump:

```
>> C:\Program Files\PostgreSQL\13\bin
>> pg_dump -U username -W -F p database_name > path_to_output_file.sql
>> Password:
>>(For example)
>> pg_dump -U dbuser04 -W -F p My_University > University_backup.sql
```



2. SQL Dump (4/7)

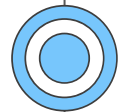


Metode 2: Menggunakan 'pg_dumpall'

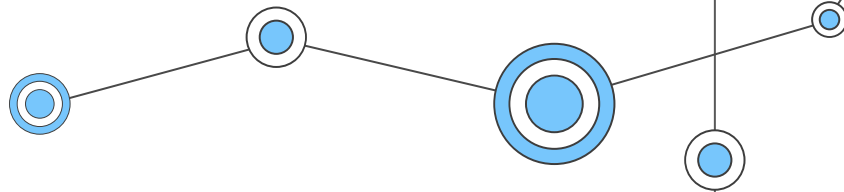
Peringatan yang tampak adalah bagaimana ' **pg_dump** ' dibatasi untuk membuat salinan satu basis data dalam satu waktu. Untuk situasi di mana beberapa basis data di server harus dicadangkan, PostgreSQL menyediakan perintah ***pg_dumpall*** .

Sebagian besar, sintaksnya tetap sama, begitu pula opsinya, kecuali - W (sejukurnya, tidak ada yang benar-benar ingin mengetikkan kata sandi untuk setiap basis data yang mereka cadangkan).

```
>> C:\Program Files\PostgreSQL\13\bin  
>> pg_dumpall -U username -F p > path_to_output_file.sql
```



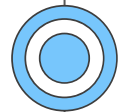
2. SQL Dump (5/7)



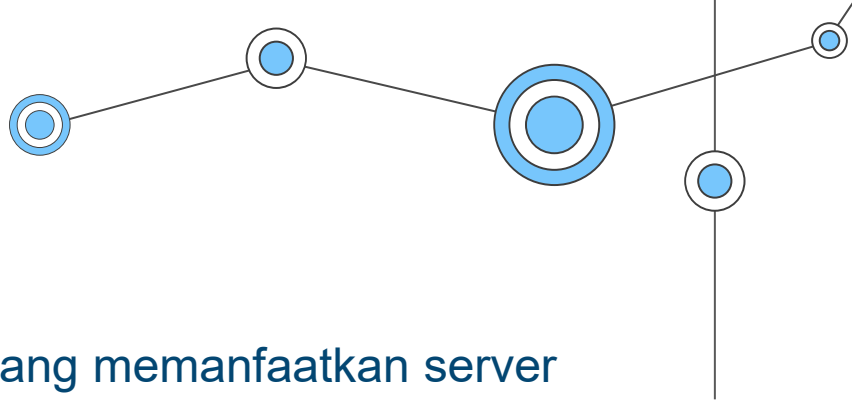
Dalam banyak kasus, perusahaan mungkin lebih suka hanya membuat salinan skema atau definisi objek tertentu, dan bukan salinan datanya. Hal ini sudah sangat diperhitungkan.

```
>> pg_dumpall --tablespaces-only > path_to_output_file.sql $for only tablespaces  
>> pg_dumpall --schema-only > path_to_output_file.sql      $for only schemas
```





2. SQL Dump (6/7)

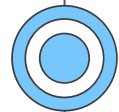


Kekurangan SQL Dump

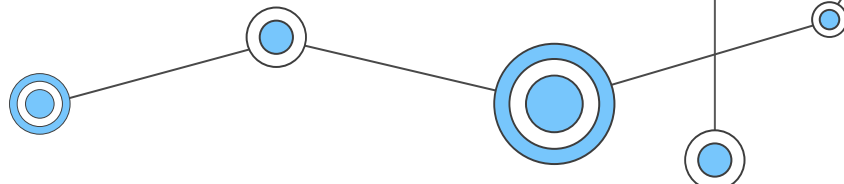
Berdasarkan banyaknya faksi IT korporasi yang memanfaatkan server PostgreSQL untuk prosedur sehari-hari, kekurangan SQL Dumps dengan mudah memberikan insentif untuk menggunakan pendekatan lain untuk Pencadangan dan Pemulihan.

- Seperti yang telah kita lihat, dump SQL secara harfiah menciptakan ulang setiap variabel berdasarkan instruksi yang tersimpan dalam berkas keluaran. Bahkan, dump ini juga berfungsi untuk membangun ulang semua indeks. Oleh karena itu, batas kecepatan pemulihan menjadi jelas.





2. SQL Dump (7/7)



Kekurangan SQL Dump

- Konsekuensi utama dari hal di atas adalah biaya overhead yang tidak diinginkan yang tidak pernah diharapkan oleh bisnis.
- Secara teknis, dengan beberapa dump yang terjadi secara bersamaan, pembalikan sintaksis merupakan masalah utama.
- Karena merupakan ekspor logis, dump SQL mungkin tidak 100% portabel.



3. Continuous Archiving (1/7)

Pengarsipan berkelanjutan atau berulang menggabungkan ide penerbitan Cadangan Tingkat Sistem Berkas dan cadangan berkas WAL (secara teknis, memungkinkan pengarsipan WAL secara terpisah). Hal ini mungkin tampak seperti Pencadangan Tingkat Sistem Berkas, tetapi dengan penekanan tambahan pada penyimpanan, bukan hanya pengarsipan, berkas WAL selama pencadangan berlangsung. Hal ini memungkinkan kita untuk mengambil 'snapshot simultan' dari basis data saat offline. Perbedaan lainnya terletak pada prosedur pemulihan - pemulihan cadangan Pengarsipan Berkelanjutan sangat berbeda.

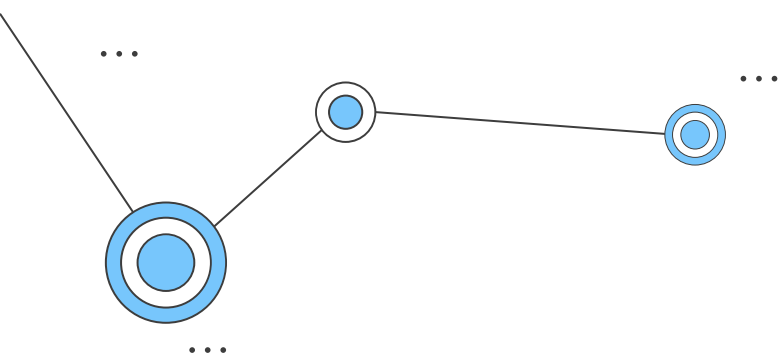
Bagian ini membahas Pencadangan Pemulihan Titik Waktu. Ini adalah bentuk pengarsipan berkelanjutan di mana pengguna dapat mencadangkan dan memulihkan data ke titik waktu tertentu di masa lalu. Istilah ini didasarkan pada arsip WAL terbaru yang dibuat.



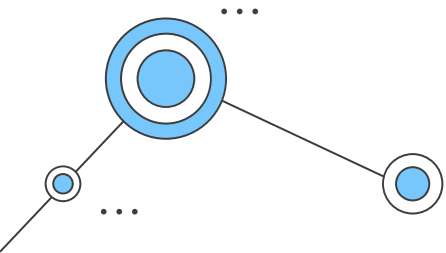
3. Continuous Archiving (2/7)

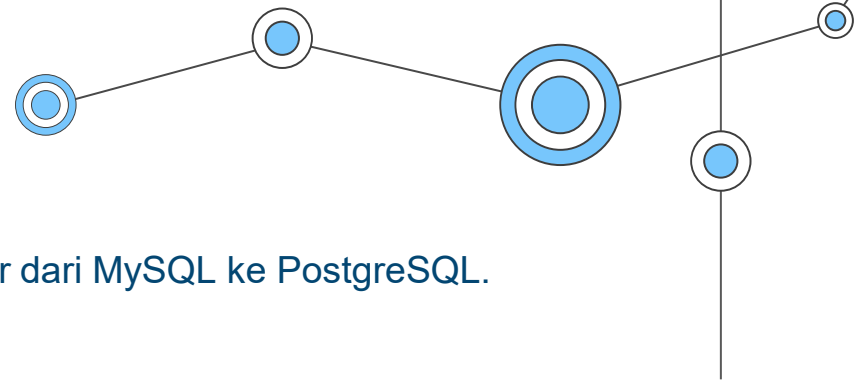
Singkatnya, prosedurnya adalah sebagai berikut:

- Aktifkan Pengarsipan WAL
- Buatlah suatu cara untuk menyimpan file WAL.
- Buat cadangan Tingkat Sistem Berkas (kali ini kita akan menggunakan **pg_basebackup**).
- Simpan cadangan sehingga DB diutar ulang ke suatu titik waktu.



Migrasi





Migrasi MySQL -> PostgreSQL

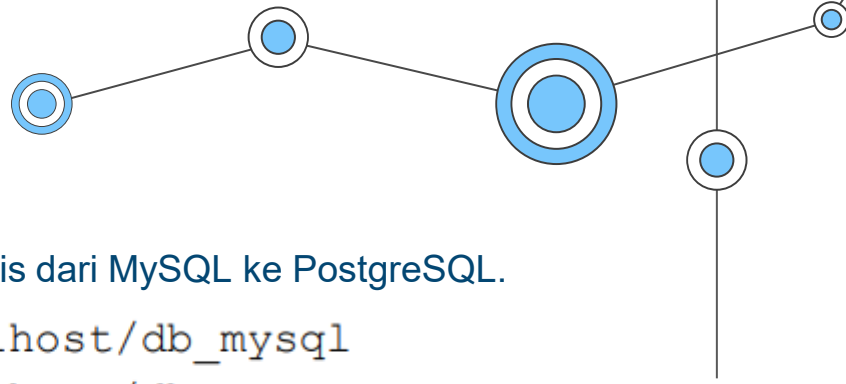
Migrasi dilakukan untuk memindahkan data dan struktur dari MySQL ke PostgreSQL.

Kelebihan dari migrasi ke PostgreSQL yaitu:

- Mendukung integritas data dan ACID compliance
- Mendukung tipe data kompleks dan fungsi kustom
- Cocok untuk system enterprise dan open-source modern



Tools Migrasi



- a. pgloader** - Tool open-source untuk migrasi otomatis dari MySQL ke PostgreSQL.

```
pgloader mysql://user:pass@localhost/db_mysql  
postgresql://postgres:pass@localhost/db_pg
```

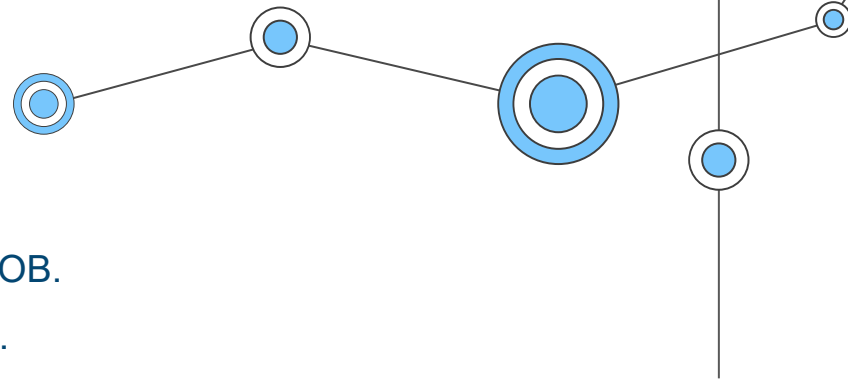
- b. Migrasi Manual** – lakukan ekspor, ubah sintaks SQL, dan impor Kembali ke PostgreSQL.

```
mysqldump -u root -p kampus > kampus.sql  
psql -U postgres -d kampus_pg -f kampus_converted.sql
```



Tantangan Migrasi

- Perbedaan tipe data seperti ENUM, TEXT, dan BLOB.
- Perbedaan fungsi built-in (IFNULL -> COALESCE).
- Penyesuaian constraint, index, dan trigger.
- Perbedaan implementasi LIMIT offset,count.



Kesimpulan

- ❑ Backup Adalah jaminan terakhir keselamatan data
- ❑ Restore Adalah jalan untuk memulihkannya
- ❑ Tanpa backup, resiko kehilangan data menjadi sangat tinggi – dan dalam dunia system informasi, kehilangan data sama dengan kehilangan asset
- ❑ Migrasi untuk memindah data dan struktur dari MySQL ke PostgreSQL
- ❑ Tools Migrasi dapat menggunakan pgloader atau Migrasi Manual
- ❑ Perlu diperhatikan perbedaan atau penyesuaian pada migrasi MySQL ke PostgreSQL

Thanks!

Do you have any questions?



Team Teaching Matakuliah Basis Data Lanjut
JTI POLINEMA