Lingkungan Database Bagian II

Oleh : Abdul Kadir

Disampaiakn Kembali Oleh : Lia Rosmalia

SQL

- SQL biasa dipakai sebagai *non-procedural language*
- Contoh, untuk mendapatkan pelanggan dengan customer-id 192-83-7465

  select *customer.customer-name*
  from *customer*
  where *customer.customer-id* = '192-83-7465'

customer-id	*customer-name*	*customer-street*	*customer-city*
192-83-7465	Johnson	12 Alma St.	Palo Alto
019-28-3746	Smith	4 North St.	Rye
677-89-9011	Hayes	3 Main St.	Harrison
182-73-6091	Turner	123 Putnam Ave.	Stamford
321-12-3123	Jones	100 Main St.	Harrison
336-66-9999	Lindsay	175 Park Ave.	Pittsfield
019-28-3746	Smith	72 North St.	Rye

(a) The *customer* table

SQL

- Untuk mendapatkan saldo dari rekening yang dipunyai oleh pelanggan dengan customer-id 192-83-7465

select *account.balance*
from *depositor, account*
where *depositor.customer-id* = '192-83-7465' **and**
depositor.account-number = *account.account-number*

account-number	*balance*
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-217	750
A-222	700

(b) The *account* table

customer-id	*account-number*
192-83-7465	A-101
192-83-7465	A-201
019-28-3746	A-215
677-89-9011	A-102
182-73-6091	A-305
321-12-3123	A-217
336-66-9999	A-222
019-28-3746	A-201

(c) The *depositor* table

SQL

- Program aplikasi mengakses *database* melalui
  - Perluasan yang memungkinkan untuk menyisipkan SQL
  - Antarmuka program aplikasi (misalnya ODBC/JDBC) yang memungkinkan *query* SQL dikirim ke database

SQL

- Contoh pada VB:

```
On Error Resume Next
   AdodcAtpm.ConnectionString = Conn.ConnectionString
   If Err.Number <> 0 Then
      KeluarTanpaSyarat = True
      MsgBox "Problem membuka koneksi", vbOKOnly, "Perhatian"
      Unload FormAccess
      Exit Sub
   End If

   AdodcAtpm.CommandType = adCmdText
   AdodcAtpm.RecordSource = "SELECT * FROM ATPM;"
   On Error GoTo 0
```

SQL

Contoh pada PHP:

```
$id_mysql = mysql_connect("localhost", $pemakai,$password);
 if (! $id_mysql)
   die("Database MySQL tak dapat dibuka");

 if (! mysql_select_db("jflora_plant", $id_mysql))
   die("Database tidak bisa dipilih");

 $hasil = mysql_query("SELECT * FROM category",
                      $id_mysql);
 if (! $hasil)
   die("Permintaan gagal dilaksanakan");

 while ( $baris = mysql_fetch_row($hasil) )
 {
   print("$baris[1]<BR>\n");
 }

 mysql_close($id_mysql);
```

Evolusi Database

- Tahun 60-an
  - Sistem pemrosesan berkas
  - DBMS tahap awal (misalnya untuk menangani proyek Apolo – pendaratan di bulan)
- Tahun 70-an
  - *Database* hierarkikal dan *network*
  - Beberapa kelemahan *database* saat itu:
    - Sulit untuk mengakses data
    - Independensi terhadap data masih sangat terbatas
    - Belum tersedia landasan teori yang kokoh
    - Konsep *database* relasional belum dikenal

Evolusi Database (lanjutan…)

- Tahun 80-an
  - *Database* relasional mulai dikenal secara meluas
  - *Database* mudah diakses melalui SQL
  - Di lingkungan PC, dBase II dan dBase III+ sangat populer
- Tahun 90-an
  - Tren *client-server* dan aplikasi Internet
  - Penerapan *database* berorientasi objek
  - *Database* multimedia
  - *Database* cerdas

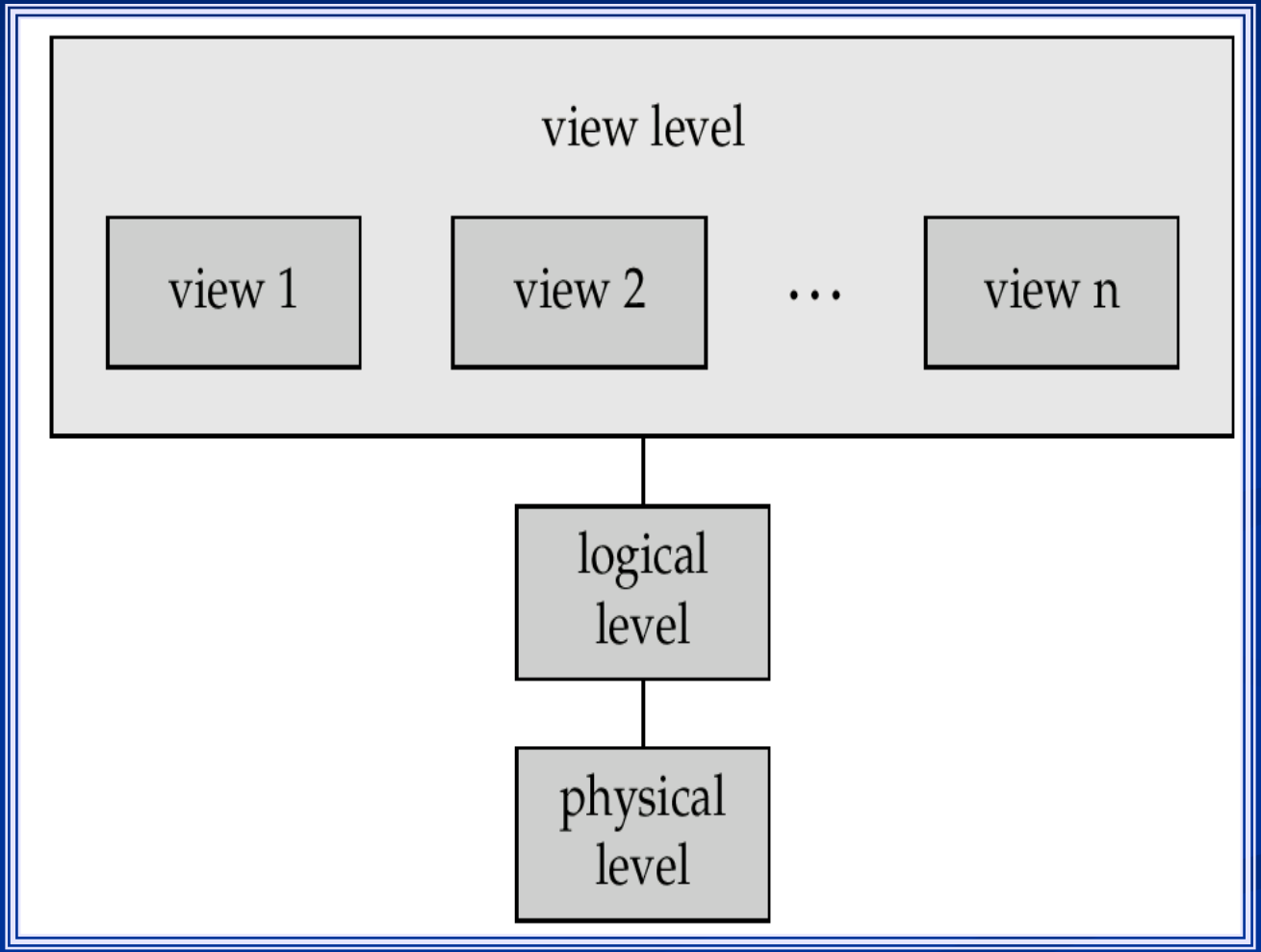
Evolusi Database (lanjutan…)

- Tahun 2000-an
  - Penanganan data yang kompleks (*multidimensional data*)
  - *Database* terdistribusi

Arsitektur Tiga Skema

- Arsitektur tiga skema merupakan suatu pendekatan yang ditujukan kepentingan abstraksi data
- Skema adalah struktur logika dalam database
- Abstraksi data dimaksudkan agar pemakai tidak perlu tahu tentang bagaimana DBMS secara detail menyimpan dan memelihara *database*

Arsitektur Tiga Skema



View Level

- *View level* merupakan lapisan tertinggi pada arsitektur tiga skema
- Pada level ini pemakai hanya mengenal struktur data yang sederhana, yang berorientasi pada kebutuhan pengguna
- Data yang dikenal oleh masing-masing pengguna bisa berbeda-beda dan kemungkinan hanya mencakup sebagian data dalam *database*
- Sebagai contoh pada level ini, seseorang pengguna mislanya tidak boleh mengakses data gaji

Logical Level

- Disebut juga *Conceptual Schema*
- Menjabarkan data apa saja yang sesungguhnya disimpan dalam *database* dan mendeksripsikan hubungan antardata
- Level ini biasa dipakai oleh DBA

Physical View

- Disebut juga *Physical Schema*
- Menjelaskan bagaimana data sesungguhnya disimpan dalam memori sekunder

Gambaran Abstraksi Data

- Contoh suatu struktur data

```
Pegawai = RECORD
            Nama : STRING[25];
            Alamat : STRING[25];
            Bagian : STRING[10];
            Gaji : LONGINT;
          END;
```

- Contoh di atas menyatakan *record* bernama Pegawai mengandung 4 Field

Gambaran Abstraksi Data (lanjutan…)

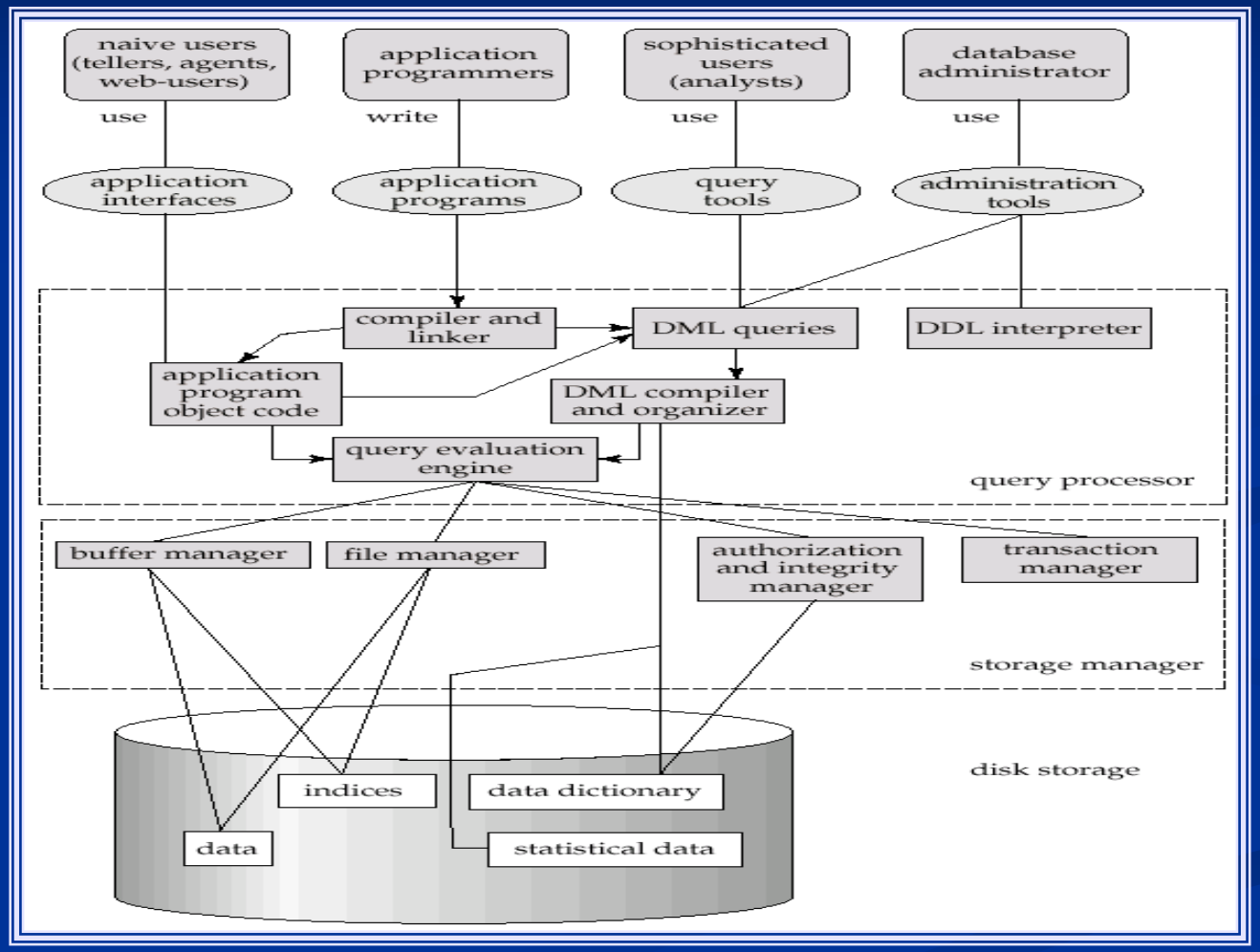
- Pada level fisik, Pegawai dapat dijabarkan sebagai blok data yang terletak dalam memori sekunder.

- Pada lapis konseptual, masing-masing *record* dijabarkan dalam definisi di depan
- Pada lapis pandangan (view), pemakai A boleh mengakses data gaji tetapi pemakai B tidak

Physical Data Independence

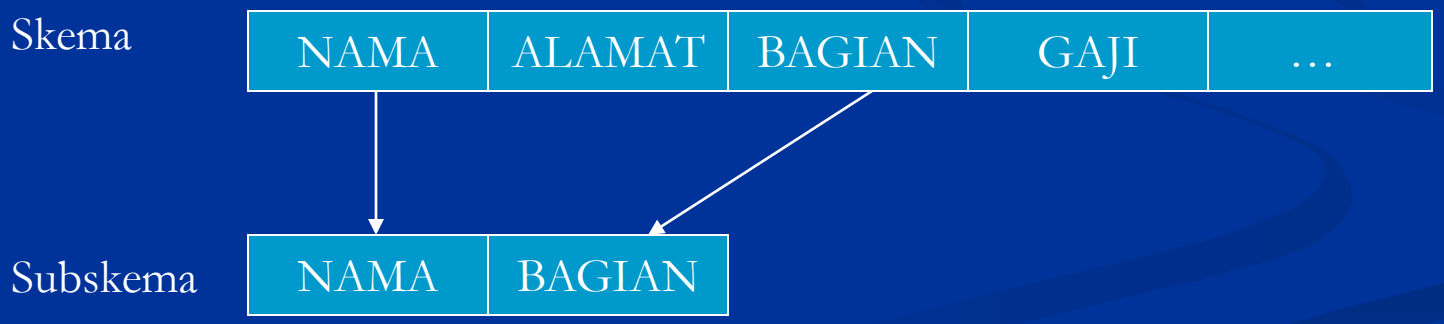
- ***Physical Data Independence*** – kemampuan untuk memodifikasi skema fisik tanpa mengubah skema logika
  - Aplikasi bergantung pada skema logika
  - Antarmuka berbagai level dan komponen harus terdefinisi dengan baik sehingga perubahan pada salah satu bagian tidak mempengaruhi yang lain

Sistem DBMS Secara Menyeluruh



Istilah-Istilah

- DDL (*Data Definition Language*)
  - Perintah-perintah yang biasa dipakai DBA untuk mendefinisikan skema ke DBMS
  - DDL juga dapat dipakai untuk membuat subskema (pandangan bagi pengguna terhadap suatu *database*).



Istilah-Istilah (lanjutan…)

- DDL (*Data Definition Language*)
  - **Contoh:**

```
create table account (
        account-number    char(10),
        balance           integer)
```

- *DDL compiler* membuat perintah seperti di atas disimpan dalam *repository*

Istilah-Istilah (lanjutan…)

- DML (*Data Manipulaton Language*)
  - Perintah-perintah yang digunakan untuk mengubah, memanipulasi, dan mengambil data pada *database*.
  - DML dapat dibagi menjadi 2 kategori:
    - **Prosedural** (menuntut pengguna menentukan data apa saja yang diperlukan dan bagaimana cara mendapatkannya)
    - **Non-prosedural** (menuntut pengguna menentukan data apa saja yang diperlukan, tetapi tidak perlu menyebutkan cara mendapatkannya secara detail)

Istilah-Istilah (lanjutan…)

- Contoh perintah prosedural:

```
OPEN INPUT KARYAWAN.
BACA-BERULANG.
    READ KARYAWAN.
    IF TGL_MASUK GREATER THAN OR EQUAL "01/01/1983"
        DISPLAY NAMA
        GO TO BACA-BERULANG.
    CLOSE KARYAWAN.
```

- Contoh perintah non-prosedural:

```
SELECT NAMA FROM KARYAWAN
    WHERE TGL_MASUK < "1983/01/01".
```

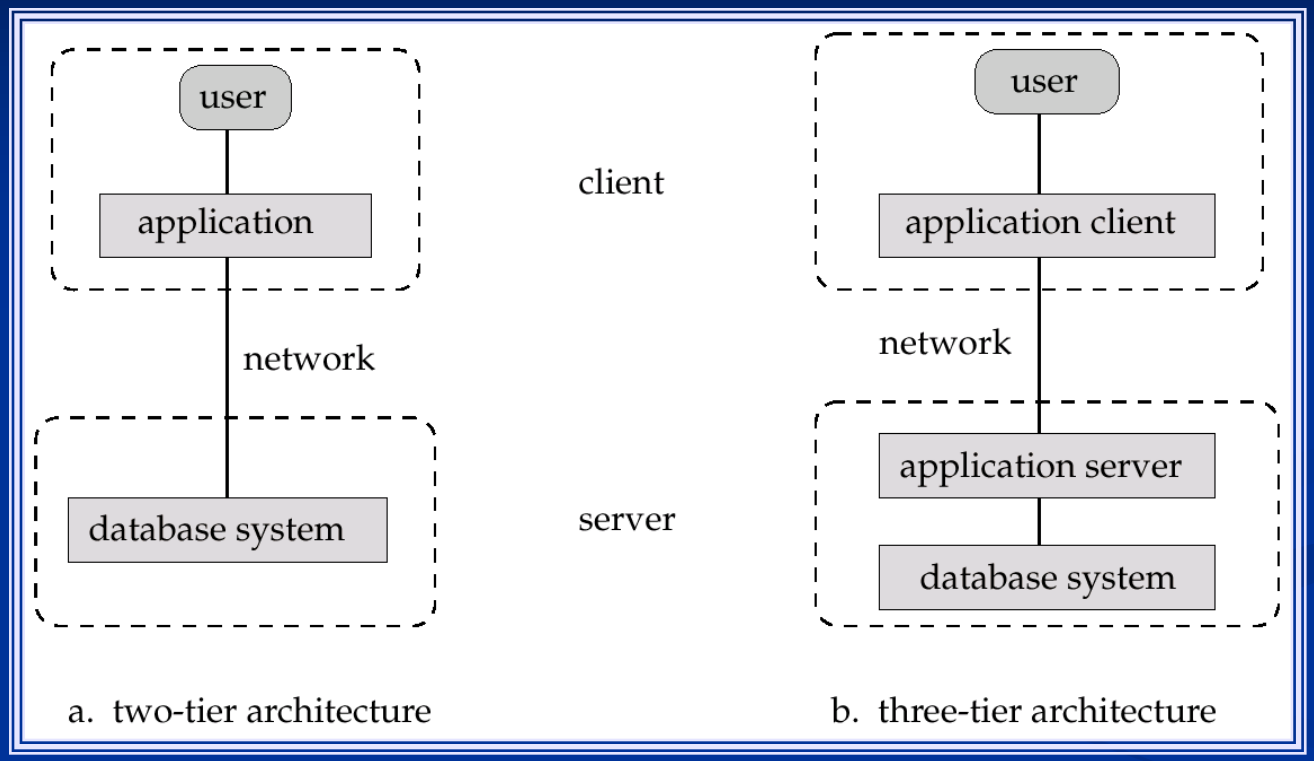
Istilah-Istilah (lanjutan…)

- *Transaction* adalah kumpulan operasi yaang melakukan sebuah fungsi yang utuh dalam suatu aplikasi *database*
- *Transaction-management component* memastikan *database* selalu dalm keadaan yang konsisten meskipun terjadi kegagalan sistem (kegagalan sumber listrik atau sistem operasi) dan kegagalan transaksi
- *Concurrency-control manager* mengendalikan interaksi antara transakasi-transaksi yang berjalan bersamaan, untuk menjamin konsistensi dalam *database*

Istilah-Istilah (lanjutan…)

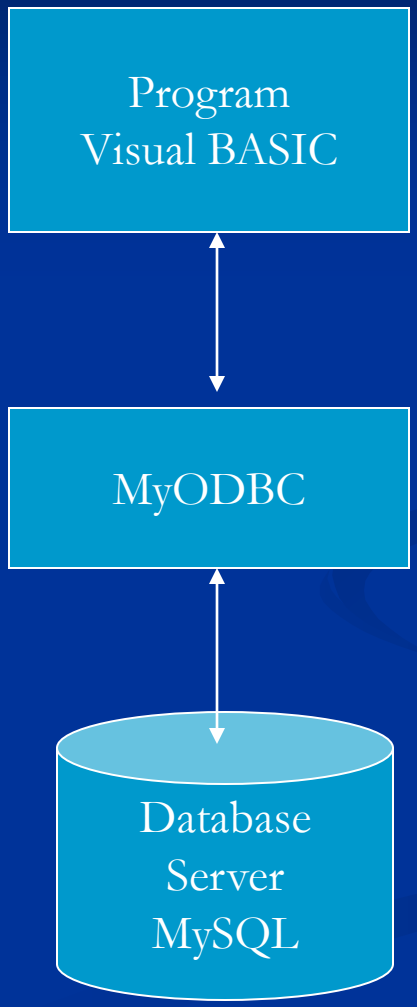
- *Storage manager* adalah modul program yang menyediakan antarmuka antara data tersimpan dalam *database* dan program aplikasi serta *query* yang dikirim ke sistem
- *Storage manager* bertanggung jawab terhadap:
  - Interaksi dengan *file manager*
  - Efisiensi penyimpanan, pengambilan, dan pengubahan data

Application Architectures

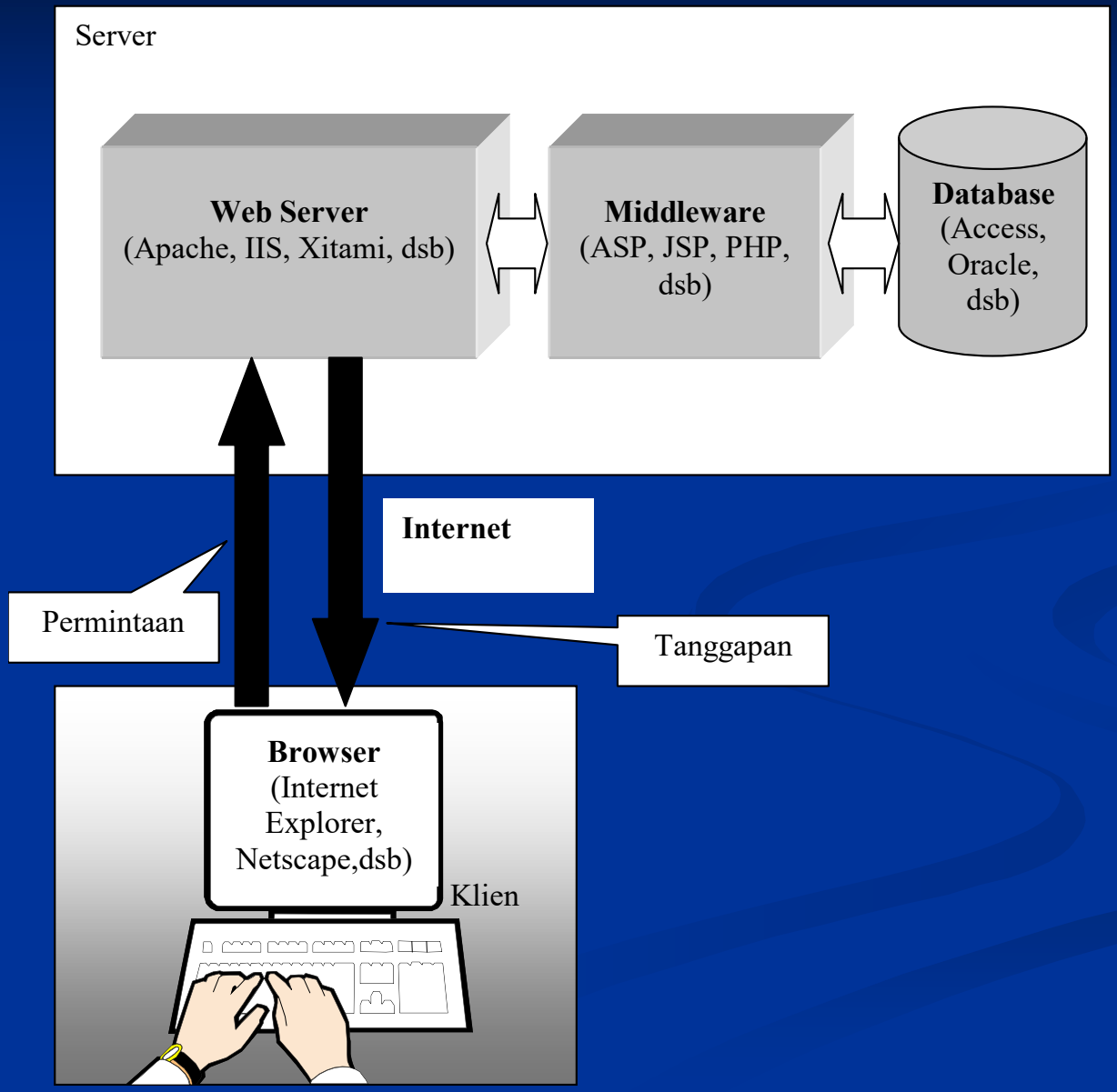


- **Two-tier architecture**: Contoh - program klien menggunakan ODBC/JDBC untuk berkomunikasi dengan database
- **Three-tier architecture**: Contoh aplikasi berbasis Web

Contoh Two-tier Architecture



Contoh Three-tier Architecture




thank you