

LAPORAN PRAKTIKUM 11
PEMROGRAMAN BERBASIS WEB
KELAS A

Disusun Untuk Memenuhi Tugas Mata Kuliah Pemrograman Berbasis Web



Disusun oleh:

Gibran Kahlil

4522210128

Dosen Pengampu:

Adi Wahyu Pribadi , S.Si., M.Kom

Program Studi Teknik Informatika
Fakultas Teknik Universitas Pancasila

2023/2024

Link GitHub: <https://github.com/gibrankahlil260404/PBW/tree/master/p11>

- **Apa itu database:** Database adalah kumpulan data yang terorganisir dan terstruktur dengan baik yang memungkinkan penyimpanan, pengelolaan, dan pengambilan informasi dengan cepat. Sistem Manajemen Basis Data (DBMS) biasanya mengelola database, dan DBMS memiliki berbagai fungsi untuk memasukkan, memperbarui, menghapus, dan mengambil data. Database juga dapat digunakan untuk berbagai tujuan, seperti menyimpan informasi bisnis, catatan medis, transaksi keuangan, dan konten digital.
- **Apa itu SQL dan kegunaannya:** SQL adalah bahasa pemrograman yang dimaksudkan untuk mengelola dan mengubah database relasional. Itu dapat melakukan berbagai operasi, seperti memasukkan data baru, menghapus data, memperbarui data, dan mengambil data dari tabel database. Dengan perintah-perintah seperti SELECT, INSERT, UPDATE, dan DELETE, SQL memungkinkan pengguna melakukan pencarian data yang kompleks, menggabungkan data dari beberapa tabel, dan mengatur dan mengelola database relasional. SQL memiliki banyak manfaat, termasuk mengelola basis data untuk aplikasi bisnis, analisis data, pemrosesan transaksi, dan banyak aplikasi web dan mobile yang memastikan data dapat diakses dan diolah secara terstruktur.

- Source Code 1-sql-query (code ini berisikan perintah untuk menampilkan id, nama, grade dari tabel tb_student sesuai dengan age/umur yang diinginkan, dari code bisa dilihat bahwa umur yang diinginkan yaitu 27 maka akan muncul semua nama yang berumur 27)

```

1-sql-query.go 9+, U X 2-sql-query-row.go 3, U 3-sql-prepare.go 3, U 4-sql-exec.go 4, U 5-sql-prepare-exec.go 4, U tb_student.sql U go.mod p11 1
File Edit Selection View Go Run Terminal Help PBW
> p01
> p02
> p03
> p04
> p05
> p06
> p07
> p08
> p10
p11
  1-sql-q... 9+, U
  2-sql-qu... 3, U
  3-sql-pre... 3, U
  4-sql-exe... 4, U
  5-sql-pre... 4, U
  go.mod 1, U
  gosum U
  tb_student.sql U
p11 > 1-sql-query.go > 58 student
1 package main
2
3 import "fmt"
4 import "database/sql"
5 import _ "github.com/go-sql-driver/mysql"
6
7 type student struct {
8     id string
9     name string
10    age int
11    grade int
12 }
13
14 func connect() (*sql.DB, error) {
15     db, err := sql.Open("mysql", "root:@tcp(127.0.0.1:3306)/db_belajar_golang")
16     if err != nil {
17         return nil, err
18     }
19     return db, nil
20 }
21
22
23 func sqlQuery() {
24     db, err := connect()
25     if err != nil {
26         fmt.Println(err.Error())
27         return
28     }
29     defer db.Close()
30
31     var age = 27
32     rows, err := db.Query("select id, name, grade from tb_student where age = ?", age)
33     if err != nil {
34         fmt.Println(err.Error())
35         return
36     }
37     defer rows.Close()
38
39     var result []student

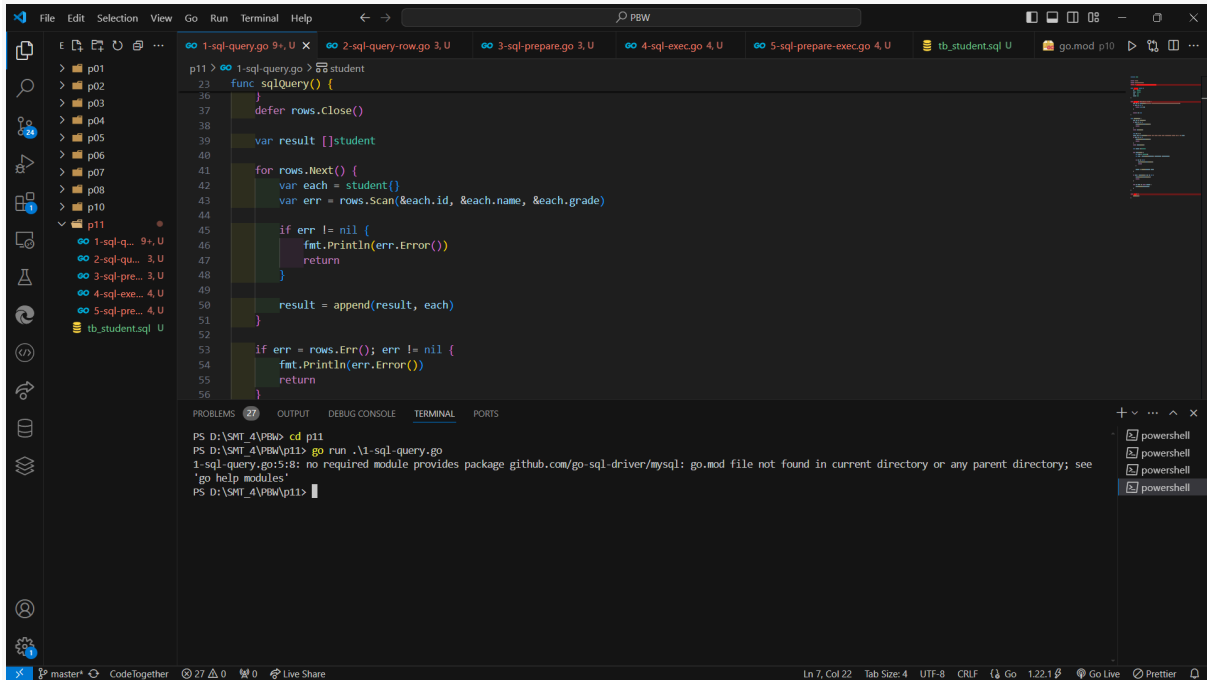
```

```

1-sql-query.go 9+, U X 2-sql-query-row.go 3, U 3-sql-prepare.go 3, U 4-sql-exec.go 4, U 5-sql-prepare-exec.go 4, U tb_student.sql U go.mod p11 1
File Edit Selection View Go Run Terminal Help PBW
p11 > 1-sql-query.go > 58 student
23 func sqlQuery() {
24     db, err := connect()
25     if err != nil {
26         fmt.Println(err.Error())
27         return
28     }
29     defer db.Close()
30
31     var age = 27
32     rows, err := db.Query("select id, name, grade from tb_student where age = ?", age)
33     if err != nil {
34         fmt.Println(err.Error())
35         return
36     }
37     defer rows.Close()
38
39     var result []student
40
41     for rows.Next() {
42         var each = student{}
43         var err = rows.Scan(&each.id, &each.name, &each.grade)
44
45         if err != nil {
46             fmt.Println(err.Error())
47             return
48         }
49
50         result = append(result, each)
51     }
52
53     if err = rows.Err(); err != nil {
54         fmt.Println(err.Error())
55         return
56     }
57
58     for _, each := range result {
59         fmt.Println(each.name)
60     }
61 }
62
63 func main() {
64     sqlQuery()
65 }

```

- Running di terminal (note: sebelum running install terlebih dahulu github.com/go-sql-driver/mysql dengan mengetikan “go install github.com/go-sql-driver/mysql” di terminal, jika tidak maka akan error seperti dibawah ini)

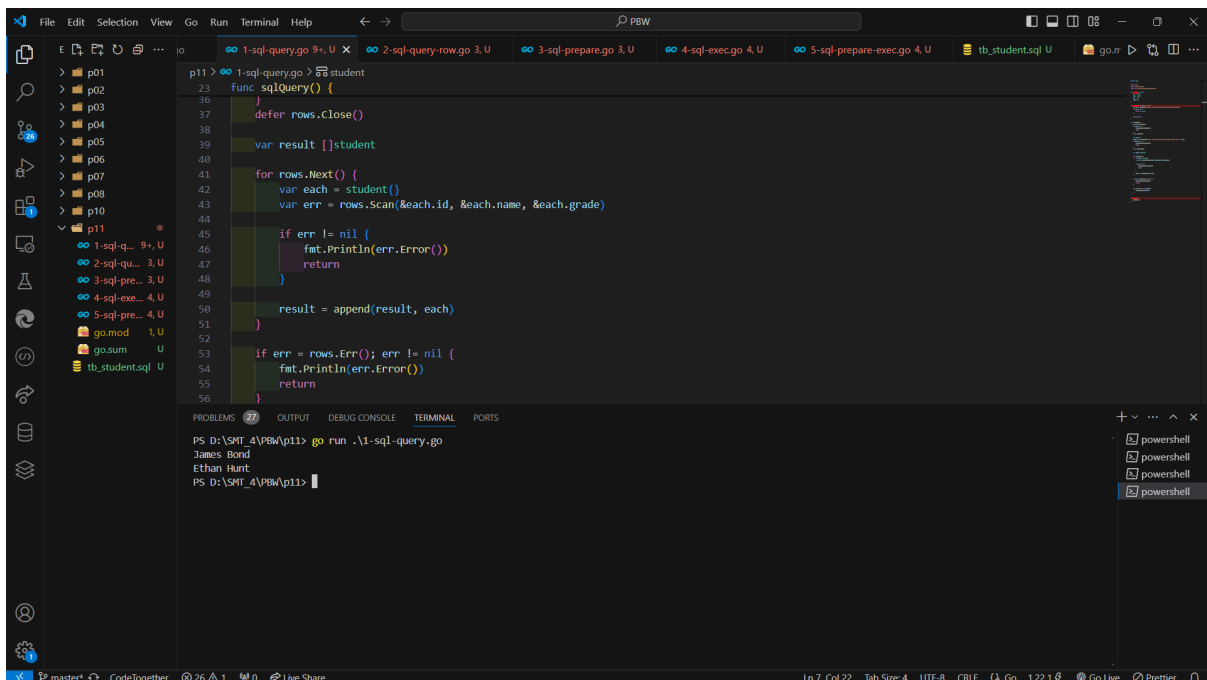


The screenshot shows the VS Code editor with a Go file named `1-sql-query.go` and a terminal window. The Go code defines a `sqlQuery` function that connects to a MySQL database and queries for students. The terminal shows the command `go run .\1-sql-query.go` being executed, which results in an error: `go.mod file not found in current directory or any parent directory; see 'go help modules'`.

```
p11> go run .\1-sql-query.go
23 func sqlQuery() {
36 }
37 defer rows.Close()
38
39 var result []student
40
41 for rows.Next() {
42     var each = student{}
43     var err = rows.Scan(&each.id, &each.name, &each.grade)
44
45     if err != nil {
46         fmt.Println(err.Error())
47         return
48     }
49
50     result = append(result, each)
51 }
52
53 if err = rows.Err(); err != nil {
54     fmt.Println(err.Error())
55     return
56 }
```

```
PS D:\SMT_4\PBW> cd p11
PS D:\SMT_4\PBW\p11> go run .\1-sql-query.go
1-sql-query.go:5:8: no required module provides package github.com/go-sql-driver/mysql; go.mod file not found in current directory or any parent directory; see
'go help modules'
PS D:\SMT_4\PBW\p11>
```

- Hasil running setelah install github.com/go-sql-driver/mysql

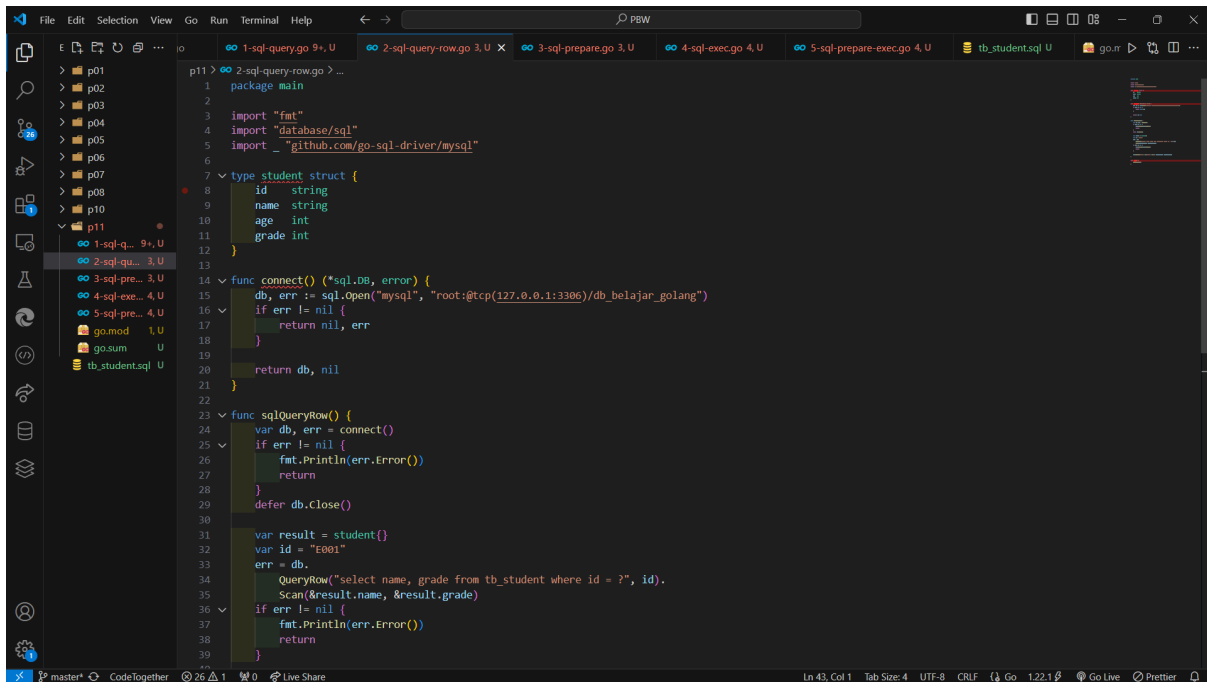


The screenshot shows the same VS Code editor and terminal window as before, but now the terminal output displays the names of the students: `James Bond` and `Ethan Hunt`. This indicates that the `github.com/go-sql-driver/mysql package has been successfully installed and is now available for the program to use.`

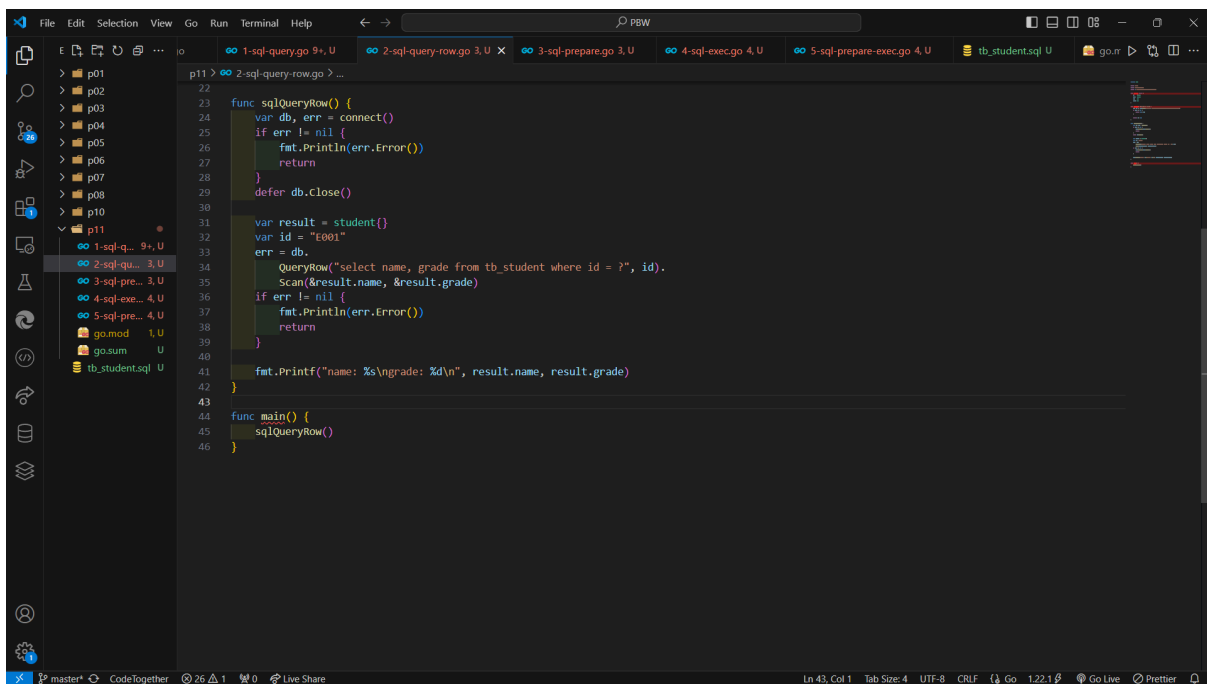
```
p11> go run .\1-sql-query.go
23 func sqlQuery() {
36 }
37 defer rows.Close()
38
39 var result []student
40
41 for rows.Next() {
42     var each = student{}
43     var err = rows.Scan(&each.id, &each.name, &each.grade)
44
45     if err != nil {
46         fmt.Println(err.Error())
47         return
48     }
49
50     result = append(result, each)
51 }
52
53 if err = rows.Err(); err != nil {
54     fmt.Println(err.Error())
55     return
56 }
```

```
PS D:\SMT_4\PBW\p11> go run .\1-sql-query.go
James Bond
Ethan Hunt
PS D:\SMT_4\PBW\p11>
```

- Source Code 2-sql-query-row (code ini berisikan perintah untuk menampilkan nama dan grade dari tabel tb_student sesuai dengan id yang diinginkan yaitu id: E001)



```
1 package main
2
3 import "fmt"
4 import "database/sql"
5 import _ "github.com/go-sql-driver/mysql"
6
7 type student struct {
8     id string
9     name string
10    age int
11    grade int
12 }
13
14 func connect() (*sql.DB, error) {
15    db, err := sql.Open("mysql", "root:@tcp(127.0.0.1:3306)/db_belajar_golang")
16    if err != nil {
17        return nil, err
18    }
19    return db, nil
20 }
21
22 func sqlQueryRow() {
23    var db, err = connect()
24    if err != nil {
25        fmt.Println(err.Error())
26        return
27    }
28    defer db.Close()
29
30    var result = student{}
31    var id = "E001"
32    err = db.
33        QueryRow("select name, grade from tb_student where id = ?", id).
34        Scan(&result.name, &result.grade)
35    if err != nil {
36        fmt.Println(err.Error())
37        return
38    }
39 }
```



```
22
23 func sqlQueryRow() {
24    var db, err = connect()
25    if err != nil {
26        fmt.Println(err.Error())
27        return
28    }
29    defer db.Close()
30
31    var result = student{}
32    var id = "E001"
33    err = db.
34        QueryRow("select name, grade from tb_student where id = ?", id).
35        Scan(&result.name, &result.grade)
36    if err != nil {
37        fmt.Println(err.Error())
38        return
39    }
40
41    fmt.Printf("name: %s\ngrade: %d\n", result.name, result.grade)
42 }
43
44 func main() {
45    sqlQueryRow()
46 }
```

- Hasil running di terminal

```

22
23 func sqlQueryRow() {
24     var db, err = connect()
25     if err != nil {
26         fmt.Println(err.Error())
27         return
28     }
29     defer db.Close()
30
31     var result = student{}
32     var id = "E001"
33     err = db.
34         QueryRow("select name, grade from tb_student where id = ?", id).
35         Scan(&result.name, &result.grade)
36     if err != nil {
37         fmt.Println(err.Error())
38         return
39     }
40     fmt.Printf("name: %s\ngrade: %d\n", result.name, result.grade)
41 }
42
43
PS D:\SMT_4\PBW\p11> go run .\2-sql-query-row.go
name: Ethan Hunt
grade: 2
PS D:\SMT_4\PBW\p11>

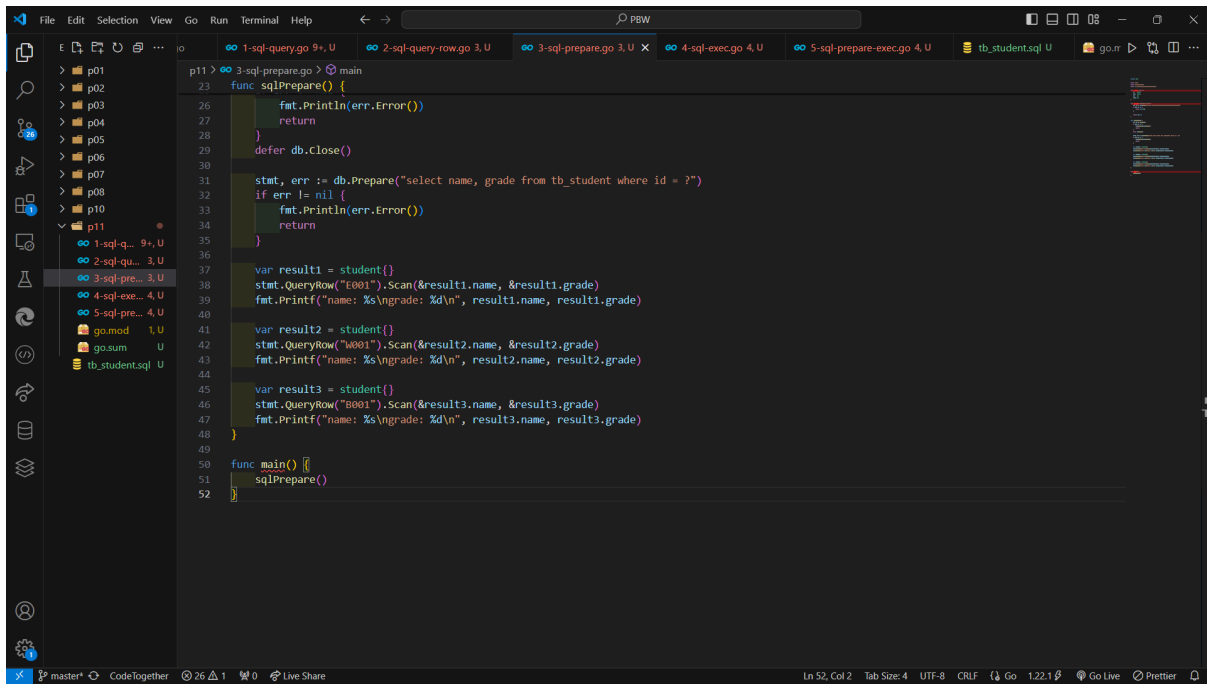
```

- Source Code 3-sql-prepare (code ini berisikan perintah untuk menampilkan nama dan grade dari tabel tb_student sesuai dengan id yang diinginkan yaitu id: E001, W001, B001)

```

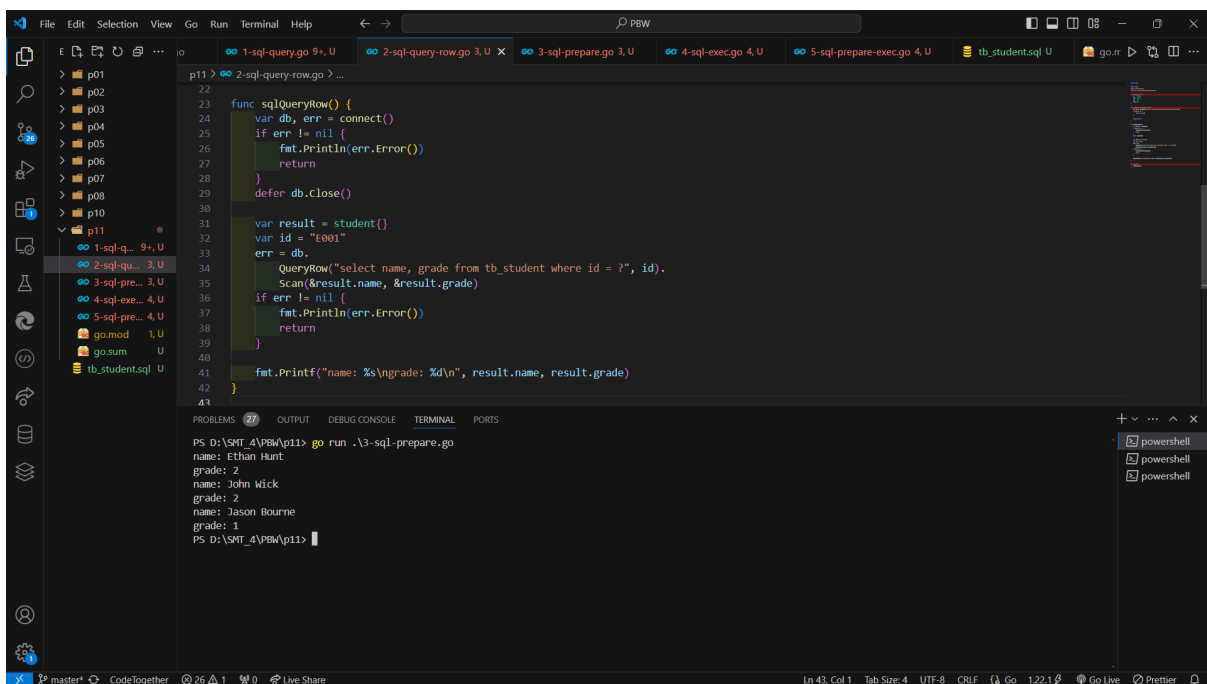
1 package main
2 type DB struct {
3     // ...
4     import "fmt"
5     import "database/sql"
6     import _ "github.com/go-sql-driver/mysql"
7
8     type student struct {
9         id string
10        name string
11        age int
12        grade int
13    }
14
15    func connect() (*sql.DB, error) {
16        db, err := sql.Open("mysql", "root:@tcp(127.0.0.1:3306)/db_belajar_golang")
17        if err != nil {
18            return nil, err
19        }
20        return db, nil
21    }
22
23    func sqlPrepare() {
24        db, err := connect()
25        if err != nil {
26            fmt.Println(err.Error())
27            return
28        }
29        defer db.Close()
30
31        stat, err := db.Prepare("select name, grade from tb_student where id = ?")
32        if err != nil {
33            fmt.Println(err.Error())
34            return
35        }
36
37        var result1 = student{}
38        stat.QueryRow("E001").Scan(&result1.name, &result1.grade)
39        fmt.Printf("name: %s\ngrade: %d\n", result1.name, result1.grade)
40    }

```



```
23 func sqlPrepare() {
24     fmt.Println(err.Error())
25     return
26 }
27 defer db.Close()
28
29 stmt, err := db.Prepare("select name, grade from tb_student where id = ?")
30 if err != nil {
31     fmt.Println(err.Error())
32     return
33 }
34
35 var result1 = student{}
36 stmt.QueryRow("E001").Scan(&result1.name, &result1.grade)
37 fmt.Printf("name: %s\ngrade: %d\n", result1.name, result1.grade)
38
39 var result2 = student{}
40 stmt.QueryRow("E001").Scan(&result2.name, &result2.grade)
41 fmt.Printf("name: %s\ngrade: %d\n", result2.name, result2.grade)
42
43 var result3 = student{}
44 stmt.QueryRow("E001").Scan(&result3.name, &result3.grade)
45 fmt.Printf("name: %s\ngrade: %d\n", result3.name, result3.grade)
46
47 }
48
49 func main() {
50     sqlPrepare()
51 }
52 }
```

- Hasil running di terminal



```
22 func sqlQueryRow() {
23     var db, err = connect()
24     if err != nil {
25         fmt.Println(err.Error())
26         return
27     }
28     defer db.Close()
29
30     var result = student{}
31     var id = "E001"
32     err = db.
33         QueryRow("select name, grade from tb_student where id = ?", id).
34         Scan(&result.name, &result.grade)
35     if err != nil {
36         fmt.Println(err.Error())
37         return
38     }
39     fmt.Printf("name: %s\ngrade: %d\n", result.name, result.grade)
40 }
41
42
43
```

PS D:\SMT_4\PBW\p11> go run .\3-sql-prepare.go

name: Ethan Hunt
grade: 2
name: John Wick
grade: 2
name: Jason Bourne
grade: 1
PS D:\SMT_4\PBW\p11>

- Source Code 4-sql-exec (Code ini berisikan perintah untuk memasukan, memperbarui dan menghapus “memasukan = id:G001, name:Galahad, age: 29, grade: 2, memperbarui = age: 28, id: G001, dan menghapus = id: G001” kedalam database)

```

1 package main
2
3 import "fmt"
4 import "database/sql"
5 import _ "github.com/go-sql-driver/mysql"
6
7 type student struct {
8     id      string
9     name    string
10    age     int
11    grade   int
12}
13
14 func connect() (*sql.DB, error) {
15    db, err := sql.Open("mysql", "root:@tcp(127.0.0.1:3306)/db_belajar_golang")
16    if err != nil {
17        return nil, err
18    }
19    return db, nil
20}
21
22 func sqlExec() {
23    db, err := connect()
24    if err != nil {
25        fmt.Println(err.Error())
26        return
27    }
28    defer db.Close()
29
30    _, err = db.Exec("insert into tb_student values (?, ?, ?, ?)", "G001", "Galahad", 29, 2)
31    if err != nil {
32        fmt.Println(err.Error())
33        return
34    }
35    fmt.Println("insert success!")
36
37    _, err = db.Exec("update tb_student set age = ? where id = ?", "28", "G001")
38    if err != nil {
39        fmt.Println(err.Error())
40    }
41}
42
43 func main() {
44    sqlExec()
45}

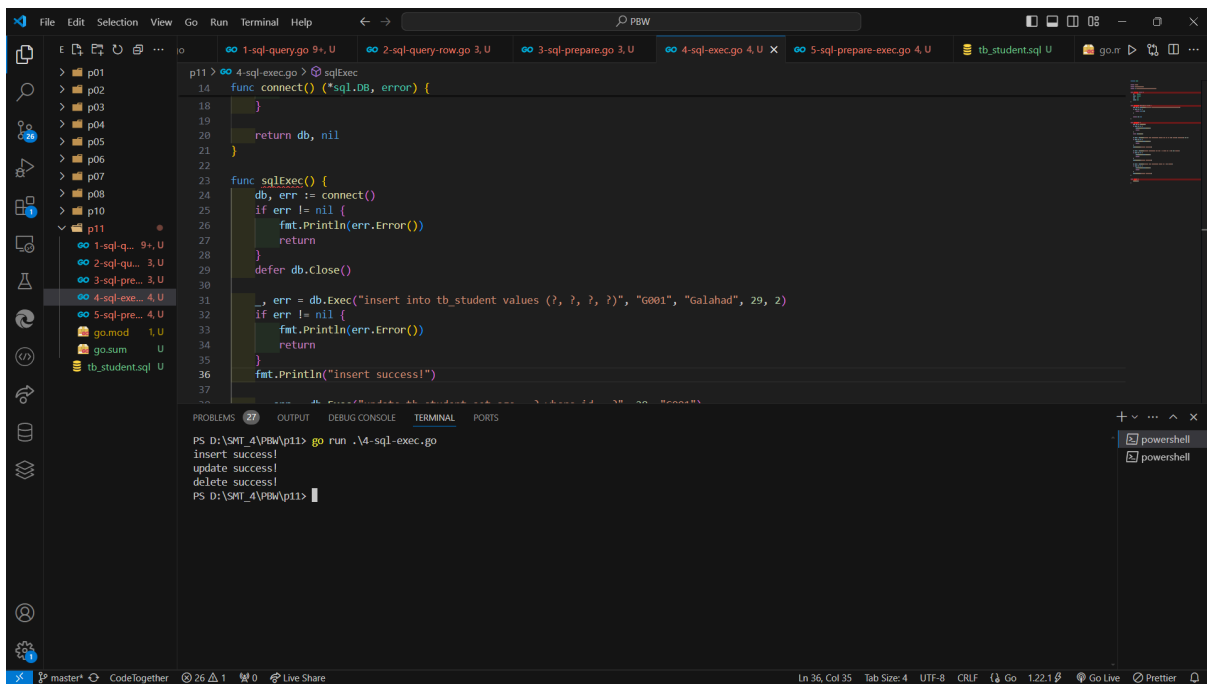
```

```

22 func sqlExec() {
23    db, err := connect()
24    if err != nil {
25        fmt.Println(err.Error())
26        return
27    }
28    defer db.Close()
29
30    _, err = db.Exec("insert into tb_student values (?, ?, ?, ?)", "G001", "Galahad", 29, 2)
31    if err != nil {
32        fmt.Println(err.Error())
33        return
34    }
35    fmt.Println("insert success!")
36
37    _, err = db.Exec("update tb_student set age = ? where id = ?", "28", "G001")
38    if err != nil {
39        fmt.Println(err.Error())
40        return
41    }
42    fmt.Println("update success!")
43
44    _, err = db.Exec("delete from tb_student where id = ?", "G001")
45    if err != nil {
46        fmt.Println(err.Error())
47        return
48    }
49    fmt.Println("delete success!")
50}
51
52 func main() {
53    sqlExec()
54}
55

```


- Hasil running di terminal



```

14 func connect() (*sql.DB, error) {
15     db, err := sql.Open("mysql", "root:@tcp(127.0.0.1:3306)/db_belajar_golang")
16     if err != nil {
17         return nil, err
18     }
19     return db, nil
20 }
21
22 func sqlExec() {
23     db, err := connect()
24     if err != nil {
25         fmt.Println(err.Error())
26         return
27     }
28     defer db.Close()
29
30     stat, err := db.Exec("insert into tb_student values (?, ?, ?, ?)", "G001", "Galahad", 29, 2)
31     if err != nil {
32         fmt.Println(err.Error())
33         return
34     }
35     fmt.Println("insert success!")
36 }
37
38 func main() {
39     sqlExec()
40 }

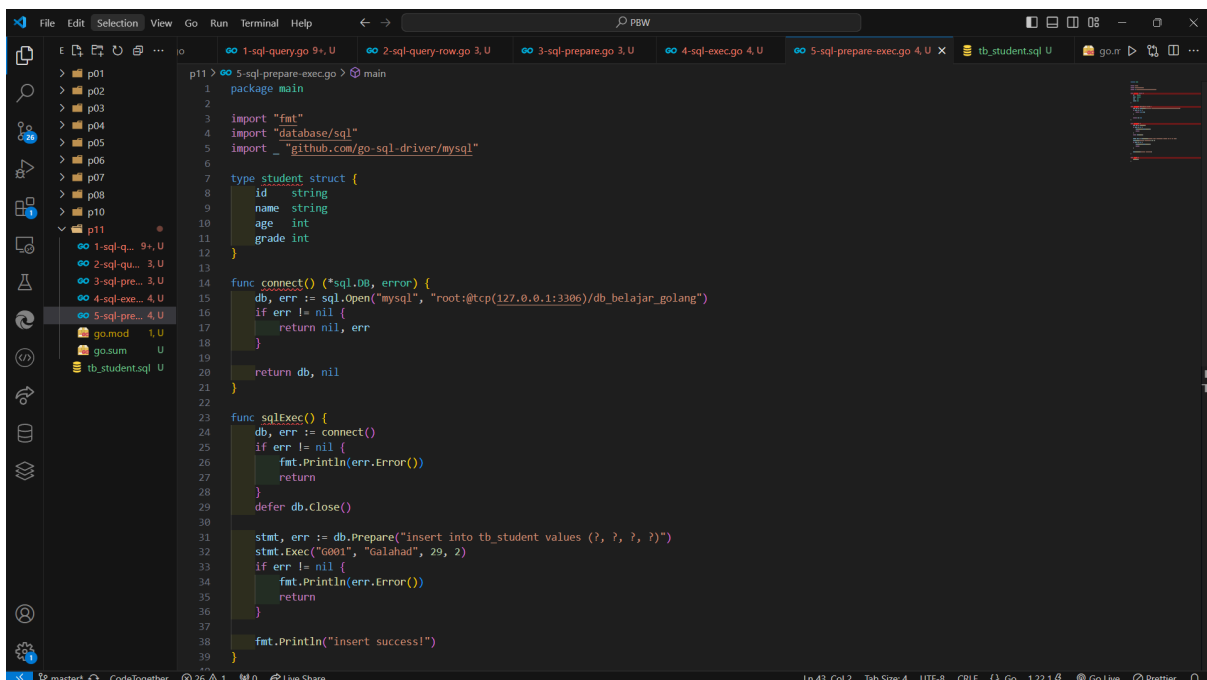
```

```

PS D:\SMT_4\PBW\p11> go run .\4-sql-exec.go
insert success!
update success!
delete success!
PS D:\SMT_4\PBW\p11>

```

- Source Code 5-sql-prepare-exec (Code ini berisikan perintah untuk memasukan “id:G001, name:Galahad, age: 29, grade: 2” kedalam database)



```

1 package main
2
3 import "fmt"
4 import "database/sql"
5 import _ "github.com/go-sql-driver/mysql"
6
7 type student struct {
8     id string
9     name string
10    age int
11    grade int
12 }
13
14 func connect() (*sql.DB, error) {
15     db, err := sql.Open("mysql", "root:@tcp(127.0.0.1:3306)/db_belajar_golang")
16     if err != nil {
17         return nil, err
18     }
19     return db, nil
20 }
21
22 func sqlExec() {
23     db, err := connect()
24     if err != nil {
25         fmt.Println(err.Error())
26         return
27     }
28     defer db.Close()
29
30     stat, err := db.Prepare("insert into tb_student values (?, ?, ?, ?)")
31     if err != nil {
32         fmt.Println(err.Error())
33         return
34     }
35     stat.Exec("G001", "Galahad", 29, 2)
36     fmt.Println("insert success!")
37 }
38
39 func main() {
40     sqlExec()
41 }

```

```
p11> 5-sql-prepare-exec.go > main
14 func connect() (*sql.DB, error) {
15     db, err := sql.Open("mysql", "root:@tcp(127.0.0.1:3306)/db_belajar_golang")
16     if err != nil {
17         return nil, err
18     }
19     return db, nil
20 }
21
22 func sqlExec() {
23     db, err := connect()
24     if err != nil {
25         fmt.Println(err.Error())
26         return
27     }
28     defer db.Close()
29
30     stat, err := db.Prepare("insert into tb_student values (?, ?, ?, ?)")
31     stat.Exec("G001", "Galahad", 29, 2)
32     if err != nil {
33         fmt.Println(err.Error())
34         return
35     }
36     fmt.Println("insert success!")
37 }
38
39 }
40
41 func main() {
42     sqlExec()
43 }
```

- Database sebelum di update

Showing rows 0 - 4 (5 total, Query took 0.0004 seconds)

```
SELECT * FROM `tb_student`
```

Extra options

	id	name	age	grade
<input type="checkbox"/>	B001	Jason Bourne	29	1
<input type="checkbox"/>	B002	James Bond	27	1
<input type="checkbox"/>	E001	Ethan Hunt	27	2
<input type="checkbox"/>	W001	John Wick	28	2

Query results operations

Print Copy to clipboard Export Display chart Create view

Bookmark this SQL query

Label: ☐ Let every user access this bookmark

Bookmark this SQL query

- Hasil running di terminal

The screenshot shows a VS Code editor with a Go file named `5-sql-prepare-exec.go` and a terminal window. The Go code defines a `connect` function to establish a MySQL connection and an `sqlExec` function to execute an SQL insert statement. The terminal output shows the program running successfully and inserting data into the `tb_student` table.

```

14 func connect() (*sql.DB, error) {
15     db, err := sql.Open("mysql", "root:@tcp(127.0.0.1:3306)/db_belajar_golang")
16     if err != nil {
17         return nil, err
18     }
19     return db, nil
20 }
21
22 func sqlExec() {
23     db, err := connect()
24     if err != nil {
25         fmt.Println(err.Error())
26         return
27     }
28     defer db.Close()
29
30     stat, err := db.Prepare("insert into tb_student values (?, ?, ?, ?)")
31     stat.Exec("G001", "Galahad", 29, 2)
32     if err != nil {
33         fmt.Println(err.Error())
34         return
35     }
36 }

```

Terminal Output:

```

PS D:\SMT_4\PBW\p11> go run .\5-sql-prepare-exec.go
insert success!
PS D:\SMT_4\PBW\p11>

```

- Database setelah di update

The screenshot shows the phpMyAdmin interface for the `db_belajar_golang` database. The `tb_student` table is selected, and the SQL tab is active. The table contains 5 rows of data, which are displayed in the table view.

id	name	age	grade
B001	Jason Bourne	29	1
B002	James Bond	27	1
E001	Ethan Hunt	27	2
G001	Galahad	29	2
W001	John Wick	28	2

Kesimpulan: database memiliki banyak fungsi diantaranya seperti menampilkan, memperbarui, menghapus database berupa field, value, kolom. Database juga harus memiliki relasi antar tabelnya agar tidak terjadi kehilangan data atau redundansi data.