ORM()

SQLORMSQL

https://pkg.go.dev/github.com/gogf/gf/v2/database/gdb


```
// SQLsql
Query(ctx context.Context, query string, args ...interface{}) (*sql.Rows,
error)
Exec(ctx context.Context, query string, args ...interface{}) (sql.Result,
error)
Prepare(ctx context.Context, query string) (*sql.Stmt, error)

//
// ()
GetAll(ctx context.Context, sql string, args ...interface{}) (Result,
error)
GetOne(ctx context.Context, sql string, args ...interface{}) (Record,
error)
GetValue(ctx context.Context, sql string, args ...interface{}) (Value,
error)
GetArray(ctx context.Context, sql string, args ...interface{}) ([]Value,
error)
GetCount(ctx context.Context, sql string, args ...interface{}) (int, error)
GetScan(ctx context.Context, objPointer interface{}, sql string, args ...
interface{}) error

//
Insert(ctx context.Context, table string, data interface{}, batch...int)
(sql.Result, error)
Replace(ctx context.Context, table string, data interface{}, batch...int)
(sql.Result, error)
Save(ctx context.Context, table string, data interface{}, batch...int)
(sql.Result, error)

// /
Update(ctx context.Context, table string, data interface{}, condition
interface{}, args ...interface{}) (sql.Result, error)
Delete(ctx context.Context, table string, condition interface{}, args ...
interface{}) (sql.Result, error)
```

1. QueryGet*
2. Exec/SQL
3. Get*
4. Insert/Replace/Savedatastring/map/slice/struct/*structslicebatch

1. ORM

```
// ("default")
db := g.DB()

// "user-center"
db := g.DB("user-center")

//
db, err := gdb.Instance()
db, err := gdb.Instance("user-center")

// Close(gdbClose)
//
```

2.

```
r, err := db.Insert(ctx, "user", gdb.Map {
    "name": "john",
})
```

3. ()

```
list, err := db.GetAll(ctx, "select * from user limit 2")
list, err := db.GetAll(ctx, "select * from user where age > ? and name
like ?", g.Slice{18, "%john%"})
list, err := db.GetAll(ctx, "select * from user where status=?", g.Slice
{1})
```

4. ()

```
one, err := db.GetOne(ctx, "select * from user limit 2")
one, err := db.GetOne(ctx, "select * from user where uid=1000")
one, err := db.GetOne(ctx, "select * from user where uid=?", 1000)
one, err := db.GetOne(ctx, "select * from user where uid=?", g.Slice{1000})
```

5.

```
r, err := db.Save(ctx, "user", gdb.Map {
    "uid"  :  1,
    "name" : "john",
})
```

6.

batch10

```
_, err := db.Insert(ctx, "user", gdb.List {
    {"name": "john_1"},
    {"name": "john_2"},
    {"name": "john_3"},
    {"name": "john_4"},
}, 10)
```

7. /

```
// db.Update/db.Delete
// UPDATE `user` SET `name`='john' WHERE `uid`=10000
r, err := db.Update(ctx, "user", gdb.Map {"name": "john"}, "uid=?", 10000)
// UPDATE `user` SET `name`='john' WHERE `uid`=10000
r, err := db.Update(ctx, "user", "name='john'", "uid=10000")
// UPDATE `user` SET `name`='john' WHERE `uid`=10000
r, err := db.Update(ctx, "user", "name=?", "uid=?", "john", 10000)
```

?SQL