

-gen dao

gen daoCLIdaodoentityGo



gf gen dao

```
$ gf gen dao -h
USAGE
    gf gen dao [OPTION]

OPTION
    -p, --path                  directory path for generated files
    -l, --link                  database configuration, the same as the
ORM configuration of GoFrame
    -t, --tables                generate models only for given tables,
multiple table names separated with ','
    -x, --tablesEx              generate models excluding given tables,
multiple table names separated with ','
    -g, --group                 specifying the configuration group name of
database for generated ORM instance,
                                it's not necessary and the default value
is "default"
    -f, --prefix                add prefix for all table of specified link
/database tables
    -r, --removePrefix          remove specified prefix of the table,
multiple prefix separated with ','
    -rf, --removeFieldPrefix    remove specified prefix of the field,
multiple prefix separated with ','
    -j, --jsonCase              generated json tag case for model struct,
cases are as follows:
                                | Case           | Example           |
                                |-----|-----|
                                | Camel          | AnyKindOfString |
                                | CamelLower     | anyKindOfString |
default
                                | Snake          | any_kind_of_string |
                                | SnakeScreaming | ANY_KIND_OF_STRING |
                                | SnakeFirstUpper | rgb_code_md5      |
                                | Kebab          | any-kind-of-string |
                                | KebabScreaming | ANY-KIND-OF-STRING |
    -i, --importPrefix          custom import prefix for generated go files
    -d, --daoPath               directory path for storing generated dao
files under path
    -o, --doPath                directory path for storing generated do
files under path
    -e, --entityPath            directory path for storing generated
entity files under path
    -t1, --tplDaoIndexPath     template file path for dao index file
    -t2, --tplDaoInternalPath  template file path for dao internal file
    -t3, --tplDaoDoPath         template file path for dao do file
    -t4, --tplDaoEntityPath    template file path for dao entity file
    -s, --stdTime               use time.Time from stdlib instead of gtime.
Time for generated time/date fields of tables
    -w, --withTime              add created time for auto produced go files
    -n, --gJsonSupport          use gJsonSupport to use *gjson.Json
instead of string for generated json fields of
tables
    -v, --overwriteDao          overwrite all dao files both inside
/outside internal folder
    -c, --descriptionTag        add comment to description tag for each
field
    -k, --noJsonTag             no json tag will be added for each field
```

Content Menu

-
-
-
-
- bool

```

-m, --noModelComment      no model comment will be added for each
field
-a, --clear              delete all generated go files that do not
exist in database
-y, --typeMapping        custom local type mapping for generated
struct attributes relevant to fields of table
-h, --help                more information about this command

EXAMPLE
gf gen dao
gf gen dao -l "mysql:root:12345678@tcp(127.0.0.1:3306)/test"
gf gen dao -p ./model -g user-center -t user,user_detail,user_login
gf gen dao -r user_

CONFIGURATION SUPPORT
Options are also supported by configuration file.
It's suggested using configuration file instead of command line
arguments making producing.
The configuration node name is "gfcli.gen.dao", which also supports
multiple databases, for example(config.yaml):
gfcli:
gen:
dao:
- link:      "mysql:root:12345678@tcp(127.0.0.1:3306)/test"
  tables:    "order,products"
  jsonCase:  "CamelLower"
- link:      "mysql:root:12345678@tcp(127.0.0.1:3306)/primary"
  path:      "./my-app"
  prefix:    "primary_"
  tables:   "user, userDetail"
  typeMapping:
    decimal:
      type:    decimal.Decimal
      import:  github.com/shopspring/decimal
    numeric:
      type:    string

```

 makemake dao

/hack/config.yaml

```

gfcli:
gen:
dao:
- link:      "mysql:root:12345678@tcp(127.0.0.1:3306)/test"
  tables:    "order,products"
  jsonCase:  "CamelLower"

- link:      "mysql:root:12345678@tcp(127.0.0.1:3306)/primary"
  path:      "./my-app"
  prefix:    "primary_"
  tables:   "user, userDetail"

# sqlitesqlitegfgf\cmd\gf\internal\cmd\cmd_gen_dao.goimportsqlite
gcc
- link: "sqlite:./file.db"

```

gfcli.gen. dao		dao	-
link		mysql, postgresql, dsn ORM	-
path	internal	daomodel	./app
group	default		default order user
prefix			order_ user_
removePref ix		,	gf_
removeFiel dPrefix		,	f_
tables			user, user_detail
tablesEx		Tables Excluding	product, order
jsonCase	CamelLower	modeljsonCamelLowerSnakeScreamingSnake FirstUpperKebabKebabScreaming	Snake
stdTime	false	time.Time*gtime.Time	true
withTime	false		
gJsonSuppo rt	false	JSON*gjson.Json	true
overwriteD ao	false	daodao/internaldao/internal	true
importPref ix	go.mod	Goimportgen dao	github.com /gogf/gf
descriptio nTag	false	desription	true
noJsonTag	false	json	
noModelCom ment	false		true
clear	false	dao/do/entity	
typeMapping	decimal: type: float64 money: type: float64 numeric: type: float64 smallmone y: type: float64	v2.5 Goimport decimal: type: decimal.Decimal import: github.com/shopspring/decimal	
daoPath	dao	DAO	
doPath	model/do	DO	
entityPath	model /entity	Entity	
tplDaoInde xPath		DAO Index	
tplDaoInte rnalPath		DAO Internal	
tplDaoDoPa th		DO	
tplDaoEnti tyPath		Entity	

<https://github.com/gogf/focus-single>



```
new-stage - category.go

// Code generated by Sdkgen 0.1 tool. DO NOT EDIT.

// +category
// +category_id int64
// +category_name string
// +category_desc string
// +category_contenttype string
// +category_preset string
// +category_order int64
// +category_start string
// +category_end string
// +category_reset string
// +category_createat *time.Time
// +category_updateat *time.Time
// +category_deleteat *time.Time

package entity

import "fmt"

// Category is the go type for table category.
type Category struct {
    id int64 `json:"id" description:"ID,自增主键"`
    name string `json:"name" description:"名称"`
    desc string `json:"desc" description:"描述"`
    contenttype string `json:"contenttype" description:"内容类型, todo, note, article, reply"`
    preset string `json:"preset" description:"预设, 用于生成模板, 不会直接插入数据库, 一个不合法的值"`
    order int64 `json:"order" description:"排序"`
    start string `json:"start" description:"开始时间"`
    end string `json:"end" description:"结束时间"`
    reset string `json:"reset" description:"重置时间, 重新生成模板, 别名为清理时间, 对于文章, 代表归档时间"`
    createat *time.Time `json:"createat" description:"创建时间"`
    updateat *time.Time `json:"updateat" description:"更新时间"`
    deleteat *time.Time `json:"deleteat" description:"删除时间"`
}
```

13da0

/internal/dao		ORMentitydo
/internal/model/do		do <ul style="list-style-type: none">• DAO-• do
/internal/model/entity		

2model

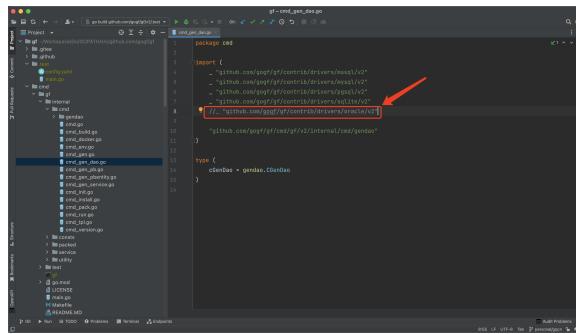
CLI model/entity CLICLI

service model

3daoDAODAOdaoctxCtxTransaction

```
21 func (k contentService) GetList(ctx context.Context, input model.ContentGetListInput) (output *_model.ContentGetListOutput, err error) {
22     var p *ContentGetListOutput
23     p = new(ContentGetListOutput)
24     dao := k.Dao()
25     ctx := ctx.Value("ContentBao")
26     if ctx == nil {
27         return p, errors.New("ContentBao is nil")
28     }
29     output := &ContentGetListOutput{
30         Columns: ContentBao.Columns,
31         Page: &ContentBao.Page,
32         Size: &ContentBao.Size,
33         Obj: &ContentBao.Obj,
34     }
35     // 调用
36     p, err = TransactionInContext(ctx, F(func(ctx context.Context, tx *gdb.TX) error) {
37         return k.GetList(tx, input)
38     })
39 }
```

gen daoOracleCLICGC



bool

```
boolbit(1)boolgen daobit(1)booltinyint(1)bool
```

status	VARCHAR(50)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	"
uniq_instanc...	VARCHAR(64)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
route_type	INT(10)	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	'1'				
network_type	VARCHAR(32)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	"
● is_ipv6	BIT(1)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

```
25      Status      string      `description:"runing/offlined"`
26      UniqInstanceId string      `description:"网络绑定唯一ID"`
27      RouteType     uint       `description:"路由类型: 0 基础网络 1 vpc 网络"`
28      NetworkType   string      `description:"网络类型: vpc"`
29      IsIpv6       bool       `description:"是否IPv6"`
30      Component    string      `description:"组件名称"`

IsIpv6 有红色框标注，带有指向它的箭头。
```