ORM-

Content Menu

-
-
  -
  -
-

Fields/Data/Scanmap/struct/

```
SHOW FULL COLUMNS FROM `xxx`
```

```
Map
nickname   nickname      match
NICKNAME   nickname      match
Nick-Name  nickname      match
nick_name  nickname      match
nick name  nickname      match
NickName   nickname      match
Nick-name  nickname      match
nick_name  nickname      match
nick name  nickname      match
```

DBTableFields/driver

```
// TableFields retrieves and returns the fields' information of specified
table of current
// schema.
//
// The parameter `link` is optional, if given nil it automatically
retrieves a raw sql connection
// as its link to proceed necessary sql query.
//
// Note that it returns a map containing the field name and its
corresponding fields.
// As a map is unsorted, the TableField struct has an "Index" field marks
its sequence in
// the fields.
//
// It's using cache feature to enhance the performance, which is never
expired util the
// process restarts.
func (db DB) TableFields(ctx context.Context, table string, schema ...
string) (fields map[string]*TableField, err error)
```

```
// ClearTableFields removes certain cached table fields of current
configuration group.
func (c *Core) ClearTableFields(ctx context.Context, table string, schema
...string) (err error)

// ClearTableFieldsAll removes all cached table fields of current
configuration group.
func (c *Core) ClearTableFieldsAll(ctx context.Context) (err error)
```

CoreCoreDBCore

```
g.DB().GetCore()
```

1user30doctor_user80

2userdoctor_useruseruser

3GRPC

1GetDoctorInfoRes

```
//
type GetDoctorInfoRes struct {
        UserInfo             *UserInfo   `protobuf:"bytes,1,opt,
name=UserInfo,proto3" json:"UserInfo,omitempty"`
        DoctorInfo           *DoctorInfo `protobuf:"bytes,2,opt,
name=DoctorInfo,proto3" json:"DoctorInfo,omitempty"`
        XXX_NoUnkeyedLiteral struct{}    `json:"-"`
        XXX_unrecognized     []byte      `json:"-"`
        XXX_sizecache        int32       `json:"-"`
}
```

2UserInfo

```
//
type UserInfo struct {
        Id                   uint32   `protobuf:"varint,1,opt,name=id,
proto3" json:"id,omitempty"`
        Avatar               string   `protobuf:"bytes,2,opt,name=avatar,
proto3" json:"avatar,omitempty"`
        Name                 string   `protobuf:"bytes,3,opt,name=name,
proto3" json:"name,omitempty"`
        Sex                  int32    `protobuf:"varint,4,opt,name=sex,
proto3" json:"sex,omitempty"`
        XXX_NoUnkeyedLiteral struct{} `json:"-"`
        XXX_unrecognized     []byte   `json:"-"`
        XXX_sizecache        int32    `json:"-"`
}
```

3DoctorInfo

```
//
type DoctorInfo struct {
        Id                   uint32   `protobuf:"varint,1,opt,name=id,
proto3" json:"id,omitempty"`
        Name                 string   `protobuf:"bytes,3,opt,name=name,
proto3" json:"name,omitempty"`
        Hospital             string   `protobuf:"bytes,4,opt,name=hospital,
proto3" json:"hospital,omitempty"`
        Section              string   `protobuf:"bytes,6,opt,name=section,
proto3" json:"section,omitempty"`
        Title                string   `protobuf:"bytes,8,opt,name=title,
proto3" json:"title,omitempty"`
        XXX_NoUnkeyedLiteral struct{} `json:"-"`
        XXX_unrecognized     []byte   `json:"-"`
        XXX_sizecache        int32    `json:"-"`
}
```

```
//
func (s *Service) GetDoctorInfo(ctx context.Context, req *pb.
GetDoctorInfoReq) (res *pb.GetDoctorInfoRes, err error) {
    // Protobuf
        res = &pb.GetDoctorInfoRes{}
        //
    // SELECT `id`,`avatar`,`name`,`sex` FROM `user` WHERE `user_id`=xxx
        err = dao.PrimaryDoctorUser.
                Ctx(ctx).
                Fields(res.DoctorInfo).
                Where(dao.PrimaryDoctorUser.Columns.UserId, req.Id).
                Scan(&res.DoctorInfo)
        if err != nil {
                return
        }
        //
    // SELECT `id`,`name`,`hospital`,`section`,`title` FROM `doctor_user`
WHERE `id`=xxx
        err = dao.PrimaryUser.
                Ctx(ctx).
                Fields(res.DoctorInfo).
                Where(dao.PrimaryUser.Columns.Id, req.Id).
                Scan(&res.UserInfo)
        return res, err
}
```

GetDoctorInfoSQL

```
SELECT `id`,`avatar`,`name`,`sex` FROM `user` WHERE `user_id`=1
SELECT `id`,`name`,`hospital`,`section`,`title` FROM `doctor_user` WHERE
`id`=1
```

- Fieldsstruct*structORMstruct
- ScanStruct/Structs*struct**structstruct