

# Embarcadero Conference 2024

---

Inovação faz parte do nosso DNA!

# Migration: Sua estrutura de banco sempre atualizada

{Gabriel Baltazar

Ainda conecta remotamente no cliente para rodar script no banco para atualizar versão do seu software? Ainda passa por problemas em desenvolvimento quando vai rodar seu projeto e seu banco está desatualizado? Vamos resolver de forma bem simples esses problemas, utilizando a técnica de migration dentro dos seus projetos Delphi.





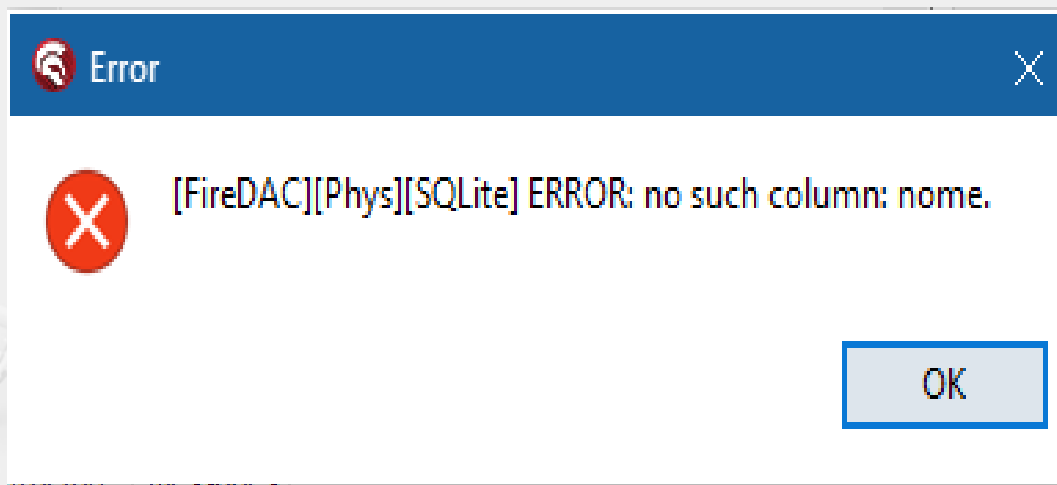
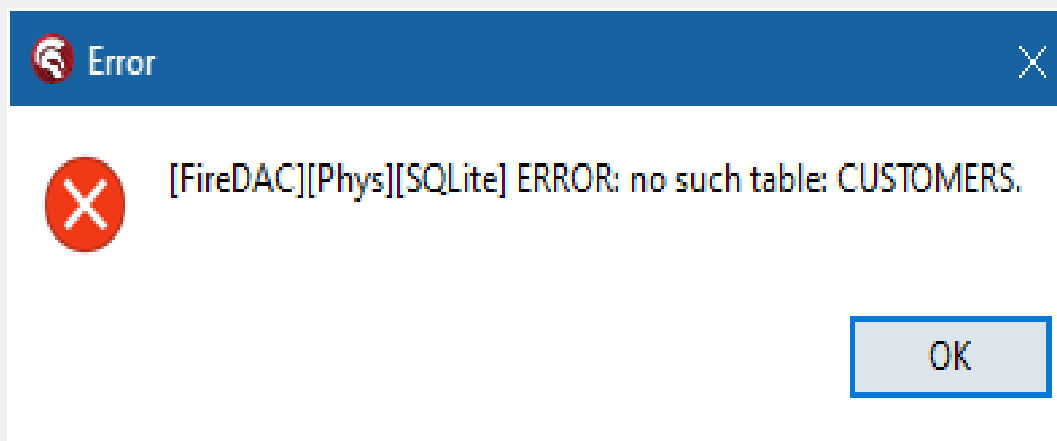
## Gabriel Baltazar

- Bacharel em Sistemas de Informação em 2010.
- Certificação Delphi Developer.
- Desenvolvimento com Delphi há mais de 10 anos.
- Atuante em projetos Open Source como Horse e ACBr.





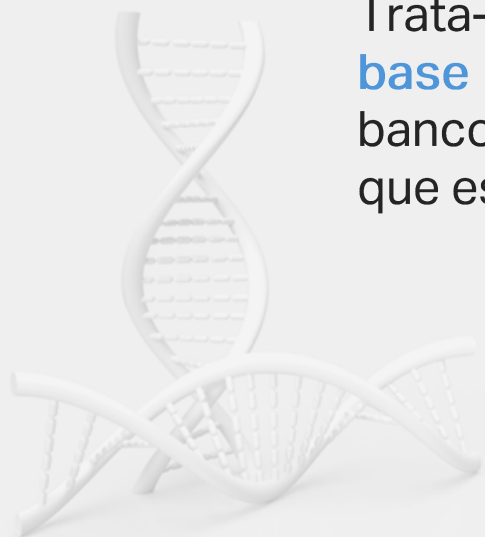
## Quem nunca?





## Migration?

- Schema Migration
- Database Migration
- Migration



Trata-se de técnicas e ferramentas que auxiliam no **versionamento da base de dados durante o desenvolvimento** e fazem as atualizações no banco por meio da **própria linguagem de programação e frameworks** que estejamos utilizando.



## Vantagens


- Melhor integração entre os times.
- Histórico de atualização.
- Rollback de versão.
- Livre de componentes e de banco de dados.





## Desvantagem

- Exige mais criticidade ao subir um commit no projeto.



```
M4DDBDemo.dproj - Projects
ProjectGroup1
  M4DDBDemo.exe
    Build Configurations (Debug)
    Target Platforms (Windows 32-bit)
    Migrations
      MCreatePopulateTables.pas
      MCreateTables.pas
      MigrationInit.pas
    units

M4DDBDemo
MCreateTables
  TCreateTables.Up

53
54 procedure TCreateTables.Up;
55 var
56   LQuery: TFDQuery;
57 begin
58   LQuery := DMDBDemo.GetQuery('DROP TABLE CUSTOMERS', False);
59   try
60     LQuery.ExecSQL;
61   finally
62     LQuery.Free;
63   end;
64
```





## Migrations - Frameworks



**M4D**

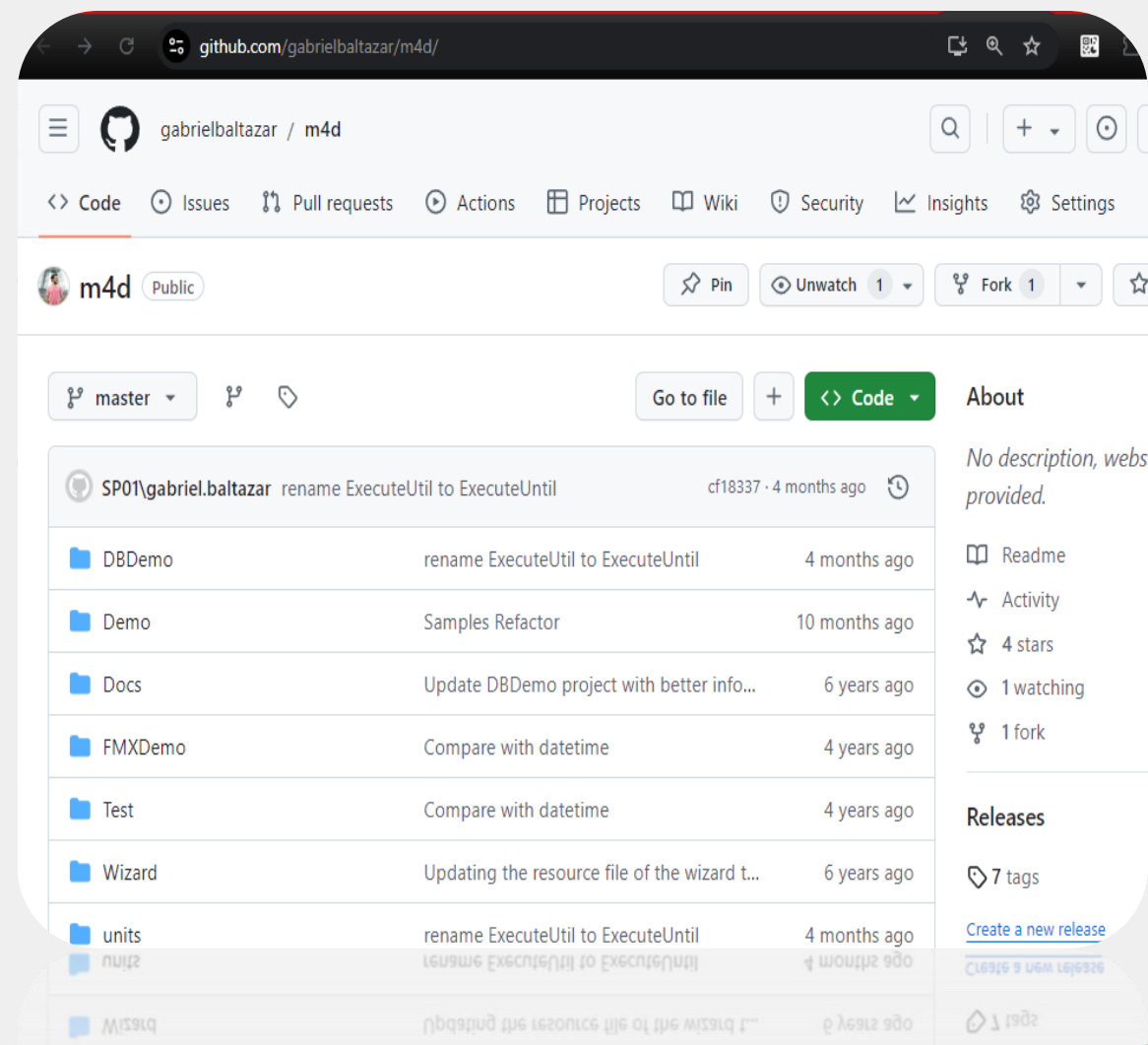




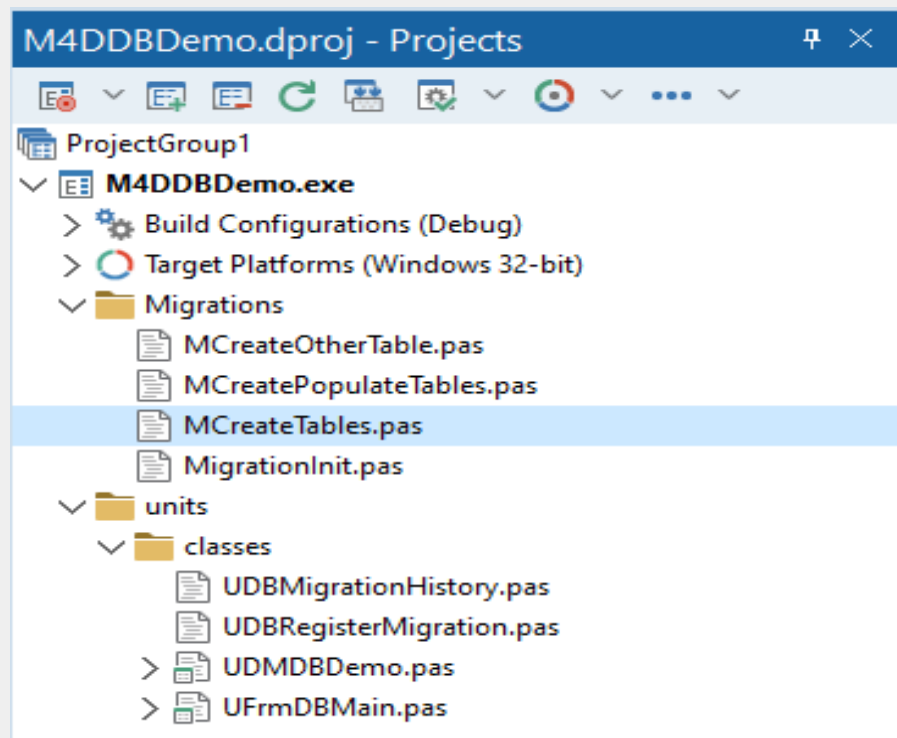


# M4D

- Projeto original de autoria do Edgar Pavão.
- <https://bitbucket.org/migration4d/m4d/src/master>
- Fork no meu github pessoal com algumas implementações a mais.
- <https://github.com/gabrielbaltazar/m4d/>



# Como funciona?



```
type
TCreateTables = class(TMigrations)
public
    procedure Setup; override;
    procedure Up; override;
    procedure Down; override;
end;
```

```
type
TCreateTables = class(TMigrations)
public
    procedure Setup; override;
    procedure Up; override;
    procedure Down; override;
end;

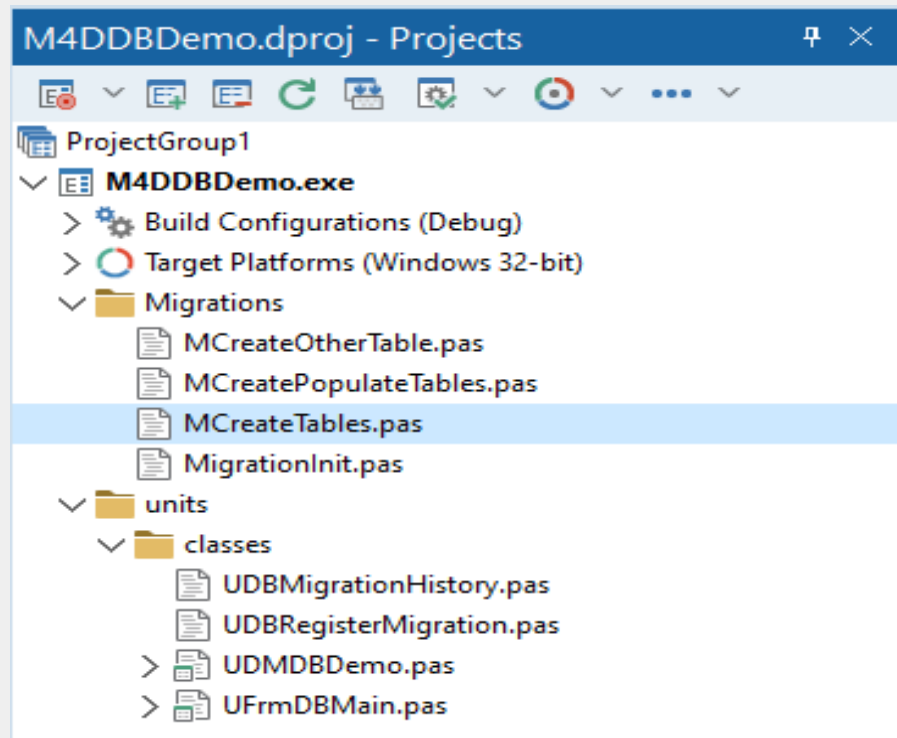
implementation

uses
    System.SysUtils, UDMDBDemo, FireDAC.Comp.Client, UDBMigrationHistory,
    M4D.MigrationsHistoryFacadeInterface;

{ TMDescription1 }

procedure TCreateTables.Setup;
begin
    Self.Version := '1.00';
    Self.SeqVersion := 1;
    Self.DateTime := StrToDateTime('05/09/2017 07:19:00');
end;
```

# Como funciona?

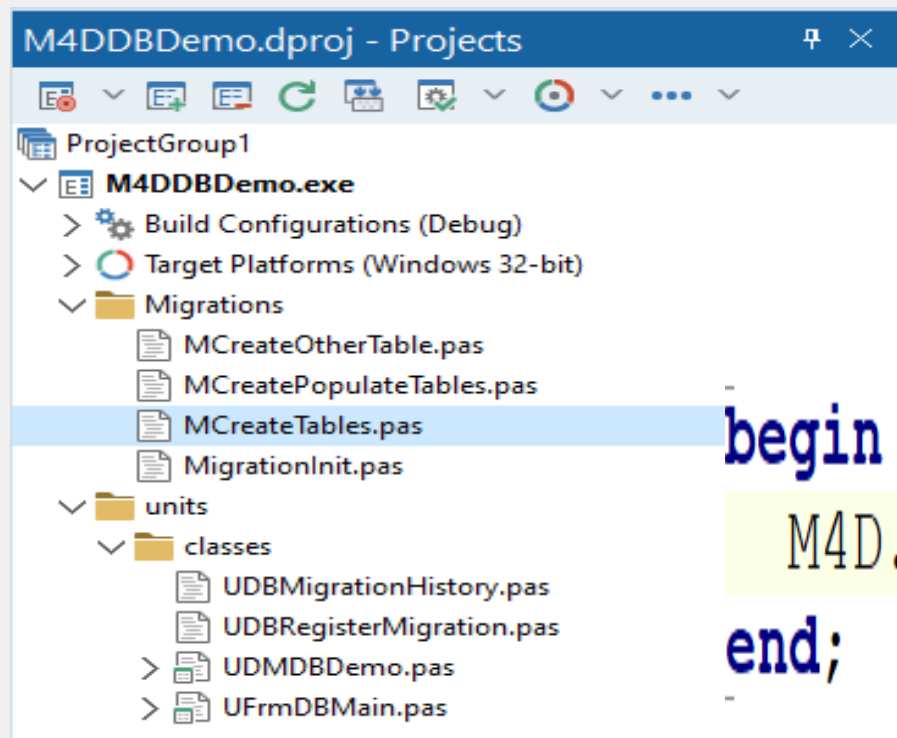


```
procedure TMCreatetables.Up;  
var  
    LQuery: TFDQuery;  
begin  
    LQuery := DMDBDemo.GetQuery('CREATE TABLE IF NOT EXISTS CUSTOMERS ' +  
        '(ID INTEGER NOT NULL, NAME VARCHAR(100) NOT NULL)', False);  
    try  
        LQuery.ExecSQL;  
    finally  
        LQuery.Free;  
    end;  
end;
```

```
type  
TMCreatetables = class(TMigrations)  
public  
    procedure Setup; override;  
    procedure Up; override;  
    procedure Down; override;  
end;
```

```
procedure TMCreatetables.Down;  
var  
    LQuery: TFDQuery;  
begin  
    LQuery := DMDBDemo.GetQuery('DROP TABLE CUSTOMERS', False);  
    try  
        LQuery.ExecSQL;  
    finally  
        LQuery.Free;  
    end;  
end;
```

# Como funciona?



```
begin
```

```
M4D.MigrationFacade.ExecutePending;
```

```
end;
```

```
procedure Execute;  
procedure ExecutePending;  
procedure ExecuteUntil(AMigrationSequence: Integer);  
procedure ExecuteRange(AStartMigrationSequence: Integer; AEndMigrationSequence: Integer..  
procedure Rollback;  
procedure RollbackPending;  
procedure RollbackUntil(AMigrationSequence: Integer);  
procedure RollbackRange(AStartMigrationSequence: Integer; AEndMigrationSequence: Integer..
```



# Como funciona?

M4D.MigrationFacade.

```
procedure Execute;  
procedure ExecutePending;  
procedure ExecuteUntil(AMigrationSequence: Integer);  
procedure ExecuteRange(AStartMigrationSequence: Integer; AEndMigrationSequence: Integer..  
procedure Rollback;  
procedure RollbackPending;  
procedure RollbackUntil(AMigrationSequence: Integer);  
procedure RollbackRange(AStartMigrationSequence: Integer; AEndMigrationSequence: Integer..
```



```
IMigrationsHistoryFacade  
IMigrationsHistoryFacade = interface  
    ['{639BAD0D-0273-4EB2-B129-D9CB70B1B108}']  
    IMigrationExecutorFacade.Clear  
    procedure Clear;  
    IMigrationsHistoryFacade.Load  
    procedure Load;  
    IMigrationsHistoryFacade.UnLoad  
    procedure UnLoad;  
    IMigrationsHistoryFacade.Add  
    procedure Add(AItem: TMigrationsHistoryItem);  
    IMigrationsHistoryFacade.Remove  
    procedure Remove(AMigrationSequence: Integer);  
    IMigrationsHistoryFacade.getHistory  
    function getHistory: TList<TMigrationsHistoryItem>; overload;  
    IMigrationsHistoryFacade.getHistory  
    function getHistory(AStartMigrationSeq: Integer): TList<TMigrationsHistoryItem>;  
    IMigrationsHistoryFacade.getHistory  
    function getHistory(AStartMigrationDateTime: TDateTime): TList<TMigrationsHistoryItem>;  
    IMigrationsHistoryFacade.Save  
    procedure Save;  
    IMigrationsHistoryFacade.LastMigration  
    function LastMigration: TMigrationsHistoryItem;  
  
    property HistoryList: TList<TMigrationsHistoryItem> read getHistory;  
end;
```



# Como funciona?



```
M4D.MigrationFacade.ExecutePending;
```

```
IMigrationsHistoryFacade
IMigrationsHistoryFacade = interface
['{639BAD0D-0273-4EB2-B129-D9CB70B1B108}']
IMigrationExecutorFacade.Clear
procedure Clear;
IMigrationsHistoryFacade.Load
procedure Load;
IMigrationsHistoryFacade.UnLoad
procedure UnLoad;
IMigrationsHistoryFacade.Add
procedure Add(AItem: TMigrationsHistoryItem);
IMigrationsHistoryFacade.Remove
procedure Remove(AMigrationSequence: Integer);
IMigrationsHistoryFacade.getHistory
function getHistory: TList<TMigrationsHistoryItem>; overload;
IMigrationsHistoryFacade.getHistory
function getHistory(AStartMigrationSeq: Integer): TList<TMigrationsHistoryItem>;
IMigrationsHistoryFacade.getHistory
function getHistory(AStartMigrationDateTime: TDateTime): TList<TMigrationsHistoryItem>;
IMigrationsHistoryFacade.Save
procedure Save;
IMigrationsHistoryFacade.LastMigration
function LastMigration: TMigrationsHistoryItem;

property HistoryList: TList<TMigrationsHistoryItem> read getHistory;
end;
```

Database: MeuERP Table: MIGRATIONS\_INFO File: C:\workspace\Delphi\Apresentacoes\EmbarcaderoConferenc

rowid	SEQUENCE	VERSION	
1	1	Create Table Empresa	2
2	2	Alter Table Empresa Add Cnpj	2

```
function TDBMigrationsHistory.LastMigration: TMigrationsHistoryItem;
begin
  if Assigned(FLastMigration) then
    FreeAndNil(FLastMigration);

  Result := nil;
  FQuery.Close;
  FQuery.SQL.Text := 'SELECT SEQUENCE, VERSION, DATETIME, ' +
    'START_OF_EXECUTION, END_OF_EXECUTION, DURATION_OF_EXECUTION ' +
    'FROM MIGRATIONS_INFO ' +
    'ORDER BY SEQUENCE DESC ' +
    'LIMIT 1';

  try
    FQuery.Open;
    if FQuery.RecordCount > 0 then
    begin
      FLastMigration := TMigrationsHistoryItem.Create;
      FLastMigration.MigrationSeq := FQuery.Fields[0].AsInteger;
      FLastMigration.MigrationVersion := FQuery.Fields[1].AsString;
      FLastMigration.MigrationDateTime := FQuery.Fields[2].AsDateTime;
      FLastMigration.StartOfExecution := FQuery.Fields[3].AsDateTime;
      FLastMigration.EndOfExecution := FQuery.Fields[4].AsDateTime;
      FLastMigration.DurationOfExecution := FQuery.Fields[5].AsFloat;
      Result := FLastMigration;
    end;
  end;
```



# Toque do cheff



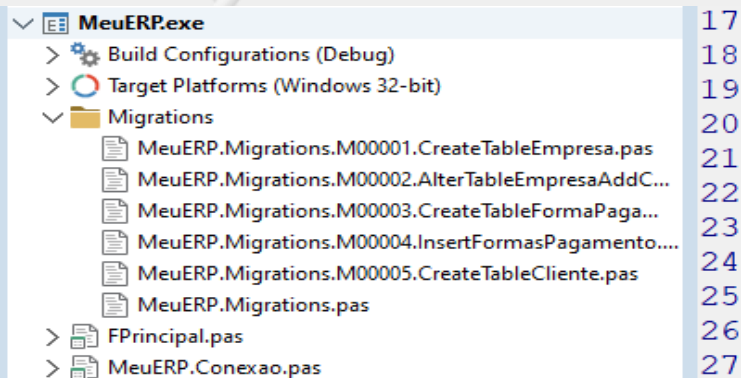
```
M4D.MigrationFacade.ExecutePending;
```

```
IMigrationsHistoryFacade
IMigrationsHistoryFacade = interface
['{639BAD0D-0273-4EB2-B129-D9CB70B1B108}']
IMigrationExecutorFacade.Clear
procedure Clear;
IMigrationsHistoryFacade.Load
procedure Load;
IMigrationsHistoryFacade.UnLoad
procedure UnLoad;
IMigrationsHistoryFacade.Add
procedure Add(AItem: TMigrationsHistoryItem);
IMigrationsHistoryFacade.Remove
procedure Remove(AMigrationSequence: Integer);
IMigrationsHistoryFacade.getHistory
function getHistory: TList<TMigrationsHistoryItem>; overload;
IMigrationsHistoryFacade.getHistory
function getHistory(AStartMigrationSeq: Integer): TList<TMigrationsHistoryItem>;
IMigrationsHistoryFacade.getHistory
function getHistory(AStartMigrationDateTime: TDateTime): TList<TMigrationsHistoryItem>;
IMigrationsHistoryFacade.Save
procedure Save;
IMigrationsHistoryFacade.LastMigration
function LastMigration: TMigrationsHistoryItem;

property HistoryList: TList<TMigrationsHistoryItem> read getHistory;
end;
```

```
function TDBMigrationsHistory.LastMigration: TMigrationsHistoryItem;
begin
    if Assigned(FLastMigration) then
        FreeAndNil(FLastMigration);

    Result := nil;
    FQuery.Close;
    FQuery.SQL.Text := 'SELECT SEQUENCE, VERSION, DATETIME, ' +
        'START_OF_EXECUTION, END_OF_EXECUTION, DURATION_OF_EXECUTION ' +
        'FROM MIGRATIONS_INFO ' +
        'ORDER BY SEQUENCE DESC ' +
        'LIMIT 1';
    try
        FQuery.Open;
        if FQuery.RecordCount > 0 then
            begin
                FLastMigration := TMigrationsHistoryItem.Create;
                FLastMigration.MigrationSeq := FQuery.Fields[0].AsInteger;
                FLastMigration.MigrationVersion := FQuery.Fields[1].AsString;
                FLastMigration.MigrationDateTime := FQuery.Fields[2].AsDateTime;
                FLastMigration.StartOfExecution := FQuery.Fields[3].AsDateTime;
                FLastMigration.EndOfExecution := FQuery.Fields[4].AsDateTime;
                FLastMigration.DurationOfExecution := FQuery.Fields[5].AsFloat;
                Result := FLastMigration;
            end;
    end;
```



```
17
18
19 begin
20     Application.Initialize;
21     Application.MainFormOnTaskbar := True;
22     Application.CreateForm(TForm1, Form1);
23     Application.CreateForm(TMeuERPConexao, MeuERPConexao);
24     TM4DRegistryMigrations.GetInstance
25         .History(TM4DMigrationsHistoryFiredac.New (MeuERPConexao.FDQMigration))
26         .ExecutePending;
27
28     Application.Run;
```



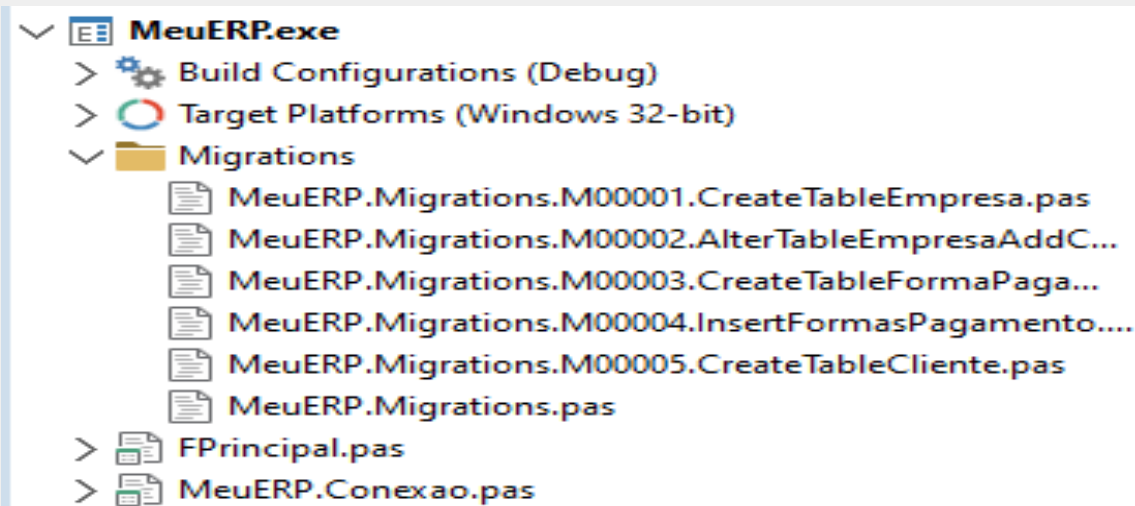
Demo!!! Vai mandar os fontes???



# Boas práticas



- **Padronizar os nomes dos arquivos:** nomes descritivos e consistentes para os arquivos de migração, seguindo um padrão específico de data e hora ou um número sequencial.



# Boas práticas



- **Evitar alterar migrations executados:** Se necessário ajustar algo que foi alterado anteriormente, o melhor caminho é criar uma nova Migration.

```
procedure TMeuERPMigrations00001CreateTableEmpresa.Up;
var
    LQuery: TFDQuery;
begin
    LQuery := MeuERPConexao.FDQMigration;
    LQuery.Close;
    try
        LQuery.SQL.Clear;
        LQuery.SQL.Add('create table if not exists empresa (');
        LQuery.SQL.Add('    id integer primary key autoincrement,');
        LQuery.SQL.Add('    nome varchar(100))');
        LQuery.ExecSQL;
    finally
        LQuery.Close;
    end;
end;
```

```
procedure TMeuERPMigrations00002AlterTableEmpresaAddCnpj.Up;
var
    LQuery: TFDQuery;
begin
    LQuery := MeuERPConexao.FDQMigration;
    LQuery.Close;
    try
        LQuery.SQL.Clear;
        LQuery.SQL.Add('alter table empresa add column cnpj varchar(20) null');
        LQuery.ExecSQL;
    finally
        LQuery.Close;
    end;
end;
```



# Boas práticas



- **Manter migration simples e focada:** crie migrações que se concentrem em uma única alteração por vez. Agrupe apenas quando as alterações envolverem uma mesma tabela, coluna ou entidade. Assim, ficará mais fácil compreender as atualizações e manter o histórico.
- **Testar migration antes da implantação:** Realizar testes minuciosos das migrações de dados em ambientes de desenvolvimento antes de aplicá-las em produção é importante para validar os ajustes e evitar erros e outros efeitos indesejados.



Dúvidas???



# Embarcadero Conference 2024

Inovação faz parte do nosso DNA!



Quer me ver na  
**#ECON25?**  
Acesse o QRCode  
e avalie minha palestra!




**Gabriel Baltazar**

 [@gabrielbaltazar.f1](https://www.instagram.com/gabrielbaltazar.f1)

 [Linkedin.com/in/gabrielbaltazar-dev](https://www.linkedin.com/in/gabrielbaltazar-dev)

 [Gabrielbaltazar.dev@gmail.com](mailto:Gabrielbaltazar.dev@gmail.com)

 [\(21\) 99849-5213](tel:(21)99849-5213)

