O nome das tabelas são salvas em constantes no arquivo '_app/Config.inc.php', você pode identificá-las pelo prefixo "`DB_`".

Nos exemplos abaixo irei usar a constante 'DB_POSTS' que faz referência a uma das tabelas do módulo de blog.

 # OPERAÇÃO SELECT (Referência _app/Conn/Read.class.php)

Iniciando a classe:

```
$Read = new Read;
```

Método ExeRead: utilizado para operações de leitura simples.
parâmetro 1: Constante da tabela do banco de dados
parâmetro 2: String SQL
parâmetro 3: Valores das chaves informadas no parâmetro 2.

IMPORTANTE: Jamais coloque valores que venham direto do usuário diretamente no **parâmetro 2,** você deve criar uma identificação para o mesmo e passar estes valores pelo **parâmetro 3** como no exemplo abaixo.

```
$Read->ExeRead(DB_POSTS, "WHERE post_id = :id AND post_status = :status",
"id=1&status=1");

//Retorna os dados operação acima
$Read->getResult();
```

Método FullRead: utilizado para operações mais complexas.
parâmetro 1: String SQL
parâmetro 2: Valores das chaves informadas no parâmetro 2.

IMPORTANTE: Jamais coloque valores que venham direto do usuário diretamente no **parâmetro 1,** você deve criar uma identificação para o mesmo e passar estes valores pelo **parâmetro 2** como no exemplo abaixo.

```
$Read->FullRead("SELECT * FROM ". DB_POSTS ." WHERE post_id = :id AND
post_status = :status ","id=1&status=1");

//Retorna os dados operação acima
$Read->getResult();
```

 # OPERAÇÃO UPDATE (Referência _app/Conn/Update.class.php)

Iniciando a classe:

```
$Update = new Update;
```

Método ExeUpdate:
parâmetro 1: Constante da tabela do banco de dados
parâmetro 2: Array de dados para atualizar
parâmetro 3: String SQL
parâmetro 4: Valores das chaves informadas no parâmetro 3.

IMPORTANTE: Jamais coloque valores que venham direto do usuário diretamente no **parâmetro 3,** você deve criar uma identificação para o mesmo e passar estes valores pelo **parâmetro 4** como no exemplo abaixo.

```
$Update->ExeUpdate(DB_POSTS,[
      "post_title" => "Novo título",
      "post_desc" => "Nova descrição"
], "WHERE post_id = :id","id=1");

//Retorna TRUE em caso de sucesso e FALSE em caso de erros
$Update->getResult();

//Retorna o número de linhas alteradas no banco!
$Update->getRowCount();
```

 # OPERAÇÃO DELETE (Referência _app/Conn/Delete.class.php)

Iniciando a classe:

```
$Delete = new Delete;
```

Método ExeDelete:
parâmetro 1: Constante da tabela do banco de dados
parâmetro 2: String SQL
parâmetro 3: Valores das chaves informadas no parâmetro 2.

IMPORTANTE: Jamais coloque valores que venham direto do usuário diretamente no **parâmetro 2,** você deve criar uma identificação para o mesmo e passar estes valores pelo **parâmetro 3** como no exemplo abaixo.

```
$Delete->ExeDelete(DB_POSTS, "WHERE post_id = :id","id=1");

//Retorna TRUE em caso de sucesso e FALSE em caso de erros
$Delete->getResult();

//Retorna o número de linhas afetadas no banco!
$Delete->getRowCount();
```

OPERAÇÃO CREATE (Referência _app/Conn/Create.class.php)

Iniciando a classe:

```
$Create = new Create;
```

Método ExeCreate:
parâmetro 1: Constante da tabela do banco de dados
parâmetro 2: Array de dados

```
$Create->ExeCreate(DB_POSTS,[
      "post_title" => "Novo post",
      "post_desc" => "descrição  do post"
]);

//Retorna o ID do registro em caso de sucesso e FALSE em caso de erros
$Update->getResult();
```

Método ExeCreateMulti:
parâmetro 1: Constante da tabela do banco de dados
parâmetro 2: Matriz de dados

```
$Create->ExeCreateMulti(DB_POSTS,[
      [
        "post_title" => "Novo post 01",
        "post_desc" => "descrição  do post"
      ],
      [
        "post_title" => "Novo post 02",
        "post_desc" => "descrição  do post"
      ]
]);

//Retorna o ID do registro em caso de sucesso e FALSE em caso de erros
$Update->getResult();
```