SQL Server (Query) - Dicas e Funções (Versões Mais Antigas do Agilityflow)

> **IMPORTANTE**: Essa documentação é referente as versões antes de 2024 que são mais antigas do agilityflow e que utilizam o Banco de Dados **SQL SERVER.** O agilityflow nas últimas versões está utilizando **POSTGRESQL** .

No agilityflow você pode buscar os dados utilizando SQL, através de queries. As queries devem ser compatíveis com **SQL Server.** Além disso existem alguns padrões que você deve seguir para obter um melhor resultado no agilityflow. Abaixo estão alguns detalhes importantes.

Tratamento de Registros Deletados

> **IMPORTANTE:**
> É de extrema importância o tratamento de registros deletados, usando as regras abaixo.
>
> Já está no nosso Roadmap o tratamento automático dos deletados, enquanto isso, o tratamento se torna obrigatório pelo desenvolvedor.

Para manter a integridade dos dados, o agilityflow nunca apaga um registro de uma tabela. Apenas o marca como deletado. Portanto, para qualquer query, é preciso excluir os deletados da lista.

Essa marcação é feita com campo deletado. Se o valor for 0, significa que ele não foi deletado e está visível no sistema. Caso esteja 1, significa que o registro já foi excluído e não é possível mais vê-lo na listagem principal do sistema.

Exemplo 1, sem o tratamento de deletados:

```
select usu_nome, usu_email from tbl_usuario
```

Resultado:



Exemplo 2, com o tratamento de deletados:

```
select usu_nome, usu_email from tbl_usuario
where deletado = 0
```

Resultado:



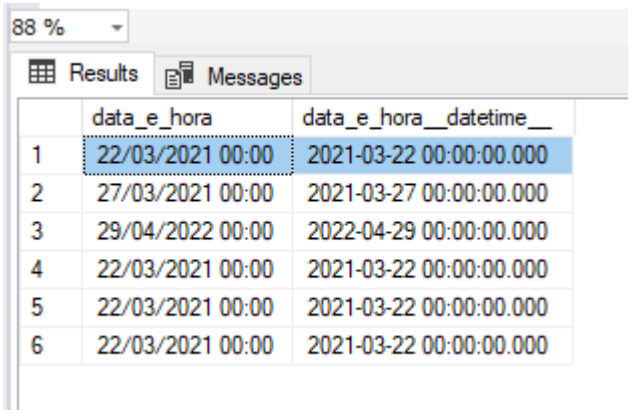
Campo do tipo Datetime (Data e Hora)

Sempre que um campo de texto com máscara do tipo data é criado, o sistema cria uma cópia com o mesmo nome seguido de "**__datetime__**" essa informação é gravada em uma coluna no banco de dados do tipo **datetime**, ao invés de varchar. Esses campos são criados para facilitar o uso tipado do dado nas queries

Exemplo de campo Datetime

Por exemplo, se o campo se chama "data_e_hora", haverá um outro chamado "data_e_hora__datetime__" e o conteúdo estará no formato padrão do sql server, **YYYY-MM-DD hh:mm:ss**

O campo "data_e_hora" estará sempre como varchar e o "data_e_hora__datetime__" estará como datetime



	data_e_hora	data_e_hora__datetime__
1	22/03/2021 00:00	2021-03-22 00:00:00.000
2	27/03/2021 00:00	2021-03-27 00:00:00.000
3	29/04/2022 00:00	2022-04-29 00:00:00.000
4	22/03/2021 00:00	2021-03-22 00:00:00.000
5	22/03/2021 00:00	2021-03-22 00:00:00.000
6	22/03/2021 00:00	2021-03-22 00:00:00.000

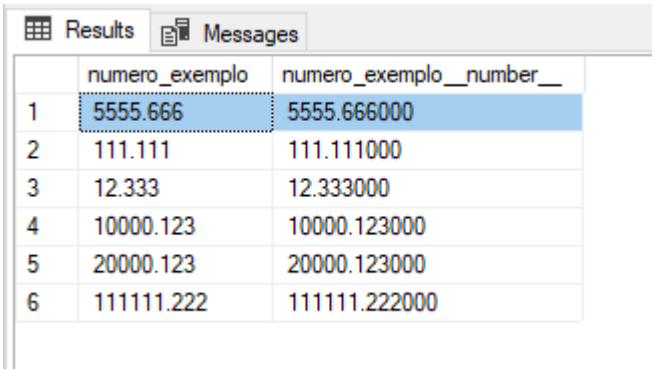
Campo do tipo Numérico (decimal, float, double, int, number)

Sempre que um campo de texto com máscara do tipo número é criado, o sistema cria uma cópia com o mesmo nome seguido de "**__number__**" essa informação é gravada em uma coluna no banco de dados do tipo **numeric(18,6),** ao invés de varchar. Esses campos são criados para facilitar o uso tipado do dado nas queries

Exemplo de campo numérico

Por exemplo, se o campo se chama "numero_exemplo", haverá um outro chamado "numero_exemplo__number__" e o conteúdo estará no formato padrão do sql server.

O campo "numero_exemplo" estará sempre como varchar e o "numero_exemplo__number__" estará como numeric(18,6)



	numero_exemplo	numero_exemplo__number__
1	5555.666	5555.666000
2	111.111	111.111000
3	12.333	12.333000
4	10000.123	10000.123000
5	20000.123	20000.123000
6	111111.222	111111.222000

Formatação de números

Nome da Função

`format(number,format,idioma)`

Parâmetro: **number**

Pode ser um número inteiro ou um decimal

Parâmetro: **format**

Aqui você passa a formatação que deve ser retornada:

'n0' = retorna o número sem nenhuma casa decimal
'n1' = retorna o número com 1 casa decimal
'n2' = retorna o número com 2 casas decimais
'n3' = retorna o número com 3 casas decimais
'n4' = retorna o número com 4 casas decimais
'n5' = retorna o número com 5 casas decimais
'n6' = retorna o número com 6 casas decimais

Parâmetro: **idioma** - **@sysCurrentLanguage**

Aqui você precisa passar o idioma do usuário logado, pois alguns idiomas invertem o . (ponto) e a , (virgula) do número.
O agilityflow guarda o idioma do usuário dentro da variável @sysCurrentLanguage então apenas passe no parâmetro esse variável.

Exemplo de utilização

Abaixo o número será formato para 5 casas decimais, no padrão do idioma do usuário logado.

```
select FORMAT(5800000.888, 'n5', @sysCurrentLanguage)
```

Retorno para o usuário que está logado no agilityflow em português ou espanhol

```
5.800.000,88800
```

Retorno para o usuário que está logado no agilityflow usando em inglês

```
5,800,000.88800
```

Funções de data no SQL Server

As principais funções para manipular datas são: **GETDATE**, **DATEPART**, **DATEADD** e **DATEDIFF**.

Um detalhe importante é que as funções de data trabalham referenciando unidades de data. As mais comuns são:

- year(ano);
- month(mês);
- day(dia);

GETDATE ( )

A função **GETDATE** retorna a data e a hora atuais do sistema.

```
SELECT GETDATE()
```

DATEPART (unidade, data)

A função **DATEPART** retorna a parte especificada de uma data como um inteiro. Observe os exemplos:

```
SELECT DATEPART ( YEAR , '2024-02-01' )
```

Reposta: 2024

```
SELECT DATEPART ( MONTH, '2024-02-01' )
```

Reposta: 2

```
SELECT DATEPART ( DAY, '2024-02-01' )
```

Reposta: 1

DATEADD (unidade, numero_unid, data)

A função **DATEADD** retorna uma nova data através da soma do número de unidades especificadas pelo valor *unidade* a uma data. Observe os exemplos:

```
SELECT DATEADD ( DAY ,6, '2024-02-01' )
```

Reposta: 2024-02-07

```
SELECT DATEADD ( MONTH ,6, '2024-02-01' )
```

Reposta: 2024-08-01

```
SELECT DATEADD ( YEAR ,6, '2024-02-01' )
```

Reposta: 2030-02-01

DATEDIFF (unidade, data1, data2)

A função **DATEDIFF** calcula a diferença entre as datas *data2 e data1*, retornando o resultado como um inteiro, cuja unidade é definida pelo valor *unidade*. Observe os exemplos:

```
SELECT DATEDIFF ( DAY , '2024-02-01' , '2024-05-25' )
```

Reposta: 114 (dias)

```
SELECT DATEDIFF ( MONTH , '2024-02-01' , '2024-05-25' )
```

Reposta: 3 (meses)

```
SELECT DATEDIFF ( YEAR , '2024-02-01' , '2026-05-25' ) )
```

Reposta: 2 (anos)

Sabendo disso, para por exemplo saber o número de dias vivido por você até hoje é:

```
SELECT DATEDIFF(DAY, suadata, GETDATE())
```

suadata deve ser substituída pela sua data de nascimento.

Outro exemplo interessante é mostrado através do código SQL abaixo. Usando funções de data, exibimos, para cada cliente, a idade em dias, meses e em anos (idade do cliente na data atual e

em 31 de dezembro).

Observe a lógica utilizada no comando **CASE**. Neste caso, é testado se o cliente já fez aniversário, comparando o mês em que ele nasceu com o mês corrente e comparando o dia em que ele nasceu com o dia corrente. Se essa comparação for verdadeira, basta diminuir o ano atual do ano de nascimento do cliente. Caso contrário (o cliente ainda não fez aniversário), temos que diminuir 1 do valor anterior.

```
SELECT NOME, NASCIMENTO,
  DATEDIFF(DAY,NASCIMENTO,GETDATE())AS DIASVIVIDOS,
  DATEDIFF(MONTH,NASCIMENTO,GETDATE()) AS MESESVIVIDOS,
  CASE WHEN
    DATEPART(MONTH,NASCIMENTO)<= DATEPART(MONTH,GETDATE()) AND
    DATEPART(DAY,NASCIMENTO)<= DATEPART(DAY,GETDATE())
  THEN
      (DATEDIFF(YEAR,NASCIMENTO,GETDATE()))
  ELSE
      (DATEDIFF(YEAR,NASCIMENTO,GETDATE()))- 1
      END AS IDADEATUAL,
  DATEDIFF(YEAR,NASCIMENTO,GETDATE())AS IDADE3112
FROM CLIENTE
```

26-05pic.JPG

Relatório

Relatório - Filtro

Sempre que um relatório possuir um filtro, é necessário incluir esse filtro na query que gera os dados do relatório.

Não importa como é a query, ela deve incluir a cláusula where, como a chamada da função Filter. Essa função, possui 3 parâmetros:

```
Filter (variável_filtro, tabela, campo)
```

- variável_filtro: é a variável que o agilityflow criou, após a configuração do filtro.

- tabela: nome da tabela (definido nos Dados Técnicos) onde se encontra o dado a ser filtrado.
- campo: é o nome do campo (conforme informado no campo Coluna Banco de Dados SQL) onde a informação que o filtro utiliza se encontra.

No exemplo abaixo, a query lista os campos *usu_nome*, *usu_email e usu_sexo* da tabela do sistema de usuário (tbl_usuario) e se inclui a função Filter, usando a variável @sexo aplicada no campo *usu_sexo*.

```
select usu_nome, usu_email, usu_sexo from tbl_usuario
where Filter(@sexo, tbl_usuario, usu_sexo)
```

Exemplo de relatório sem filtro

Baseado na query acima, mas sem filtro configurado no relatório.

```
select usu_nome, usu_email, usu_sexo from tbl_usuario
```



Exemplo de relatório com filtro

O mesmo exemplo acima, com o filtro no campo sexo.



```
select usu_nome, usu_email, usu_sexo from tbl_usuario
where Filter(@sexo, tbl_usuario, usu_sexo)
```

relatorio_filtro.gif

Caso seja criado um novo filtro, basta colocar "*and*" depois do primeiro Filter, e colocar o segundo Filter. Não importa a quantidade de filtros, desde que sejam adicionados todos os Filter e os *and*.

```
select usu_nome, usu_email, usu_sexo from tbl_usuario
where Filter(@sexo, tbl_usuario, usu_sexo) and
    Filter(@nome, tbl_usuario, usu_nome) and
    Filter(@email, tbl_usuario, usu_email)
```

Relatório - Tabela de Dados

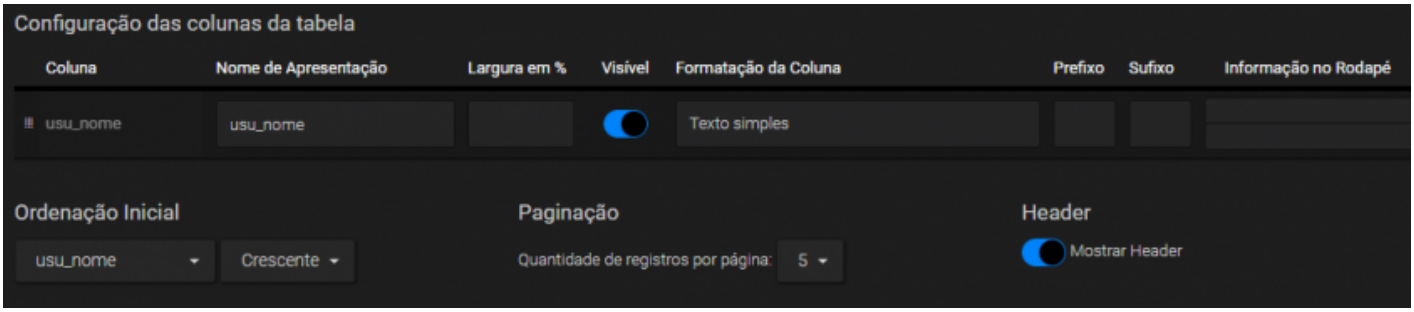
Para as tabelas de dados, existem algumas regrinhas para a customização via Query Sql

1 - Lembre-se de tratar os dados que já foram deletados, usando no where "...deletado = 0.."

2 - **Não** pode conter 'Order by' no select. Essa informação será controlada automaticamente pelo agilityflow e você poderá manipular essa informação na tela de configuração, como na imagem abaixo.

3 - **Não** pode conter o controle de 'Top' no select. Essa informação será controlada automaticamente pelo agilityflow e você poderá manipular essa informação na tela de configuração, como na imagem abaixo.

4 - **Não** pode conter regras de paginação. Essa informação será controlada automaticamente pelo agilityflow e você poderá manipular essa informação na tela de configuração, como na imagem abaixo.



Relatório - Gráfico

Para gráficos, existem algumas regrinhas para a customização via Query Sql

1 - Lembre-se de tratar os dados que já foram deletados, usando no where "...deletado = 0.."

2 - A query precisa iniciar com 'select **top** ' e o máximo do Top para essa query é **100**

3 - Diferentemente da query da tabela de dados, nessa query pode sim conter regras de 'Order by'

Relatório - Label

Para as labels do relatório, existem algumas regrinhas para a customização via Query Sql

1 - Lembre-se de tratar os dados que já foram deletados, usando no where "...deletado = 0.."

2 - A query precisa iniciar com 'select **top** ' e o máximo do Top para essa query é **1**

3 - Diferentemente da query da tabela de dados, nessa query pode sim conter regras de 'Order by'

Formulário

Lista Dinâmica - Regras

Os campos que são carregados com Lista dinâmica, podem ser customizados utilizando Query:

Algumas regras importantes:

1 - O resultado final da query precisa sempre ser os dados do formulário que você usou como base de dados

2 - Lembre-se de tratar os dados que já foram deletados, usando no where "...deletado = 0.."

3 - A query deve retornar duas colunas com os seguintes *alias*: "Name" e "Value". O "Name" deve ser a mesma coluna definida no campo de apresentação (Nome) e o "Value", deve ser a coluna "id" do formulário definido como base de dados (Cliente). Exemplo:

4 - Na query, o campo de apresentação "Name" pode ser um campo concatenado entre outras colunas.

Exemplo simples:

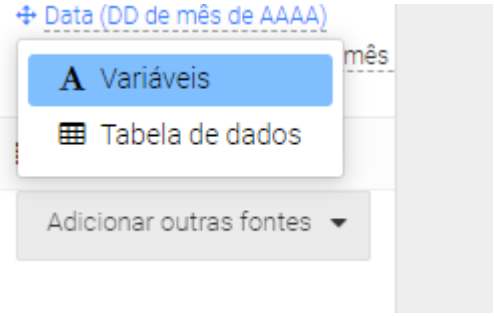
```
SELECT Id as Value, [CAMPO_DE_APRESENTACAO] as Name FROM [TABELA] where deletado = 0
```

Report Print - Relatório de Impressão

Para as labels do relatório, existem algumas regrinhas para a customização via Query Sql

Para acessar a customização, clique no botão "Adicionar Outras Fontes de Dados" e selecione o tipo:

1. **Variáveis:** para retornar uma query com apenas 1 linha e várias colunas, cada coluna se torna um campo separado para ser utilizado no relatório
2. **Tabela de Dados:** para retornar uma tabela com diversas colunas e diversas linhas e usa-las em conjunto em uma tabela



Algumas regras importantes:

1 - Lembre-se de tratar os dados que já foram deletados, usando no where "...deletado = 0.."

2 - Utilizar no "where" da query, o parâmetro **@formularioid** para filtrar pelo formulário que está sendo solicitada a impressão, caso contrário trará informações de formulários aleatórios.

3 - A query precisa iniciar com 'select **top** ' e o máximo do Top para essa query quando for uma query do tipo:

- **Tabela de Dados** é **500**,
- **Variáveis** é **1**

4 - Essa query pode sim conter regras de 'Order by'

Revision #26
Created 14 March 2019 16:39:31 by agilityflow
Updated 30 January 2025 15:08:56 by agilityflow