GrupoBoticário
EDUCA.GB
ACADEMIA DE DADOS

Neste material você encontrará o resumo dos conceitos abordados no módulo

SQL em BigQuery

Aula 1

Dê seus primeiros passos em SQL

↓

VÍDEO **01**

Por que **SQL?**

SQL é uma linguagem extremamente poderosa e está diretamente ligada a **DADOS.**

Para usar SQL existem vários sistemas, como:

- **SQL Server;**
- **Access;**
- **BigQuery.**

Usaremos o BigQuery que faz parte do Google Cloud Platform.

VÍDEO **02**

Acessando o **BigQuery**

Para iniciarmos, acesse o **GCP** pelo endereço:

https://console.cloud.google.com/

A plataforma possui diversos serviços, entre eles o **BigQuery.**
Dentro do BigQuery encontramos a seguinte estrutura:

PROJETO > CONJUNTO DE DADOS > TABELA

Ao clicar em uma tabela, encontramos as seguintes abas:

- **Esquema:** Vemos informações da estrutura da tabela.

- **Detalhes:** Informações adicionais como tamanho da tabela, data de criação entre outras.

- **Visualizar:** Dados armazenados na tabela.

VÍDEO **03**

Estrutura **SQL**

O SQL possui comandos específicos para manipulação de dados. São eles:

Insert - Criação de registros;

Select - Leitura de registros;

Update - Atualização de registros;

Delete - Exclusão de registros.

VÍDEO 04

Select

Use a aba **"Editor"** para digitar seus comandos SQL dentro do BigQuery.

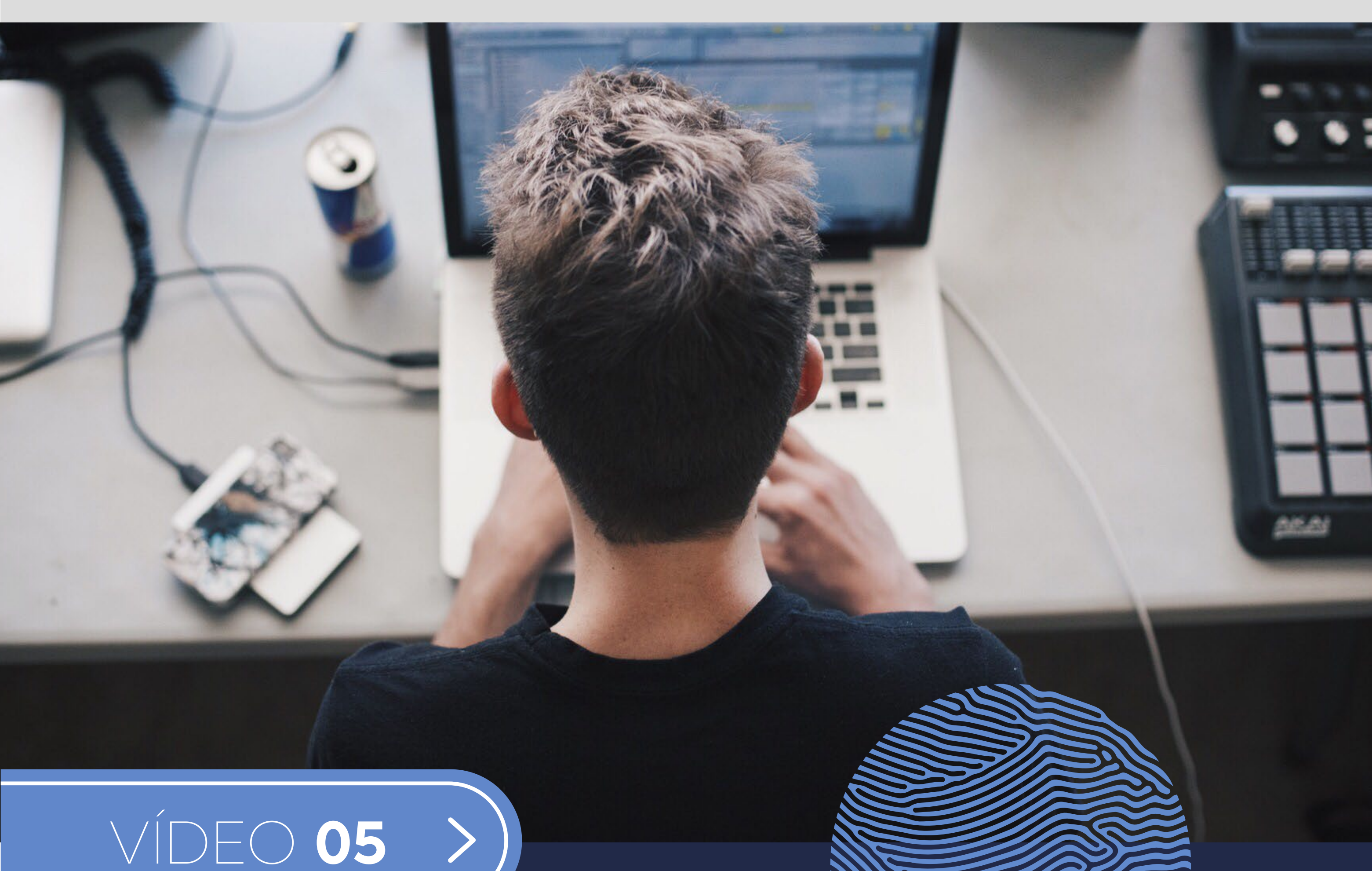
A estrutura do select é a seguinte:

select [coluna1] from `[nome do projeto].[nome do conjunto de dados].[nome da tabela]`

Veja no exemplo:

```
select c_name from `data-treinamento.base_treinamento.clientes`
```

Aperte o executar e use as opções de navegação para ir até as próximas páginas.



VÍDEO 05

Adicionando colunas na consulta

Podemos trazer vários campos da tabela separados por vírgula dentro da tabela.

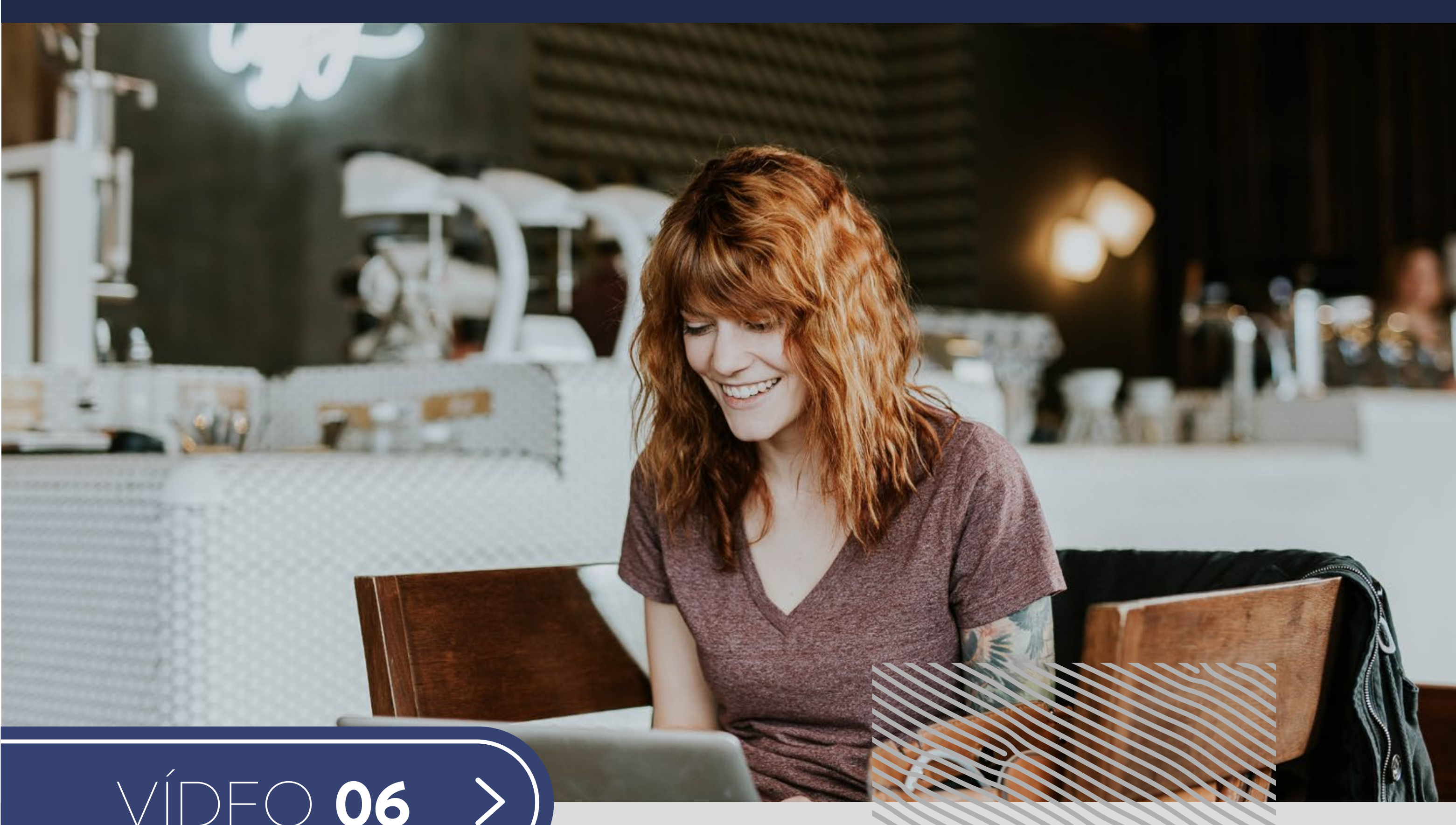
A estrutura do select é a seguinte:

select [coluna1],[coluna2],[coluna3] from `[nome do projeto].[nome do conjunto de dados].[nome da tabela]`

Veja no exemplo:

```
select c_name, c_phone from `data-treinamento.base_treinamento.clientes`
```

No BigQuery, **NÃO** utilize 'select *' pois ele possui um grande gasto operacional, encarecendo a consulta. Ao invés disso, especifique o nome das colunas.



VÍDEO 06

Tipos de Dados

Os principais tipos de dados do BigQuery são:

- **STRING:** para representar textos.
  Ex.: "José da Silva"

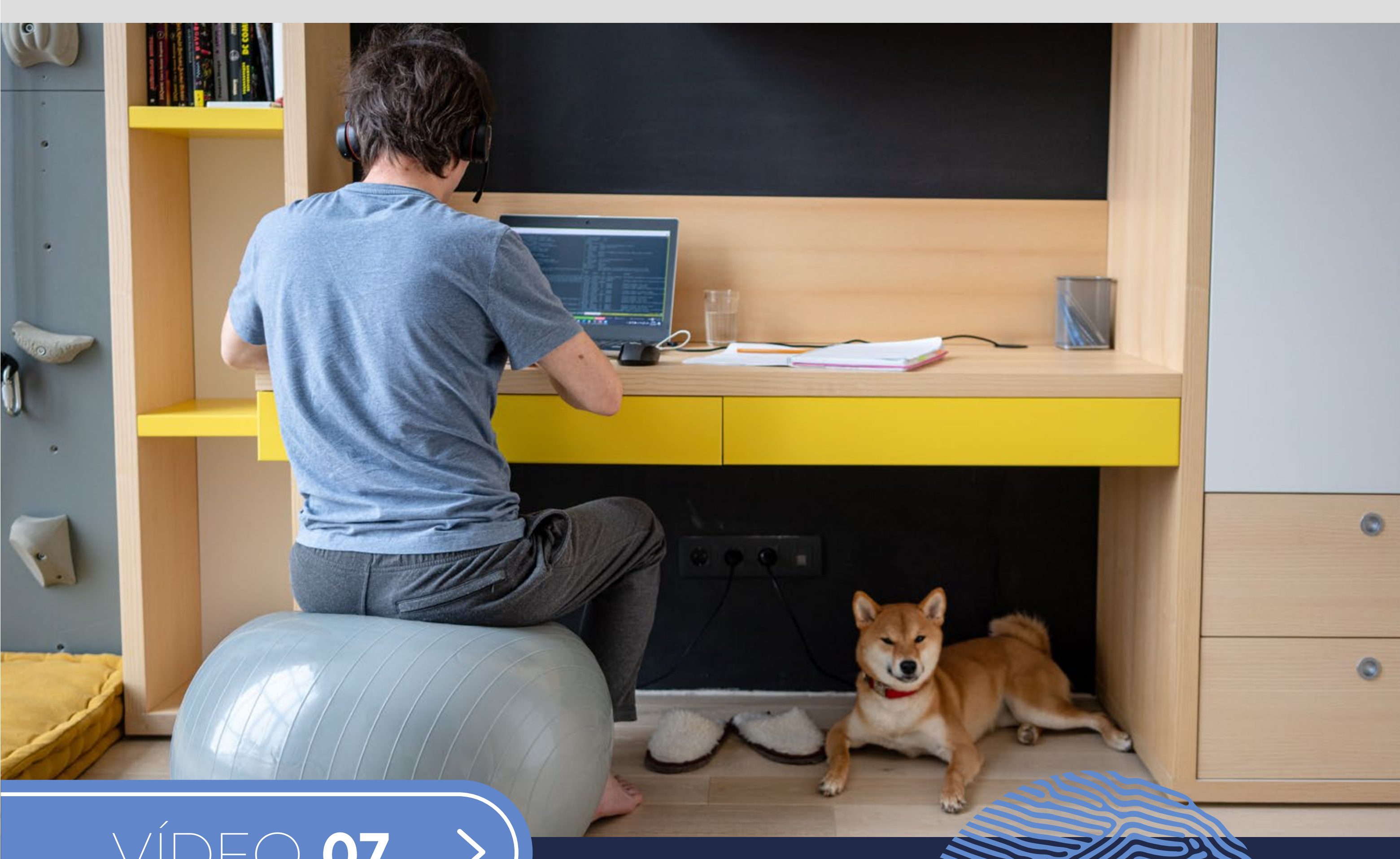
- **INTEGER:** para representar números inteiros.
  Ex.: 32

- **FLOAT:** para representar números com casas decimais.
  Ex.: 43,1

- **DATE:** para representar datas.
  Ex.: 12/04/2021

- **BOOLEAN:** para representar "verdadeiro" ou "falso".
  Ex.: TRUE

E para representar um registro que não possui dados, independentemente do seu tipo, o SQL usa a palavra-chave **NULL.**



VÍDEO 07

Operadores Aritméticos

No caso de tipos numéricos como **Integer** ou **Float**, podemos usar operadores aritméticos.

Um comando possível seria:

select c_name, c_acctbal + 250 from `data-treinamento.base_treinamento.clientes`

Os principais operadores aritméticos são:

+ : Soma

- : Subtração

*** : Multiplicação**

/ : Divisão

Nas operações SQL, podemos também fazer cálculos entre colunas, como no exemplo:

```
SELECT
 c_name,
 c_acctbal + bonus
FROM
 `data-treinamento.base_treinamento.clientes`
```

VÍDEO 08

Funções Matemáticas

Para realizar cálculos mais complexos, o BigQuery fornece funções matemáticas como o **Round**, utilizado para o arredondamento de valor de uma coluna.

Um exemplo de uso, seria:

```
select ROUND(c_acctbal) from
`data-treinamento.base_treinamento.clientes`
```

Na documentação do BigQuery podemos encontrar todas as funções matemáticas disponibilizadas.

VÍDEO 09

Cláusula **Where**

Para filtrar dados no nosso banco de dados, usamos o comando **Where**. Veja no exemplo abaixo, onde solicitamos para que sejam trazidos apenas os registros com o valor de *c_acctbal* menor que zero:

```
select c_name, c_acctbal from
 `data-treinamento.base_treinamento.clientes` where c_acctbal < 0
```

Dentro do Where podemos usar operadores lógicos como:

< Menor

> Maior

>= Maior igual

<= Menor igual

= Igual

<> Diferente

Podemos também usar funções lógicas que retornam valores booleanos. Veja no exemplo:

```
select c_name, c_acctbal from
 `data-treinamento.base_treinamento.clientes`
where ENDS_WITH(c_name,'a')
```

VÍDEO 10

Operadores ***And, Or* e *Not.***

Para filtrar usando estruturas lógicas mais complexas, podemos usar o **AND, OR** ou **NOT.** Quando queremos trazer registros que atendam ambas as condições, usamos o **AND.**

```
select c_name from  `data-treinamento.base_treinamento.clientes`
where c_acctbal < 0 and ENDS_WITH(c_name,'a')
```

Quando queremos trazer registros que atendam ao menos uma das condições, usamos o **OR.**

```
select c_name from  `data-treinamento.base_treinamento.clientes`
where ENDS_WITH(c_name,'a')  or ENDS_WITH(c_name,'b')
```

Podemos combinar várias condições ao mesmo tempo, utilizando **parênteses** para representar a prioridade de verificação conforme expressões matemáticas. Veja o exemplo:

```
select c_name from  `data-treinamento.base_treinamento.clientes`
where (( c_acctbal > 2000)and(STARTS_WITH(c_phone,'891')))
or  (( c_acctbal > 4000)and(STARTS_WITH(c_phone,'492')))
```

Caso queira negar uma expressão, você pode usar a expressão **NOT**. Nesse exemplo, o select retorna registros que não possuem valor maior que dois mil:

```
select c_name,c_acctbal from
`data-treinamento.base_treinamento.clientes`
where NOT( c_acctbal > 2000)
```

VÍDEO **11**

Conceito *Join*

Quando queremos cruzar dados entre tabelas diferentes usamos os comandos **JOINs**, semelhante ao **PROCV** do **Excel.**

Seu funcionamento se baseia na teoria de conjuntos, onde podemos unir dados de dois conjuntos ou mais. Para unir os dados, usamos identificadores chaves em um dos conjuntos que referencia a uma coluna específica do outro conjunto.

No exemplo abaixo temos duas tabelas:

Categoria	
ID	**Nome**
1	Hatch
2	SUV
3	Sedan
4	Tratores

Automóvel		
ID	**Nome**	**ID Categoria**
1	Gol	1
2	Jeta	3
3	T-Cross	2
4	Fox	1
5	Nivus	2
6	Amarok	NULO

Para fazer uma junção entre as tabelas, usamos o **INNER JOIN**. Ele traz somente os registros que estão relacionados.

Inner Join entre as tabelas Categoria e Automóvel:

Resultado do Inner Join				
ID	**Nome**	**ID**	**Nome**	**ID Categoria**
1	Hatch	1	Gol	1
3	Sedan	2	Jeta	3
2	SUV	3	T-Cross	2
1	Hatch	4	Fox	1
2	SUV	5	Nivus	2

Podemos também usar o **LEFT JOIN** e o **RIGHT JOIN** para unir tabelas retornando todos os dados da tabela à esquerda (*LEFT JOIN*) ou dados da tabela à direita (*RIGHT JOIN*).

No exemplo abaixo usamos o **Right Join** para retornar os campos da direita com os registros relacionados. Mas notem que mesmo que um registro da direita não tenha registro relacionado, ele retorna, preenchendo os campos com **NULO.**

Right Join entre as tabelas Categoria e Automóvel:

Resultado do Right Join				
ID	**Nome**	**ID**	**Nome**	**ID Categoria**
1	Hatch	1	Gol	1
3	Sedan	2	Jeta	3
2	SUV	3	T-Cross	2
1	Hatch	4	Fox	1
2	SUV	5	Nivus	2
NULO	NULO	NULO	Amarok	NULO

VÍDEO **12**

Joins na **Prática**

Para usar os JOINs no SQL, usamos a seguintes sintaxes:

[Tabela da esquerda]
INNER JOIN

[Tabela da direita]
ON

([Tabela da esquerda].[coluna do identificador] = [Tabela da direita].[coluna do identificador])

Veja no exemplo:

```
SELECT
`data-treinamento.base_treinamento.clientes`.c_name,

`data-treinamento.base_treinamento.pedidos`.o_orderkey

FROM  `data-treinamento.base_treinamento.clientes`

INNER JOIN
`data-treinamento.base_treinamento.pedidos`

ON
(`data-treinamento.base_treinamento.clientes`.c_custkey =
`data-treinamento.base_treinamento.pedidos`.o_custkey)
```

GrupoBoticário
EDUCA.GB