

Bd para gerenciamento de RH.

Aluno: Arthur Henrique Santana dos Santos

Curso: Administrador de Banco de dados



Minimundo

Uma empresa possui um departamento de recursos humanos responsável pela gestão dos funcionários e seus dados.

Essa empresa teria funcionários que trabalham em diferentes departamentos e ocupam diferentes cargos, cada um com seu próprio salário e benefícios.

Os funcionários podem ser avaliados periodicamente com a pontuação de desempenho de 1 a 5 estrelas em relação ao seu desempenho, sua evolução profissional é acompanhada ao longo do tempo através do histórico de emprego.

Essas informações são registradas no sistema de gerenciamento de recursos humanos para auxiliar no planejamento e tomada de decisões relacionadas à gestão de pessoas.

É fundamental a criação de um banco de dados seguro para a armazenagem e manipulação desses dados.

Os dados que devem ser armazenados dos funcionários são nome, data de nascimento, e-mail, CPF, data de admissão, departamento que está alocado, cargo, salário, benefício, seus registros de ponto, férias e telefone tendo dois obrigatórios e dois opcionais no máximo.

Um funcionário deve ser alocado em um e somente um departamento e cargo específico, e cada departamento e cargo deve ter um nome único.

O salário de um funcionário deve ser registrado de acordo com o seu cargo.

Um funcionário pode receber nenhum ou vários benefícios, que devem ser registrados no sistema com seu nome e tipo. Ex: Saúde, alimentação, transporte, bonificação.

Para cada funcionário deve ter registrado os seus dependentes (caso haja), contendo informações deles, como: nome, data de nascimento e grau de parentesco.

Cada benefício pode estar associado a um ou mais funcionários.

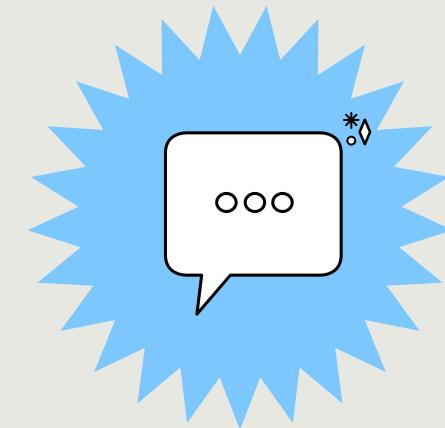
O registro de ponto do usuário deve armazenar o horário, data e tipo de registro, que pode ser de entrada, saída para almoço, retorno do almoço e saída.

O banco também deve registrar a folha de ponto dos funcionários, onde deve constar mês, ano, total de horas trabalhadas e total de faltas.

Para cada funcionário há uma folha de pagamento, onde consta o mês, ano, impostos e valor.

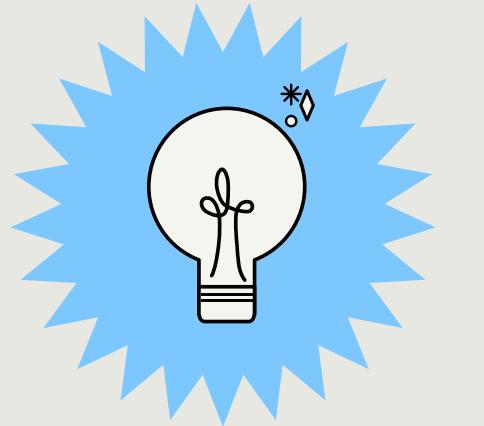


Modelos



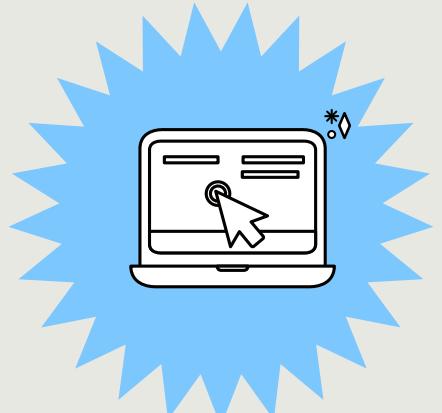
Conceitual

Um modelo conceitual é uma descrição do banco de dados que será projetado de forma independente da ferramenta computacional de SGBD. Esse modelo representa os dados de cada tabela sem se preocupar com a forma pela qual esses serão registrados no SGBD utilizado.



Lógico

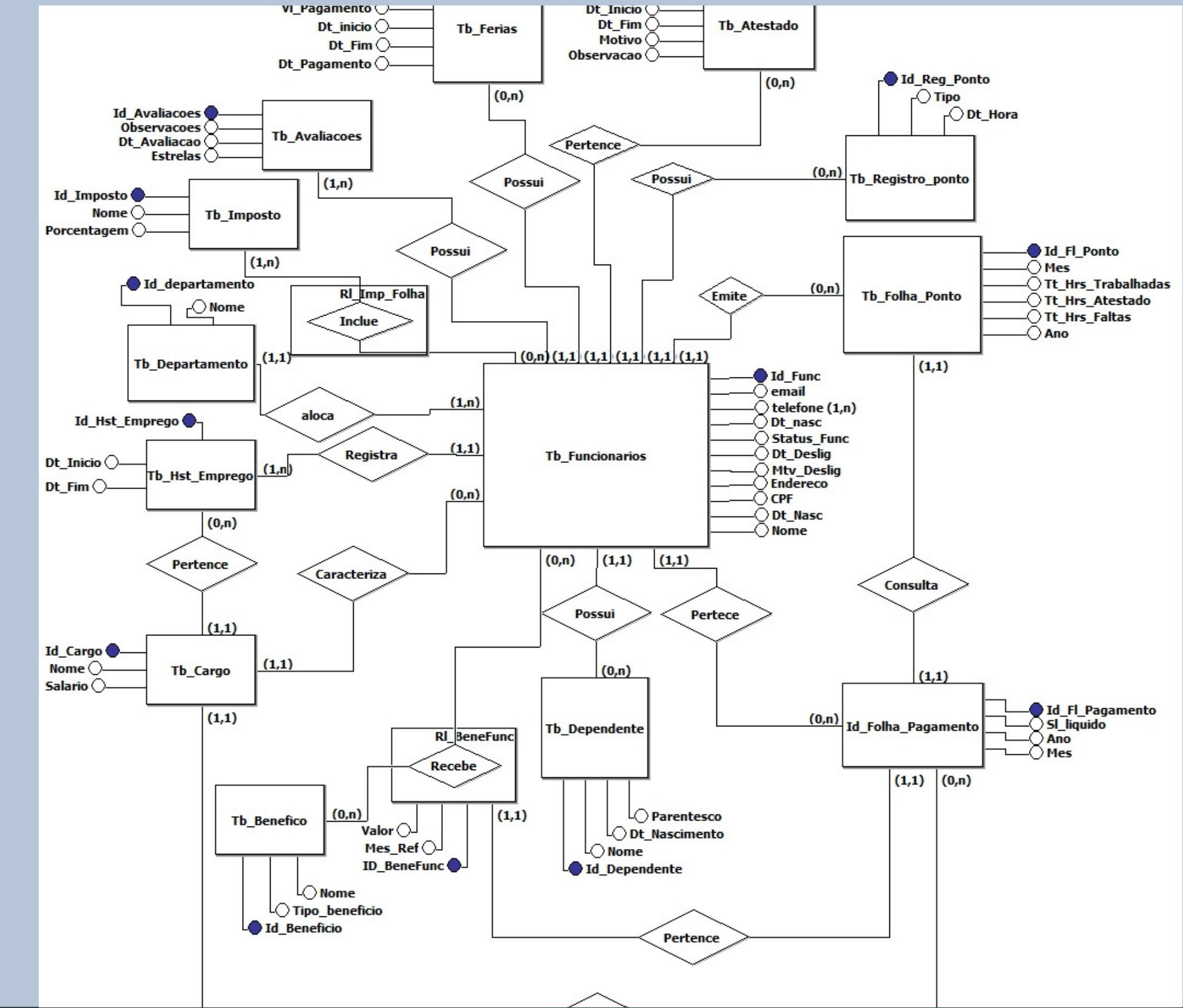
O modelo lógico é um diagrama que representa visualmente as relações lógicas entre os recursos, e resultados de um programa. O modelo lógico serve para definir como um sistema deve ser implementado, independentemente do sistema de gerenciamento de banco de dados que está sendo usado.



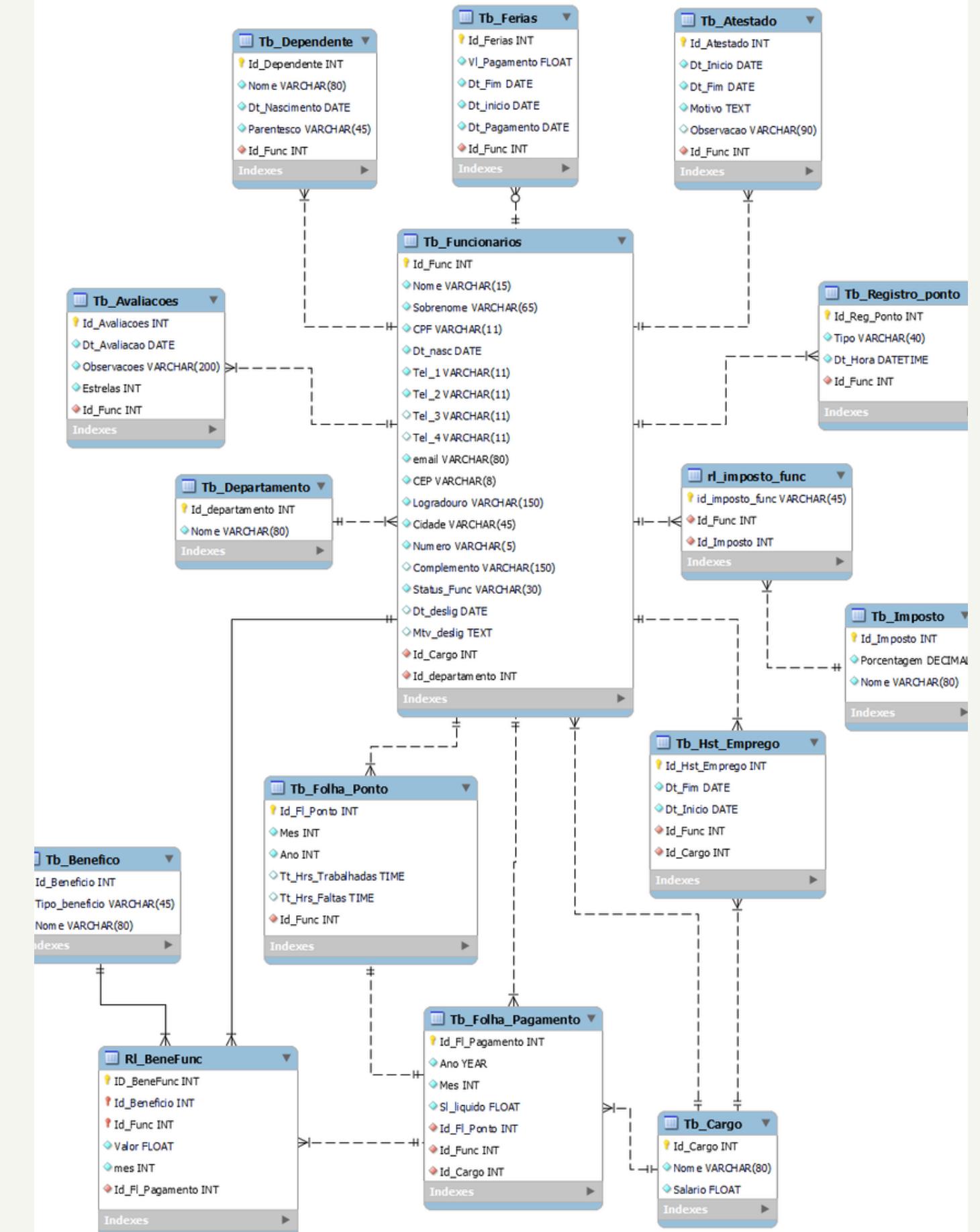
Físico

O modelo físico é a modelagem física do modelo de banco de dados. Neste caso leva-se em conta as limitações impostas pelo SGBD escolhido e deve ser criado sempre com base nos exemplos de modelagem de dados produzidos no modelo conceitual e lógico.

Modelo conceitual

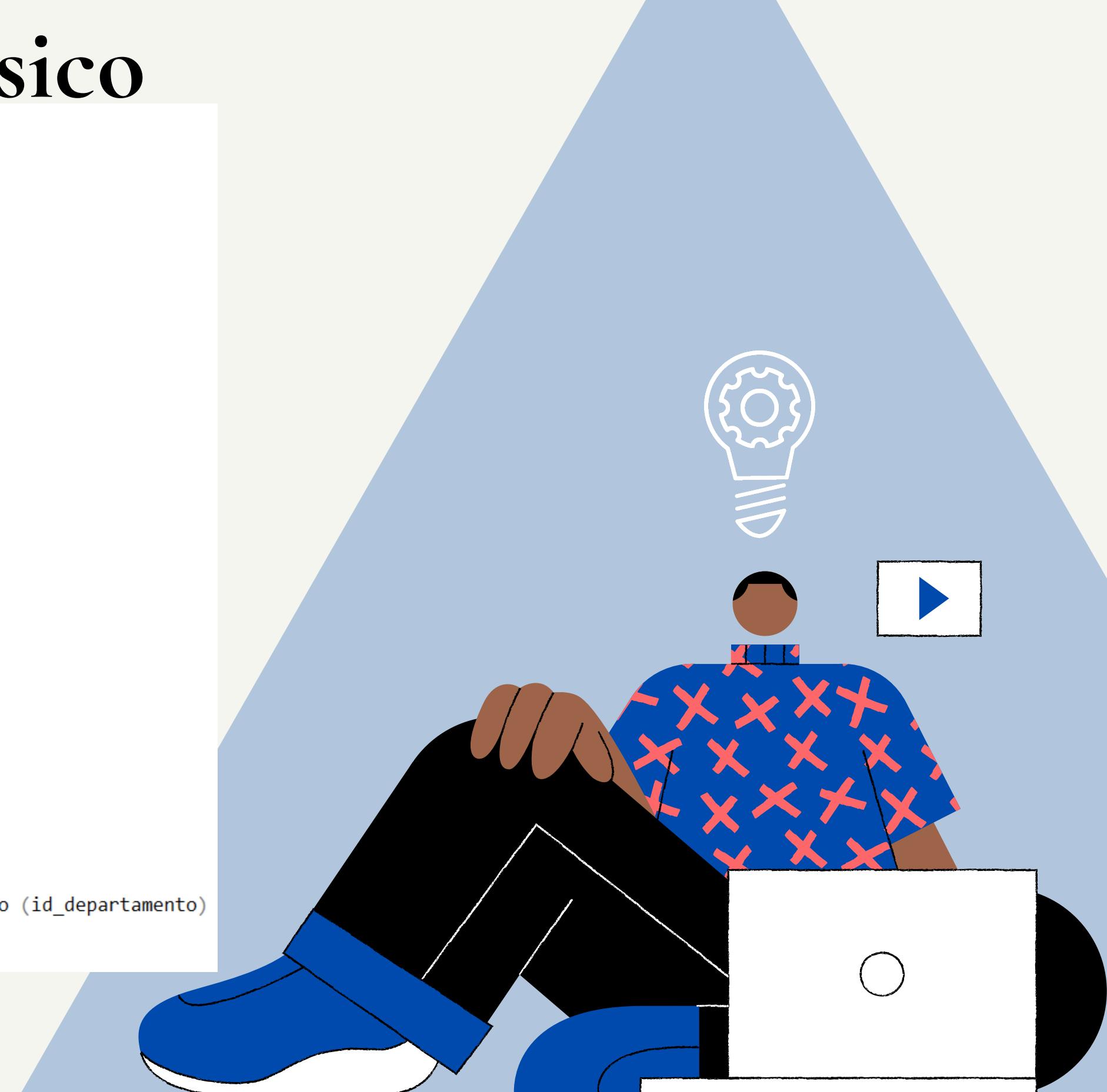


Modelo Lógico



Modelo físico

```
34 CREATE TABLE tb_departamento(  
35     id_departamento      INT IDENTITY      NOT NULL,  
36     nome                  VARCHAR(80)      NOT NULL,  
37     PRIMARY KEY (id_departamento),  
38 )  
39 GO  
40  
41 CREATE TABLE tb_funcionario(  
42     id_func      INT IDENTITY      NOT NULL,  
43     nome         VARCHAR(15)      NOT NULL,  
44     sobrenome    VARCHAR(65)      NOT NULL,  
45     CPF          VARCHAR(11) UNIQUE      NOT NULL,  
46     dt_nasc      DATE          NOT NULL,  
47     tel_1        VARCHAR(11)      NOT NULL,  
48     tel_2        VARCHAR(11)      NOT NULL,  
49     tel_3        VARCHAR(11)      NULL,  
50     tel_4        VARCHAR(11)      NULL,  
51     email        VARCHAR(80)      NOT NULL,  
52     CEP          VARCHAR(8)       NOT NULL,  
53     logradouro   VARCHAR(150)     NOT NULL,  
54     cidade       VARCHAR(45)      NOT NULL,  
55     numero       VARCHAR(5)       NOT NULL,  
56     complemento  VARCHAR(150)     NULL,  
57     status_func   VARCHAR(30)      NOT NULL,  
58     dt_deslg     DATE          NULL,  
59     mtv_deslg    TEXT          NULL,  
60     id_cargo     INT           NOT NULL,  
61     id_departamento INT          NOT NULL  
62     PRIMARY KEY (id_func),  
63     CONSTRAINT fk_cargo_funcionario FOREIGN KEY (id_cargo) REFERENCES tb_cargo (id_cargo),  
64     CONSTRAINT fk_departamento_funcionario FOREIGN KEY (id_departamento) REFERENCES tb_departamento (id_departamento)  
65 )  
66 GO
```



Views, Funções, Procedures e Triggers

Views

Views em um banco de dados são consultas armazenadas no banco de dados que criam uma ilusão de ser uma tabela e podem ser usadas em diversas operações para simplificar as queries e facilitar o acesso a determinadas informações, conformar melhor com o modelo lógico e permitir controlar melhor o acesso aos dados para determinados usuários.

Funções

Funções são recursos que permitem realizar operações sobre dados. Existem diferentes tipos de funções SQL, como aritméticas, de carácter, de data e de conversão. As funções SQL executam uma lista arbitrária de declarações SQL, retornando o resultado da última consulta da lista.

Procedures

Stored Procedure é um conjunto de comandos em SQL que podem ser executados de uma só vez, como em uma função. Ele armazena tarefas repetitivas e aceita parâmetros de entrada para que a tarefa seja efetuada de acordo com a necessidade individual

Triggers

Um trigger é um processo que funciona como gatilho para executar algumas ações automaticamente sempre que uma mudança ocorre no banco de dados. Ou seja, é principalmente uma função voltada para a automação: sempre que uma ação ocorre, o trigger executa outra ação como resposta imediata

Selects

Além de armazenar os dados, o banco também deve apresenta-los.

SELECT é uma cláusula SQL que permite selecionar dados de um banco de dados.



```
-- 1. Consulta do codigo, nome, dos funcionário com respectivo cargo, salario, apenas dos funcionários ativos em ordem alfabetica
SELECT F.id_func, F.nome, C.nome AS Cargo
FROM tb_funcionario AS F, tb_cargo AS C
WHERE F.id_cargo = C.id_cargo;

-- 2. Consulta o nome e salario dos funcionários que ganham mais de R$5.000,00 e possuem um dependentes menores de idade.
SELECT DISTINCT F.id_func, F.Nome, C.salario
FROM tb_funcionario AS f
LEFT JOIN tb_dependente d ON F.Id_Func = D.Id_Func
INNER JOIN tb_cargo AS C ON C.id_cargo = F.id_cargo
WHERE C.salario > 5000 AND DATEDIFF(YEAR, d.dt_nascimento, GETDATE()) < 18;

-- 3. Consulta do número de registros de ponto dos funcionário em um determinado ano.
SELECT F.nome AS Funcionario, YEAR(RP.dt_hora) AS Ano, COUNT(*) AS 'Numero de Registros'
FROM tb_funcionario AS F
INNER JOIN tb_registro_ponto AS RP ON F.id_func = RP.id_func
GROUP BY F.nome, YEAR(RP.dt_hora);

-- 4. Consultar o nome, CPF e salário dos funcionários cujo o nome comece com a letra C e possuem cargo com salário maior que R$5.000,00
SELECT F.nome, F.CPF, C.salario
FROM Tb_Funcionario AS F
INNER JOIN tb_cargo AS C ON F.id_cargo = C.id_cargo *
WHERE F.nome LIKE 'C%' AND C.salario > 2000 ;

-- 5. Selecionar o nome, data de inicio e fim das ferias dos funcionários que receberam pagamento maior que R$5.000,00
SELECT F.nome AS Nome, CONVERT( VARCHAR, FR.dt_inicio, 103) AS 'Data de inicio',
CONVERT ( VARCHAR, FR.dt_fim, 103) AS 'Data de fim'
FROM tb_funcionario AS F
INNER JOIN tb_ferias AS FR ON F.id_func = FR.id_func
WHERE FR.vl_pagamento > 5000;
```

“

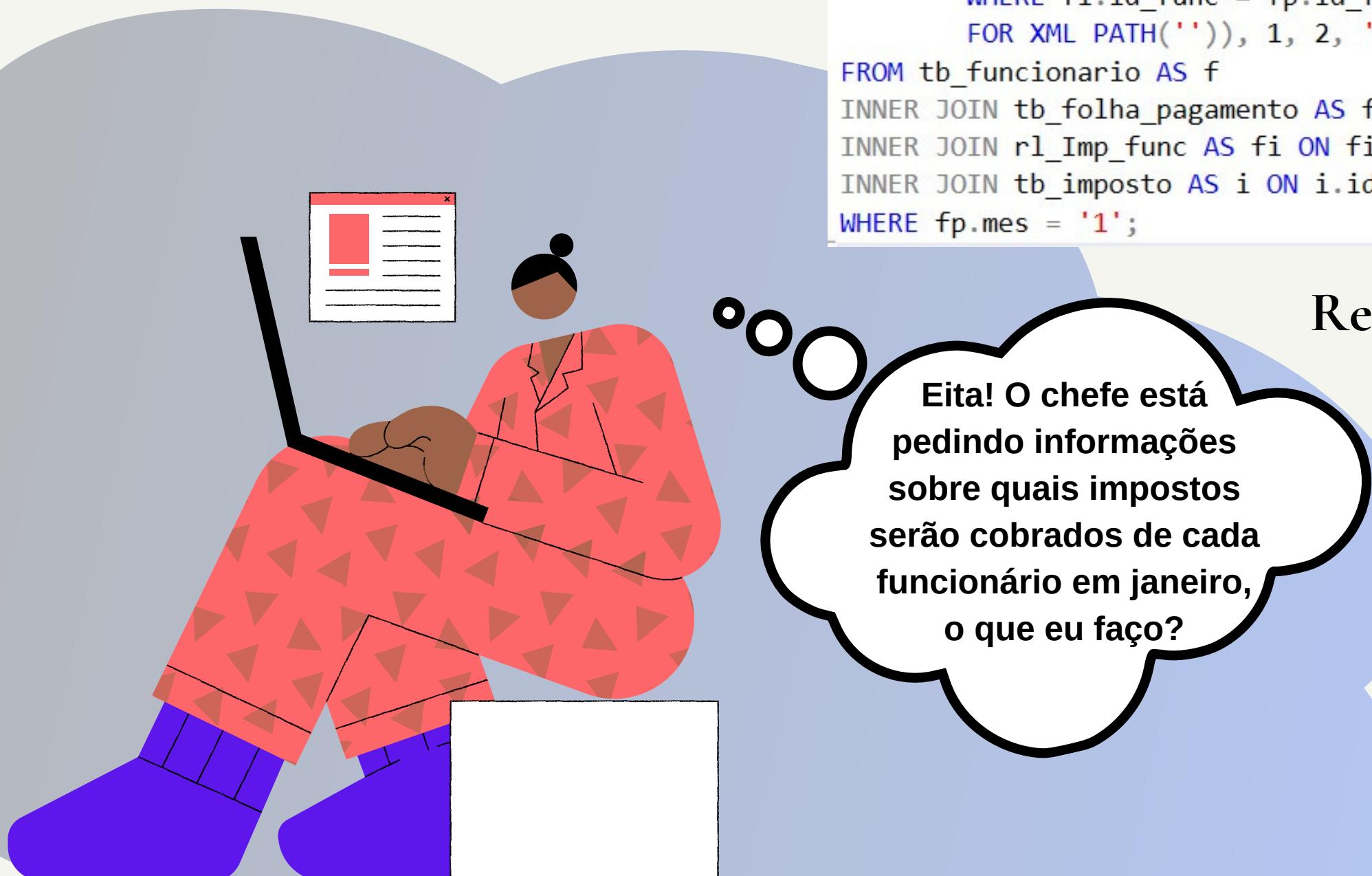
O setor de Recursos Humanos (RH) é fundamental para as empresas e a análise de dados pode ser uma ferramenta poderosa para o RH. A análise de dados pode ajudar a identificar padrões e tendências de comportamento dentro da empresa, o que possibilita o planejamento de ações que direcionam a gestão de pessoas a resultados cada vez mais positivos.



Situações e soluções dentro do setor de RH.

Solução:

```
-- 10. Selecionar o funcionario com seu respectivo imposto da folha de pagamento referente ao mês de janeiro.  
SELECT DISTINCT f.nome, STUFF((SELECT ', ' + i.nome  
    FROM tb_imposto AS i  
    INNER JOIN rl_imp_func AS fi ON i.id_imposto = fi.id_imposto  
    WHERE fi.id_func = fp.id_func  
    FOR XML PATH('')), 1, 2, '') AS Imposto  
FROM tb_funcionario AS f  
INNER JOIN tb_folha_pagamento AS fp ON f.id_func = fp.id_func  
INNER JOIN rl_Imp_func AS fi ON fi.id_func = fp.id_func  
INNER JOIN tb_imposto AS i ON i.id_imposto = fi.id_imposto  
WHERE fp.mes = '1';
```



Resultado:

1	Ana	IRRF, FGTS, INSS
2	Bruno	IRRF, FGTS, INSS
3	Camila	IRRF, FGTS, INSS
4	Carolina	IRRF, FGTS, INSS
5	Gabriela	FGTS, INSS
6	Gustavo	IRRF, FGTS, INSS
7	Isabela	IRRF, FGTS, INSS
8	José	FGTS, INSS
9	Maria	FGTS, FGTS, INSS
10	Pedro	FGTS, INSS
11	Rafael	FGTS, INSS

Criando View:

-- 1. Mostrar Nome, cargo, salario e código da folha de ponto e mês de referencia de todos os funcionários.

```
CREATE VIEW relacao_funcionarios AS
SELECT f.nome AS Funcionario, c.nome AS Cargo, c.salario AS Salario,
fl.id_fl_pagamento AS 'Cod. folha de pagamento',
fl.mes AS 'Mes de referencia'
FROM tb_funcionario f
INNER JOIN tb_cargo c ON c.id_cargo = f.id_cargo
INNER JOIN tb_folha_pagamento fl ON fl.id_func = f.id_func;
```

Ter que escrever o comando inteiro todas as vezes é muito cansativo, poderia ter uma forma mais facil...

Agora quando quiser os dados é só chamar a view:
"SELECT * FROM relacao_funcionarios"

Resultado:

	Funcionario	Cargo	Salario	Cod. folha de pagamento	Mes de referencia
1	Gustavo	Auxiliar de Serviços Gerais	2812,5	1	1
2	Gabriela	Auxiliar administrativo	1486,34	2	1
3	Camila	Medico	22327,76	3	1
4	Rodrigo	Gerente de rh	7097,42	4	1
5	Bruno	Administrador	4775	5	1
6	Isabela	Gerente Financeiro	4114	6	1
7	Rafael	Secretario trainee	1348,93	7	1
8	Carolina	Medico	22327,76	8	1
9	Pedro	Ajudante Geral	1575,6	9	1
10	Ana	Tecnico de exames	1925	10	1
11	José	Recepçãoista	1340,15	11	1
12	Maria	Jovem aprendiz	740	12	1

Função:

```
-- 1. função que calcula o valor do desconto de IRRF do funcionário.  
CREATE FUNCTION calcularIRRF (@id_func INT, @mes INT)  
RETURNS FLOAT  
AS  
BEGIN  
    DECLARE @salariobruto FLOAT  
    DECLARE @porcentagem FLOAT  
    DECLARE @valor_desconto FLOAT  
    SELECT @salariobruto = salario FROM tb_cargo WHERE id_cargo  
        = (SELECT id_cargo FROM tb_funcionario WHERE id_func = @id_func)  
    SELECT @porcentagem = i.porcentagem FROM tb_funcionario f  
        INNER JOIN rl_imp_func rl ON rl.id_func = f.id_func  
        INNER JOIN tb_imposto i ON i.id_imposto = rl.id_imposto  
        INNER JOIN tb_folha_ponto fp ON fp.id_func = f.id_func  
        WHERE f.id_func = @id_func AND fp.mes = @mes AND i.nome = 'IRRF'  
    SET @valor_desconto = @salariobruto * (@porcentagem / 100)  
    RETURN @valor_desconto  
END;
```

Chamando a função:

"**SELECT dbo.calcularIRRF (1, 1)**
AS 'Valor a ser descontado na folha referente ao IRRF' "

Resultado:

Valor a ser descontado na folha referente ao IRRF	
1	225



Marcondes teve uma ótima ideia para facilitar as consultas de seus chefes, ele criou uma procedure que cria uma visão geral dos funcionários de uma empresa.

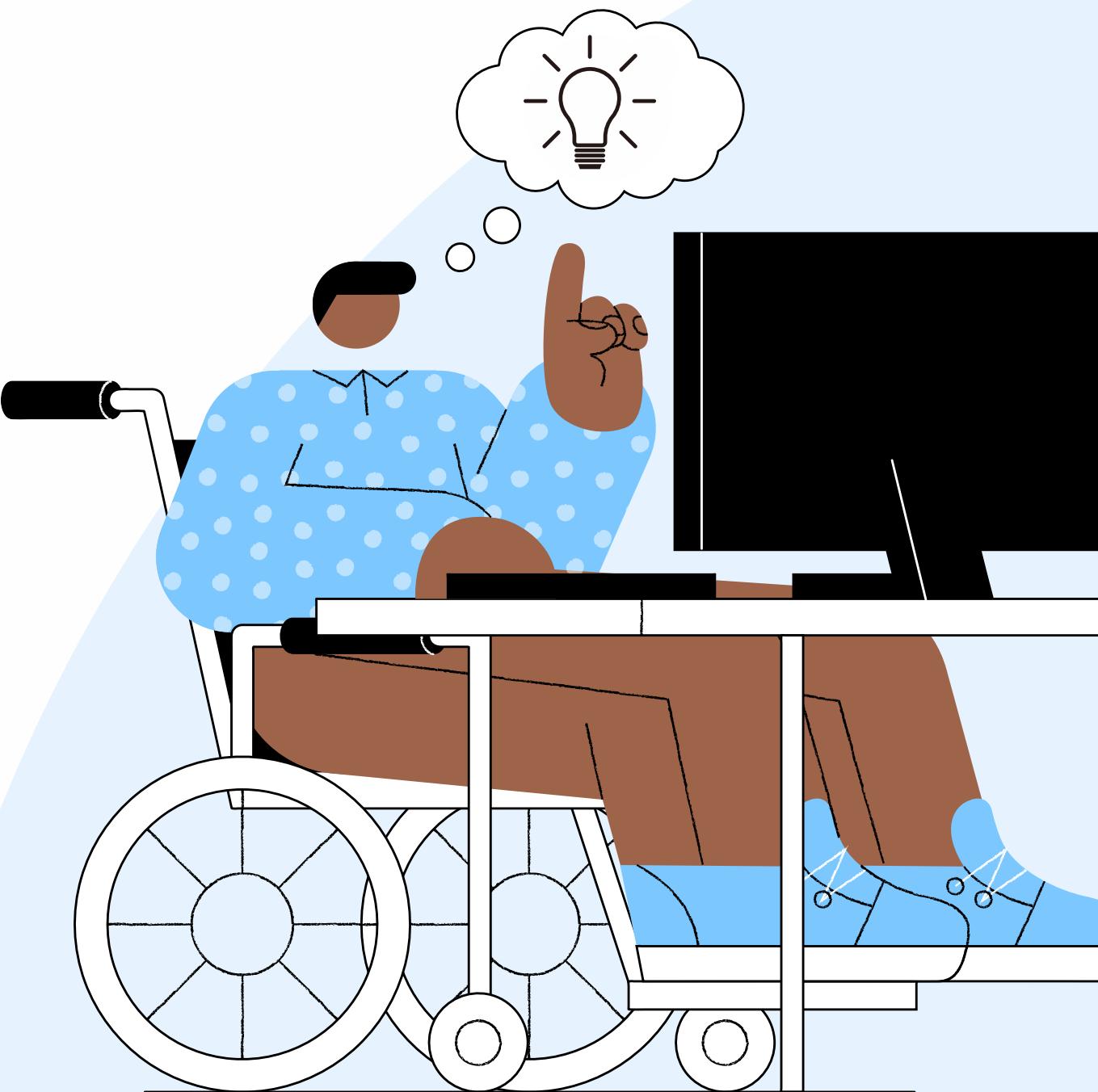
Script:

```
CREATE PROCEDURE visao_geral
AS
BEGIN
SELECT DISTINCT F.id_func ASCodigo, F.nome AS Funcionario, C.nome AS Cargo, d.nome AS Departamento,
STUFF((SELECT ', ' + B.nome
FROM tb_beneficio AS B
INNER JOIN rl_benefunc AS R ON B.id_beneficio = R.id_beneficio
WHERE R.id_func = F.id_func
FOR XML PATH('')), 1, 2, '') AS Beneficio,
C.salario
FROM tb_funcionario AS F
INNER JOIN tb_cargo AS C ON F.id_cargo = C.id_cargo
INNER JOIN rl_benefunc AS R ON R.id_func = F.id_func
INNER JOIN tb_departamento AS d On d.id_departamento = F.id_departamento
WHERE F.status_func = 'Ativo'
ORDER BY F.id_func
END;

EXECUTE visao_geral;
```

Resultado:

	Codigo	Funcionario	Cargo	Departamento	Beneficio	salario
1	1	Gustavo	Auxiliar de Serviços Gerais	Limpeza	Férias	2812,5
2	2	Gabriela	Auxiliar administrativo	Administrativo	vale-transporte, Vale alimetacao	1486,34
3	4	Camila	Medico	Atendimento	Férias	22327,76
4	5	Rodrigo	Gerente de rh	rh	vale-transporte, Vale alimetacao	7097,42
5	7	Bruno	Administrador	Administrativo	Vale refeição, Vale combustível	4775
6	8	Isabela	Gerente Financeiro	Financeiro	Vale refeição, vale-transporte	4114
7	9	Rafael	Secretario trainee	Administrativo	Vale refeição, vale-transporte	1348,93
8	10	Carolina	Medico	Atendimento	Vale alimetacao, Adicional noturno, vale-transporte	22327,76
9	11	Pedro	Ajudante Geral	Manutenção	Vale refeição, vale-transporte	1575,6
10	12	Ana	Tecnico de exames	Atendimento	Vale refeição, vale-transporte	1925
11	13	José	Recepcionista	Recepção	Vale refeição, vale-transporte	1340,15
12	14	Maria	Jovem aprendiz	Administrativo	Vale refeição, vale-transporte	740





Script:

```
-- 1. Sempre que inserir uma folha de ponto será calculada o salário liquido e inserido na tabela folha de pagamento.

CREATE TRIGGER folha_de_pagamento
ON tb_folha_ponto
AFTER INSERT AS
BEGIN
    DECLARE
        @ano INT,
        @mes INT,
        @sl_liquido FLOAT,
        @id_fl_ponto INT,
        @id_func INT,
        @id_cargo INT
    SELECT @ano = ano, @mes = mes, @id_fl_ponto = id_fl_ponto, @id_func = id_func FROM INSERTED
    SELECT @id_cargo = f.id_cargo FROM tb_funcionario AS f WHERE f.id_func = @id_func
    SELECT @sl_liquido = dbo.salario_liquido (@id_func, @mes)
    INSERT INTO tb_folha_pagamento (ano, mes, sl_liquido, id_fl_ponto, id_func, id_cargo)
    VALUES (@ano, @mes, @sl_liquido, @id_fl_ponto, @id_func, @id_cargo);
END
GO
```

Através dessa trigger, sempre que uma folha de ponto for inserida no banco de dados automaticamente é calculado o salário liquido do funcionário deduzindo os impostos.



Sobre o banco:

Tabelas: 15.

Atributos: 66.

SGBD: SQL Server Management Studio - 2019,
MySQL Workbench.

Ferramentas: Br modelos.

Tipo de banco: Relacional.



Arthur Santos

Administrador de banco de dados

Obrigado!

