UNIVERSIDADE FEDERAL DE LAVRAS

TRABALHO PRÁTICO INTRODUÇÃO AO BANCO DE DADOS

ANDREIA JOSE DA SILVA

BRUNO DE ALMEIDA DE PAULA

DANIEL ALMEIDA FRIEDRICH

ELIAS DE JESUS MIRANDA

GABRIELA SILVA MEMENTO

INTRODUÇÃO AO SISTEMA DE BANCO DE DADOS

LAVRAS-MG

2022

Descrição

Levantamento de dados para um sistema de um salão de beleza, é feito o cadastro do cliente, funcionário e fornecedor, ao agendar um procedimento é feito um pré questionário para saber se o cliente pode ou não fazer o procedimento.

Ao realizar o atendimento há uma descrição dos detalhes do que foi feito durante o atendimento, após o atendimento o cliente irá fazer o pagamento no caixa.

Os produtos ficam no salão, é feita a saída dos produtos, com data, nome do produto e quantidade utilizada, também é feita atualização ao adquirir produtos do fornecedor, é armazenado o preço cotado e a data de entrega dos produtos. Existe um histórico com a data do produto que foi comprado e o preço pago nessa compra.

O Cadastro de Pessoa é composto por: cpf (que são chaves - ou seja, são valores únicos), nome/razaoSocial, telefone, endereço(logradouro, bairro, numero, cidade, estado, país, cep), data de nascimento e email.

O cadastro dos atendimentos necessita de idAtendimento(chave), descrição, preço, tempo, resultado, nota e observação.

O cadastro dos produtos necessita do códigoProduto(chave), nome e quantidade.

O cliente além dos dados de Pessoa tem aniversário e idade como atributos derivados, cada funcionário, além dos dados de Pessoa, possui uma função, um salário e um número de registro, cada fornecedor é cadastrado com os dados de Pessoa mais o cnpj(chave) e a Razão Social.

O cliente é atendido por um funcionário e cada atendimento possui um IdAtendimento, preço, tempo gasto nos processos e descrição do atendimento. Os atendimentos geram saída dos produtos do salão e aumentam o saldo do caixa.

Os clientes devem responder à anamnese que apresenta as seguintes perguntas:

Utilizou piscina?

Fez Banho de sol?

Está gestante?

Tem alergia?

A abertura de caixa possui saldoAnterior, entrada, saída, saldoAtual, descrição da movimentação e data de movimentação.

Diagrama Entidade Relacionamento

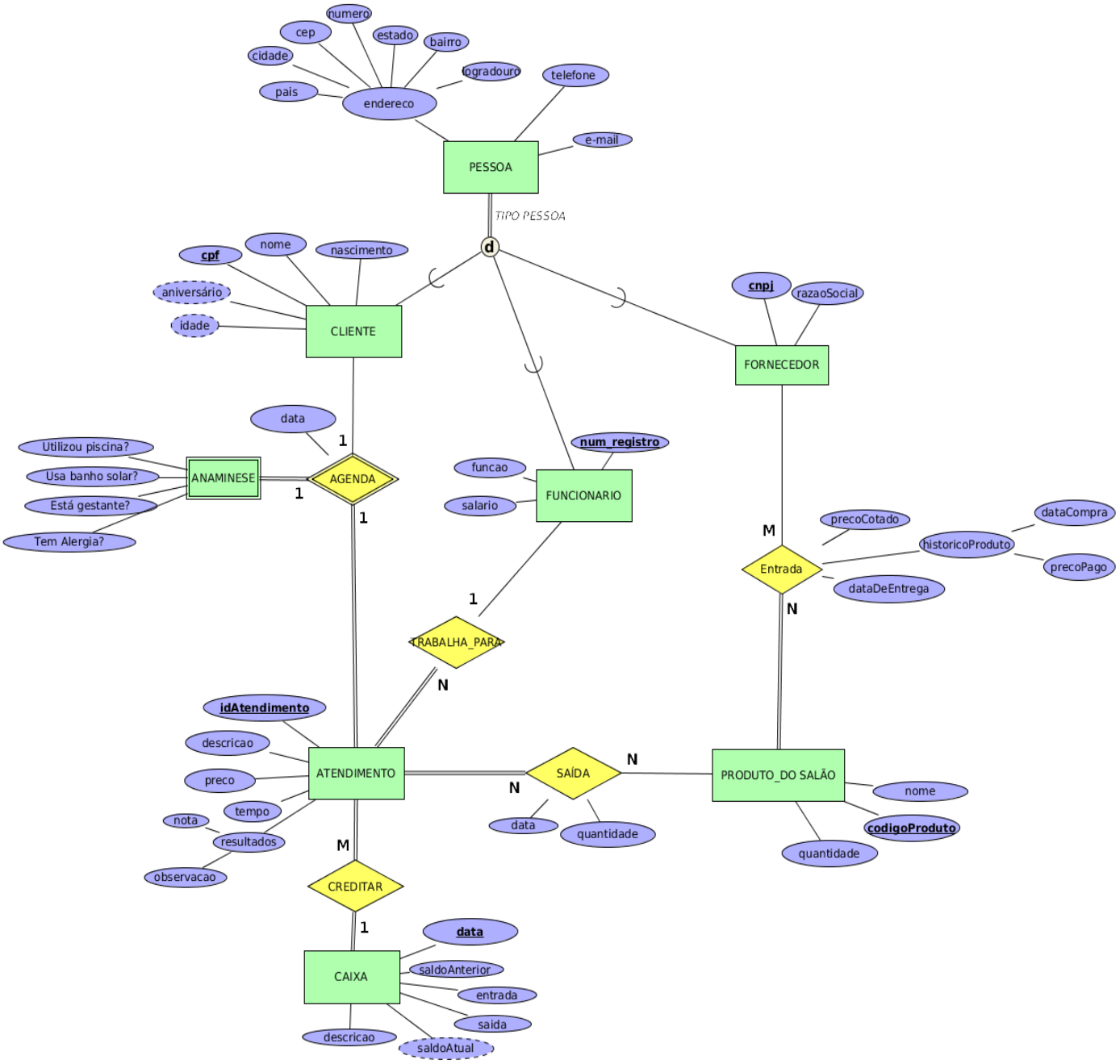


Figura 1.1: Diagrama Entidade Relacionamento de um Salão de Beleza.

Dicionário

Entidades

Tipo Entidade	Pessoa		
Descrição	Especifica o cadastro de pessoa		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
telefone	Número do telefone	Inteiro(11) Formato: (dd)ddddd-dddd	N
e-mail	Endereço eletrônico	Character(50)	N
cidade	Nome da cidade	Character(50)	N
numero	Número da residência	Inteiro(5) positivo	N
bairro	Nome do bairro	Character(30)	S
estado	Sigla do estado	Character(2)	N
logradouro	Nome do logradouro (rua, avenida, alameda etc.)	Character(80)	N
cep	Código de endereçamento postal	Character(8) Formato: dd.ddd-ddd	N
pais	Nome do país	Character(40)	N

Tipo Entidade	Cliente		
Descrição	Especificação do cadastro pessoa relativa ao cliente		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
nome	Nome completo do cliente	Character(100)	N
cpf	Cpf do cliente	Character(11)	N
aniversario	Aniversário do cliente	Data dia(2)positivo / mês(2) positivo	N
nascimento	Data de nascimento	Data	N
idade	Idade do cliente	Intero(2) positivo	N

Tipo Entidade	Funcionários		
Descrição	Especificação do cadastro pessoa relativa ao funcionário		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
funcao	Função do funcionário	Character(80)	N
salario	Salário mensal	Real(8,2) positivo	N
numRegistro	Número de registro do funcionário	Inteiro(4)positivo	N

Tipo Entidade	Fornecedor		
Descrição	Especificação do cadastro pessoa relativa ao fornecedor		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
cnpj	Cnpj da empresa	Character(14)	N
razaoSocial	Nome da empresa	Character(100)	N

Tipo Entidade	Atendimento		
Descrição	Contém dados dos atendimentos		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
idAtendimento	Código de Identificação do atendimento	Inteiro(3) positivo	N
descricao	Descreve o atendimento	Character(50)	N
preco	Preço do procedimento	Real (8,2) positivo	N
tempo	Horas gastas no atendimento	Inteiro (4) positivo	N
nota	Nota que o cliente dá para o atendimento, de 0 a 10	Inteiro(2)	S
observacao	Observação sobre o atendimento	Character(100)	S

Tipo Entidade	Anamnese		
Descrição	Conjunto de perguntas para o cliente responder antes de cada atendimento		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
Utilizou piscina?	Pergunta	Booleano	N
Usa banho solar?	Pergunta	Booleano	N
Esta gestante?	Pergunta	Booleano	N
Tem alergia?	Pergunta	Booleano	N

Tipo Entidade	Caixa		
Descrição	Contém as movimentações financeira diárias		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
data	Data em que se realizou a movimentação	Data	N
saldoAnterior	Saldo anterior do caixa	Real(8,2) positivo	N
entrada	Valor pago pelo atendimento	Real(8,2) positivo	N
saida	Gastos a serem pagos	Real(8,2) positivo	N
saldoAtual	Saldo atual do caixa	Real(8,2) positivo	N
descricao	Descrição das entradas e saídas	Character(40)	N

Tipo Entidade	ProdutoDoSalao		
Descrição	Especificação do cadastro produto do salão		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
codigoProduto	Código de Identificação do produto	Inteiro(13) positivo	N
nome	Nome do produto	Character(80)	N
quantidade	Quantidade de produto em estoque	Inteiro (4) positivo	N

Relacionamentos

Tipo Relacionamento	Agenda		
Descrição	Relacionamento entre cliente e procedimento		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
data	Data de agendamento do procedimento	Data	N

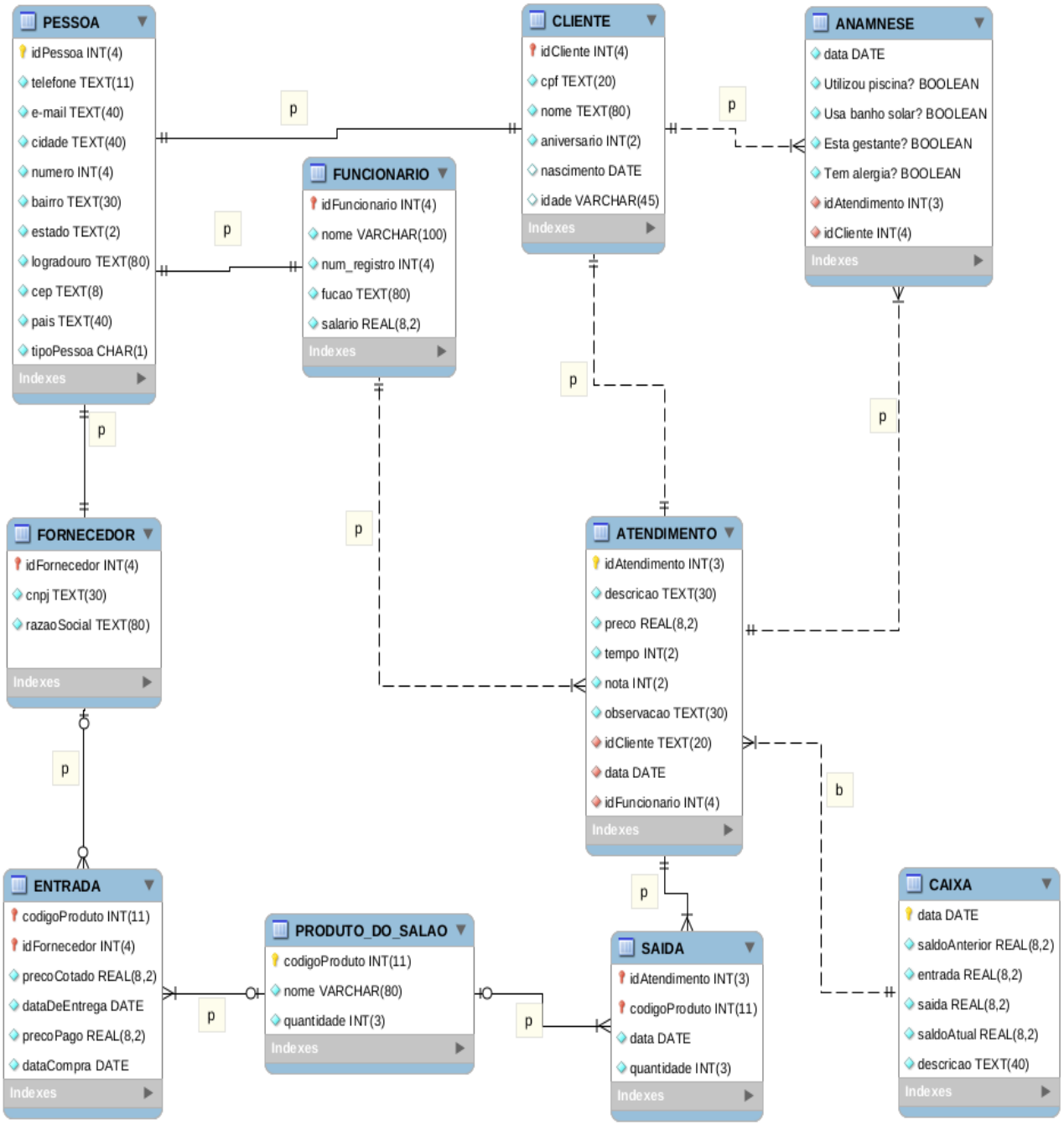
Tipo Relacionamento	Trabalha-Para
Descrição	Um funcionário trabalha realizando procedimentos para diversos clientes

Tipo Relacionamento	Creditar
Descrição	Valor que será pago pelo atendimento, o caixa receberá esse valor

Tipo Relacionamento	Saida		
Descrição	Feito o atendimento é registrado a quantidade gasta de produto para retirar do estoque		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
quantidade	Quantidade de produto utilizada	Inteiro(4) positivo	N
data	Data da saída do produto	Data	N

Tipo Relacionamento	Entrada		
Descrição	Relacionamento de entrada de produto no estoque		
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
precoCotado	Valor do preço cotado	Real(8,2) positivo	N
dataDeEntrega	Data da entrega dos produtos	Data	N
dataCompra	Data que o produto foi comprado	Data	N
precoPago	Preço pago anteriormente	Real(8,2) positivo	N

Diagrama Relacional



Dicionário de dados – Relacional

Tabela	Pessoa
Descrição	Especifica o cadastro de pessoa
Atributos	
Nome	**Descrição**
idPessoa	Número de identificação de pessoa
telefone	Número do telefone
email	Endereço eletrônico
cidade	Nome da cidade
numero	Número da residência
bairro	Nome do bairro
estado	Sigla do estado
logradouro	Nome do logradouro (rua, avenida, alameda etc.)
cep	Código de endereçamento postal
pais	Nome do país
tipoPessoa	Mostra o tipo de pessoa sendo: C - Cliente, F - Funcionário e N- Fornecedor

Tabela	Cliente
Descrição	Especificação do cadastro pessoa relativa ao cliente
Atributos	
Nome	**Descrição**
idCliente	Número de identificação do cliente herdada de idPessoa
cpf	Número do cpf do cliente
nome	Nome completo do cliente
nascimento	Data de nascimento

*Em Cliente existem mais dois valores - aniversário e idade - que serão obtidos à partir da data de nascimento, não serão salvos no banco de dados

Tabela	Funcionario
Descrição	Especificação do cadastro pessoa relativa ao funcionário
Atributos	
Nome	**Descrição**
idFuncionario	número de identificação do funcionário herdada de idPessoa
num_registro	Número de matrícula do funcionário
nome	Nome completo do funcionário
funcao	Função do funcionário
salario	Salário mensal

Tabela	Fornecedor
Descrição	Especificação do cadastro pessoa relativa ao fornecedor
Atributos	
Nome	**Descrição**
idFornecedor	Número de identificação do fornecedor herdada de idPessoa
cnpj	Cnpj da empresa
razaoSocial	Nome da empresa

Tabela	Atendimento
Descrição	Contém dados dos atendimentos
Atributos	
Nome	**Descrição**
idAtendimento	Código de Identificação do atendimento
descricao	Descreve o atendimento
preco	Preço do procedimento
tempo	Horas gastas no atendimento
nota	Nota que o cliente dá para o atendimento, de 0 a 10
observacao	Observação sobre o atendimento
idCliente	Referência ao identificador do cliente
idFuncionario	Referência ao identificador do funcionário
data	Referência a data do recebimento do pagamento

Tabela	Anamnese
Descrição	Conjunto de perguntas para o cliente responder antes de cada atendimento
Atributos	
Nome	**Descrição**
UtilizouPiscina	Pergunta se o cliente utilizou a piscina
FezBanhoDeSol	Pergunta se o cliente fez o banho solar
EstaGestante	Pergunta o cliente se é gestante
TemAlergia	Pergunta se o cliente tem alguma alergia
idCliente	Referência ao identificador do cliente
idAtendimento	Referência o número do atendimento
data	Data que foi preenchida a anamnese

Tabela	Caixa
Descrição	Contém as movimentações financeira diárias
Atributos	
Nome	**Descrição**
data	Data em que se realizou a movimentação
saldoAnterior	Saldo anterior do caixa
entrada	Valor pago pelo atendimento
saida	Gastos a serem pagos
descricao	Descrição das entradas e saídas

*Em Caixa existe saldoAtual que será obtido a partir do SaldoAnterior + transações de entrada ou saída do caixa, não será salvo no banco de dados.

Tabela	ProdutoDoSalao
Descrição	Especificação do cadastro produto do salão
Atributos	
Nome	**Descrição**
codigoProduto	Código de Identificação do produto
nome	Nome do produto
quantidade	Quantidade de produto em estoque

Tabela	Saída
Descrição	Feito o atendimento é registrado a quantidade gasta de produto para retirar do estoque
Atributos	
Nome	**Descrição**
quantidade	Quantidade de produto utilizada
data	Data da saída do produto
idAtendimento	Referência id do atendimento
codigoProduto	Referência do código do produto

Tabela	Entrada
Descrição	Relacionamento de entrada de produto no estoque
Atributos	
Nome	**Descrição**
precoCotado	Valor do preço cotado
dataDeEntrega	Data da entrega dos produtos
dataCompra	Data que o produto foi comprado
precoPago	Preço pago anteriormente
codigoProduto	Referência ao código do produto
idFornecedor	Referência à identificação do Fornecedor

Implementação em SQL

Primeiramente criamos o esquema salao onde são agrupadas as tabelas e outros objetos que pertencem ao mesmo Banco de Dados.

```
create schema salao;
```

Para utilizar o Banco de Dados salao temos que utilizar o comando use;

```
use salao;
```

Para criar as tabelas do Banco de Dados também utilizamos o comando CREATE mas agora associado com o comando TABLE no lugar de SCHEMA. É feita a verificação se a tabela já existe para criar a tabela com IF NOT EXISTS. É onde ficarão armazenados os dados que inclui o nome da tabela, seus atributos e suas restrições. Alguns atributos podem ser NOT NULL, atributo que não permite valor nulo, ou NULL atributo que permite valor nulo, padrão. Todas as tabelas contém chave primária PRIMARY KEY, algumas têm chave estrangeira FOREIGN KEY e UNIQUE INDEX chave derivada. Os dados podem ser apagados ou atualizados em cascata, DELETE/UPDATE CASCADE, ou no caso de haver alguma condição que impeça a atualização ou remoção é usado o comando DELETE/UPDATE ACTION ou RESTRICT . Foram criadas as seguintes tabelas:

Pessoa

```
CREATE TABLE IF NOT EXISTS `Pessoa` (
  `idPessoa` INT(4) NOT NULL,
  `telefone` CHAR(11) NOT NULL,
  `email` VARCHAR(50) NULL DEFAULT 'Sem email',
  `cidade` VARCHAR(50) NOT NULL,
  `numero` INT(5) NOT NULL,
  `bairro` VARCHAR(30) NOT NULL,
  `estado` CHAR(2) NOT NULL,
  `logradouro` VARCHAR(80) NOT NULL,
  `cep` CHAR(8) NOT NULL,
  `pais` VARCHAR(40) NOT NULL,
  `TipoPessoa` CHAR(1) NOT NULL,
  PRIMARY KEY (`idPessoa`))
ENGINE = InnoDB;
```

Cliente

```
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `Cliente` (
  `idCliente` INT(4) NOT NULL,
  `cpf` CHAR(11) NOT NULL,
  `nome` VARCHAR(100) NOT NULL,
  `nascimento` DATE NOT NULL,
  PRIMARY KEY (`idCliente`),
  UNIQUE INDEX `cpf_UNIQUE` (`cpf` ASC),
  CONSTRAINT `fk_CLIENTE_1`
    FOREIGN KEY (`idCliente`)
    REFERENCES `Pessoa` (`idPessoa`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Funcionario

```
CREATE TABLE IF NOT EXISTS `Funcionario` (
  `idFuncionario` INT(4) NOT NULL,
  `numRegistro` INT(4) NOT NULL,
  `nome` VARCHAR(45) NOT NULL,
  `fucao` VARCHAR(80) NOT NULL,
  `salario` REAL(8,2) NOT NULL,
  PRIMARY KEY (`idFuncionario`),
  UNIQUE INDEX `num_registro_UNIQUE` (`numRegistro` ASC),
  CONSTRAINT `fk_FUNCIONARIO_1`
    FOREIGN KEY (`idFuncionario`)
    REFERENCES `Pessoa` (`idPessoa`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Fornecedor

```
CREATE TABLE IF NOT EXISTS `Fornecedor` (
  `idFornecedor` INT(4) NOT NULL,
  `cnpj` CHAR(14) NOT NULL,
  `razaoSocial` VARCHAR(100) NOT NULL,
  UNIQUE INDEX `cnpj_UNIQUE` (`cnpj` ASC),
  PRIMARY KEY (`idFornecedor`),
  CONSTRAINT `fk_FORNECEDOR_1`
    FOREIGN KEY (`idFornecedor`)
    REFERENCES `Pessoa` (`idPessoa`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Caixa

```
CREATE TABLE IF NOT EXISTS `Caixa` (
  `data` DATE NOT NULL,
  `saldoAnterior` REAL(8,2) NOT NULL,
  `entrada` REAL(8,2) NOT NULL,
  `saida` REAL(8,2) NOT NULL,
  `descricao` VARCHAR(40) NOT NULL,
  PRIMARY KEY (`data`))
ENGINE = InnoDB;
```

Atendimento

```
CREATE TABLE IF NOT EXISTS `Atendimento` (
  `idAtendimento` INT(4) NOT NULL,
  `descricao` VARCHAR(50) NOT NULL,
  `preco` REAL(8,2) NOT NULL,
  `tempo` INT(4) NOT NULL,
  `nota` INT(2) NOT NULL,
  `observacao` VARCHAR(100) NOT NULL DEFAULT'Sem Observação',
  `data` DATE NOT NULL,
  `idFuncionario` INT(4) NOT NULL,
  PRIMARY KEY (`idAtendimento`),
  INDEX `fk_ATENDIMENTO_1_idx` (`data` ASC),
  INDEX `fk_ATENDIMENTO_FUNCIONARIO1_idx` (`idFuncionario` ASC),
  CONSTRAINT `fk_ATENDIMENTO_1`
    FOREIGN KEY (`data`)
    REFERENCES `Caixa` (`data`)
    ON DELETE RESTRICT
    ON UPDATE RESTRICT,
  CONSTRAINT `fk_ATENDIMENTO_FUNCIONARIO1`
    FOREIGN KEY (`idFuncionario`)
    REFERENCES `Funcionario` (`idFuncionario`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Anamnese

```
CREATE TABLE IF NOT EXISTS `Anamnese` (
  `idAtendimento` INT(3) NOT NULL,
  `idCliente` INT(4) NOT NULL,
  `UtilizouPiscina` TINYINT(1) NOT NULL,
  `FezBanhoDeSol` TINYINT(1) NOT NULL,
  `EstaGestante` TINYINT(1) NOT NULL DEFAULT False,
  `TemAlergia` TINYINT(1) NOT NULL,
  `data` DATE NOT NULL,
  INDEX `fk_ANAMNESE_1_idx` (`idAtendimento` ASC),
  INDEX `fk_ANAMNESE_2_idx` (`idCliente` ASC),
  PRIMARY KEY (`idAtendimento`, `idCliente`),
  CONSTRAINT `fk_ANAMNESE_1`
    FOREIGN KEY (`idAtendimento`)
    REFERENCES `Atendimento` (`idAtendimento`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_ANAMNESE_2`
    FOREIGN KEY (`idCliente`)
    REFERENCES `Cliente` (`idCliente`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

ProdutoSalao

```
CREATE TABLE IF NOT EXISTS `ProdutoSalao` (
  `codigoProduto` INT(13) NOT NULL,
  `nome` VARCHAR(100) NOT NULL,
  `quantidade` INT(3) NOT NULL,
  PRIMARY KEY (`codigoProduto`))
ENGINE = InnoDB;
```

Saida

```
CREATE TABLE IF NOT EXISTS `Saida` (
  `idAtendimento` INT(4) NOT NULL,
  `codigoProduto` INT(13) NOT NULL,
  `data` DATE NOT NULL,
  `quantidade` INT(4) NOT NULL,
  PRIMARY KEY (`idAtendimento`, `codigoProduto`),
  INDEX `fk_SAIDA_2_idx` (`codigoProduto` ASC),
  CONSTRAINT `fk_SAIDA_1`
    FOREIGN KEY (`idAtendimento`)
    REFERENCES `Atendimento` (`idAtendimento`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_SAIDA_2`
    FOREIGN KEY (`codigoProduto`)
    REFERENCES `ProdutoSalao` (`codigoProduto`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Entrada

```
CREATE TABLE IF NOT EXISTS `Entrada` (
  `codigoProduto` INT(11) NOT NULL,
  `idFornecedor` INT(4) NOT NULL,
  `precoCotado` REAL(8,2) NOT NULL,
  `dataDeEntrega` DATE NOT NULL,
  `precoPago` REAL(8,2) NOT NULL,
  `dataCompra` DATE NOT NULL,
  PRIMARY KEY (`idFornecedor`, `codigoProduto`),
  INDEX `fk_ENTRADA_1_idx` (`codigoProduto` ASC),
  CONSTRAINT `fk_ENTRADA_1`
    FOREIGN KEY (`codigoProduto`)
    REFERENCES `ProdutoSalao` (`codigoProduto`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_ENTRADA_2`
    FOREIGN KEY (`idFornecedor`)
    REFERENCES `Fornecedor` (`idFornecedor`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Exemplos de ALTER TABLE

O comando ALTER TABLE é utilizado para alterar dados de uma tabela. Aqui acrescentamos uma nova coluna na tabela Cliente chamada ClienteVip. E logo em seguida utilizamos o DROP para excluir a coluna adicionada anteriormente.

```
--  Acrescentar uma nova coluna chamada  ClienteVip na tabela Cliente

ALTER TABLE Cliente ADD COLUMN ClienteVip CHAR(1)   NOT NULL;

-- Apaga a colouna adicionada em cliente
ALTER TABLE Cliente DROP ClienteVip CASCADE;
```

Para utilizarmos mais o comando ALTER TABLE e o DROP criamos duas tabelas (ClienteVip e ServicoPrestado) para que não houvesse mudanças no Banco de Dados criados.

```
CREATE TABLE ClienteVip (
    codigoCliente           INT(2)        NOT NULL,
    nomeCliente             VARCHAR(80)   NOT NULL,
    telefoneCliente         CHAR(11)      NOT NULL,
    Numservico              INT(2)        NOT NULL,
    PRIMARY KEY (codigoCliente)
);

CREATE TABLE ServicoPrestado(
idServico           INT(2)          NOT NULL,
tipoServico         VARCHAR(80)     NOT NULL,
precoServico        REAL(8,2)       NOT NULL,

PRIMARY KEY (idServico)
);
```

```sql
-- Adicionar uma chave estrangeira à tabela ClienteVip

ALTER TABLE ClienteVip ADD CONSTRAINT
fk_ser FOREIGN KEY (NumServico) REFERENCES
ServicoPrestado (idServico);

-- Remover a chave estrangeira da tabela ClienteVip

ALTER TABLE ClienteVip DROP CONSTRAINT
fk_ser;

-- Apaga as tabelas ClienteVip e ServicoPrestado
DROP TABLE ClienteVip;
DROP TABLE ServicoPrestado;
```

Exemplos de Insert

Foram colocados diversos dados no Banco de Dados para fazer consultas e operações. Para colocar os dados em uma tabela temos que utilizar INSERT INTO <nome da tabela> VALUES <dados a serem colocados em ordem igual a criação da tabela>

Inserção de dados na tabela Pessoa

```sql
insert into Pessoa
values (1,'74563821','josealberto@gmail.com','Conconhas',54,'Bairro da Conceição','MG','bloco A', 37493033,'Brasil','F');
insert into Pessoa
values (2,84758392,'Almir@gmail.com','Barcatua',23,'Vila Itapemirim','MG','Proximo a esquina', 3748374,'Brasil','F');
insert into Pessoa
values (3,93856283,'bennet@gmail.com','Conconhas',54,'Bairro da Conceição','MG','bloco A', 37493033,'Brasil','F');
insert into Pessoa
values (4,94726483,'jessicaMartins@gmail.com','Conconhas',23,'Rondente','MG','Frente', 87574839,'Brasil','F');
insert into Pessoa
values (5,93827485,'Ricardo@gmail.com','Conconhas',43,'Vale dos Itambés','MG','A Direita da via São Sebastião', 05870483,'Brasil','F');
insert into Pessoa
values (6,74563821,'Alana@gmail.com','Conconhas',344,'Gato Preto','MG','A Direita do mercado vila nova', 93847920,'Brasil','F');
insert into Pessoa
values (7,74563821,'Guilhermedaz@gmail.com','Lavras',233,'Castelo','MG','Proximo a Ipê', 83775839,'Brasil','C');
insert into Pessoa
values (8,94837458,'improvAlvaro@gmail.com','Conconhas',23,'Expertilho','MG','Proximo a saída da cidade', 74657382,'Brasil','N');
insert into Pessoa
values (9,92837465,'delegarEntregas@gmail.com','Conconhas',34,'Rio Negro','MG','Distrito industrial', 42849483,'Brasil','N');
insert into Pessoa
values (10,83746272,'Emaep@gmail.com','Conconhas',234,'Rio Negro','MG','Distrito industrial', 93847292,'Brasil','N');
insert into Pessoa
values (11,74563821,'josiclelia@gmail.com','Ibituruna',78,'Santa Cruz','MG','Praça', 3773839,'Brasil','C');
```

Inserção de dados na tabela Fornecedor

```
insert into Fornecedor
values (9,84756374483940,'Delegar Entregas Inc.');
insert into Fornecedor
values (10,92834445833403,'Emaep Serviços e entregas');
insert into Fornecedor
values (8,52523868900091,'ImprovAll');
insert into Fornecedor
values (15,50825601000200,'Produtos Skala');
insert into Fornecedor
values (16,17970378250009,'Lola Cosméticos');
```

Inserção de dados na tabela Cliente

```
insert into Cliente
values (7,93384637293,'Guilherme Pires da Silva','1986-11-04');
insert into Cliente
values (11,83725347586,'Josiclélia Saraiva','1982-05-11');
insert into Cliente
values (12,20473648495,'Tadeu de Melo Splinder','1992-10-20');
insert into Cliente
values (13,10472574758,'Cleiton Lula da Silva','1989-11-04');
insert into Cliente
values (14,93846700394,'Claudinara Vieira Maia','1986-03-04');
```

Inserção de dados na tabela Funcionario

```
insert into Funcionario
values (1,21,'José Alberto de Fraguas','Hidratação',112);
insert into Funcionario
values (2,9283,'Almir Nogueira de Pádua','Tintura',1200.75);
insert into Funcionario
values (3,4435,'Bernardo Nogueira Neturdino','Corte e Tintura',2100.23);
insert into Funcionario
values (4,8274,'Jéssica Martins Fontes','Maquiagem',1780.50);
insert into Funcionario
values (5,8375,'Ricardo Spindula de Jesus','Corte e Hidratação',2148.00);
insert into Funcionario
values (6,9978,'Alanna Tavares Bragança','Manicure',1100.00);
```

Inserção de dados na tabela Caixa

```
insert into Caixa
values('2020-08-11',2300.00,30.00,00.00,'recebimento serviço manicure');
insert into Caixa
values('2020-08-12',2300.00,80.00,00.00,'recebimento escova');
insert into Caixa
values('2020-08-13',2300.00,60.00,00.00,'recebimento de hidratação');
insert into Caixa
values('2020-08-14',2300.00,100.00,00.00,'recebimento de coloração');
insert into Caixa
values('2020-08-15',2300.00,00.00,250.00,'Pagamento fornecedor');
insert into Caixa
values('2020-08-16',2300.00,60.00,00.00,'recebimento de hidratação');
insert into Caixa
values('2020-08-17',2300.00,100.00,00.00,'recebimento de coloração');
```

Inserção de dados na tabela Atendimento

```
insert into Atendimento
values (1,'Coloração',100.00,120,10,'Observação existente','2020-08-11',6);
insert into Atendimento
values (2,'Manicure',60.00,45,09,'Observação existente','2020-08-11',5);
insert into Atendimento
values (3,'Escova',150.00,90,08,'Observação existente','2020-08-12',4);
insert into Atendimento
values (4,'Maquiagem',250.00,120,09,'Observação existente','2020-08-13',3);
insert into Atendimento
values (5,'Penteado',300.00,180,10,'Observação existente','2020-08-14',2);
insert into Atendimento
values (6,'Manicure',60.00,45,09,'Observação existente','2020-08-15',2);
insert into Atendimento
values (7,'Hidratação',80.00,120,10,'Observação existente','2020-08-16',2);
insert into Atendimento
values (8,'Hidratação',80.00,120,9,'Observação existente','2020-08-17',1);
```

Inserção de dados na tabela Anamnese

```
insert into Anamnese
values(1,7,0,0,0,1,'2020-08-07');
insert into Anamnese
values(2,11,0,0,0,1,'2020-08-08');
insert into Anamnese
values(3,14,0,0,0,1,'2020-08-09');
insert into Anamnese
values(4,13,0,0,0,1,'2020-08-10');
insert into Anamnese
values(5,12,0,0,0,1,'2020-08-11');
insert into Anamnese
values(6,11,0,0,0,1,'2020-08-12');
insert into Anamnese
values(7,14,0,0,0,1,'2020-08-13');
insert into Anamnese
values(8,7,0,0,0,1,'2020-08-14');
```

Inserção de dados na tabela ProdutoSalao

```
insert into ProdutoSalao
values (23,'Xampu Pantene',30);
insert into ProdutoSalao
values (24,'Condicionador Pantene',30);
insert into ProdutoSalao
values (10,'Esmalte Risque Rosa',5);
insert into ProdutoSalao
values (11,'Esmalte Risque Preto',7);
insert into ProdutoSalao
values (12,'Esmalte Risque Vermelho',10);
insert into ProdutoSalao
values (5,'Lixa para unha',25);
insert into ProdutoSalao
values (6,'Alicate para cutícula',10);
insert into ProdutoSalao
values (9,'Tinta Colorama Loiro',15);
insert into ProdutoSalao
values (13,'Mascara de Hidratação Pantene',8);
```

Inserção de dados na tabela Entrada

```
insert into Entrada
values (23,15,'15.00','2020-02-02','14.00','2020-01-18');
insert into Entrada
values (24,15,'16.90','2020-02-02','16.00','2020-01-18');
insert into Entrada
values (10,16,'25.00','2020-02-21','24.50','2020-01-21');
insert into Entrada
values (10,9,'5.99','2020-02-21','4.99','2020-01-21');
insert into Entrada
values (11,9,'5.99','2020-02-21','4.99','2020-01-21');
insert into Entrada
values (12,9,'5.99','2020-02-02','4.99','2020-01-22');
insert into Entrada
values (9,10,'45.00','2020-02-02','42.50','2020-01-22');
```

Inserção de dados na tabela Saída

```
insert into Saida
values (1,23,'2020-08-10',1);
insert into Saida
values (1,24,'2020-08-11',1);
insert into Saida
values (2,10,'2020-08-12',1);
insert into Saida
values (3,5,'2020-08-13',1);
insert into Saida
values (4,6,'2020-08-14',1);
insert into Saida
values (5,9,'2020-08-15',1);
insert into Saida
values (6,11,'2020-08-16',1);
insert into Saida
values (7,12,'2020-08-17',1);
```

Exemplos de atualização de dados nas tabelas

O comando utilizado para atualizar os dados de uma tabela é o UPDATE <nome da tabela> SET <novo dado> WHERE <dado atual >

```
--  atualizando o id do fornecedor de num 10 para num 2
update Fornecedor
set idFornecedor = 2
where idFornecedor = 10;
```

```
-- atualizando o salário do funcionario de nome 'Jéssica Martins Fontes'
update Funcionario
set salario = salario * 1.1
where nome = 'Jéssica Martins Fontes';
```

```
-- alterando a data de nascimento do Cliente pelo cpf
update Cliente
set nascimento = '1986-11-24'
where cpf = '93384637293';
```

```
-- atualiza o nome do produto do Fornecedor de id num 2
update ProdutoSalao
set nome = 'Xampu Pantene Nova Versão'
where codigoProduto in (select codigoProduto
                        from Entrada
                        where idFornecedor = 2);
```

```sql
-- houve troca do funcionario que realizou o atendimento então
-- atualizamos o id do funcionário

select *
from Atendimento;

update Atendimento
set idFuncionario = '5'
where idFuncionario in (select idFuncionario
                        from Funcionario
```

Exemplos de exclusão de dados

Para excluir dados da tabela utilizamos o comando DELETE FROM <nome da tabela> WHERE <condição do dado a ser excluído>. É utilizado quando se deseja deletar uma ou mais linhas de uma tabela.

```sql
-- Exclui funcionários de nome José de Fraguas
DELETE FROM Funcionario
WHERE nome = 'José Alberto de Fraguas';

-- Exclui Pessoa de id igual a 10
DELETE FROM Pessoa
WHERE idPessoa = 10;

-- Exclui Cliente com data de nascimento igual 1982-05-11
DELETE FROM Cliente
WHERE nascimento='1982-05-11';

-- Exclui Funcionario que tem salario maior que 2000,00
DELETE FROM Funcionario
WHERE salario>2000.00;
select *
from Funcionario;
-- Apagando atendimentos feito pelo funcionário de id número 3
DELETE FROM Atendimento
WHERE idFuncionario IN (SELECT idFuncionario
            FROM Funcionario
            WHERE idFuncionario = 6);
```

Exemplo de consultas no Banco de Dados

Para consultar dados em uma tabela há diversas maneiras e comandos específicos, segue abaixo exemplos das consultas:

SELECT - seleciona quais atributos serão exibidos

FROM - qual tabela será consultada

WHERE - condição que atende a solicitação

ORDER BY - seleciona por uma ordem no caso crescente

BETWEEN - entre valores específicos

COUNT(*) - conta a quantidade de linhas

MAX() - retorna o valor máximo entre uma quantidade de dados

MIN() - retorna o valor mínimo entre uma quantidade de dados

AVG() - retorna a média de uma quantidade de dados

LIKE% - compara uma cadeia de caracteres, se houver na busca uma parte da palavra que compõe essa cadeia retorna o valor

GROUP BY - organiza por grupos distintos que possuem uma característica em comum

JOIN - une duas tabelas de acordo com um atributo em comum

LEFT/ RIGTH OTHER JOIN - une duas tabelas mesmo de não houver atributos em comum, LEFT os atributos à esquerda recebem valor NULL se não houver atributos em comum e RIGHT os atributos à direita recebem valor NULL se não houver atributos em comum.

ALL - compara todos dado uma condição

IS NULL - verifica se o valor é nulo

```sql
-- --  selecionando o produto do código de número 23
select nome, quantidade
from ProdutoSalao
where codigoProduto = 23;


-- -- selecionando os produtos do fornecedor de código de número 6

select P.nome
from ProdutoSalao P join Entrada E on P.codigoProduto = E.codigoProduto
where E.idFornecedor = 6;
select nome, nascimento
from Cliente
order by nome;

-- selecionando os produtos com valores entre R$10 e R$80
select P.nome, E.precoPago
from Entrada E join ProdutoSalao P on P.codigoProduto = E.codigoProduto
where E.precoPago between 10.0 and 80.0;

-- contando total funcionários, maior, menor salario e média dos salários
select count(*), max(salario), min(salario), avg(salario)
from Funcionario;
```

```sql
-- recupera a quandidade de atendimento com duração de 120 minutos
select count(*) AS Quantidade from `Atendimento` group by tempo having tempo = 120;

-- recupera o nome o preço e a nota de todos os atendimentos exeto os que utilizaram o produto de codigo 23 e aplica
-- desconto de 20%;
select descricao, ROUND((preco*0.8), 2) AS preco, nota from `Atendimento`
where `idAtendimento` in(select `idAtendimento` from `Saida` where not codigoProduto=23);

-- recupera o nome e salários dos funcionário de exerce a função de manicure e que recebe um salario entre 1000,40 e 2100,00
select salario, nome From `Funcionario` where salario between 1000.40 and 2100.00 and fucao='Manicure'
order by salario desc;

-- recupera o nome e a quantidade dos produtos que tem "Esmalte" no nome
select nome, quantidade from ProdutoSalao PS join Entrada EN on PS.codigoProduto = EN.codigoProduto
where nome like "%Esmalte%" order by quantidade desc;

-- recupera o idFornecedor e o codigo do produto Que não realizou entregas
select F.idFornecedor, codigoProduto from Entrada E right outer join Fornecedor F
on E.idFornecedor = F.idFornecedor where E.codigoProduto is null;

-- recupera a cidade e razao social dos fornecedores localizado em Conconha ou Lavras;
select P.cidade, F.razaoSocial from Pessoa P join Fornecedor F on P.idPessoa = F.idFornecedor
where P.cidade='Conconhas' or p.cidade='Lavras';
```

```sql
-- recupera o nome dos funcionários dos clientes e funcionarios
-- e a razao social dos fornecedores
select F.nome, P.TipoPessoa
from Funcionario F join Pessoa P on P.idPessoa = F.idFuncionario
union
select C.nome, P.TipoPessoa
from Cliente C join Pessoa P on P.idPessoa = C.idCliente
union
select N.razaoSocial, P.TipoPessoa
from Fornecedor N join Pessoa P on P.idPessoa = N.idFornecedor;

-- lista os produtos com saída menor que 10

select nome,quantidade
from ProdutoSalao
where quantidade < all
   (select quantidade
    from ProdutoSalao
    where quantidade>10);
```

```sql
-- lista os produtos com saída menor que 2000

select nome, salario
from Funcionario
where salario < all
   (select salario
    from Funcionario
    where salario>2000);
```

Exemplos de criação de de 3 visões

Visões são feitas para que o usuário tenha acesso a somente alguns atributos do banco de dados, dessa forma é possível limitar o que cada usuário pode acessar no banco.

```sql
-- Cria uma visão com a quantidade de funcionario e a soma dos salario de todos os funcionarios
CREATE VIEW FunQuant (qtdeFuncs, somaSalario) AS
SELECT COUNT(*), SUM(salario)
FROM Funcionario;
-- teste
SELECT *
FROM FunQuant;
```

```sql
-- Cria uma visão com o id do Atendimento, descrição e o tempo
CREATE VIEW Atend (idAtendimento, descricao,tempo) AS
SELECT idAtendimento, descricao,tempo
FROM Atendimento;
-- teste
SELECT *
FROM Atend;
```

```sql
-- Cria uma visão com id do funcionario e o id do atendimento que o funcionario fez
CREATE VIEW FuncAten AS
SELECT idFuncionario,idAtendimento
FROM Funcionario NATURAL JOIN Atendimento
WHERE nome = 'José Alberto de Fraguas';
-- teste
SELECT *
FROM FuncAten;
```

Exemplos de criação de usuários

Diferentes tipos de usuários são criados para que possa ter acesso, modificação e atualização dos dados, dessa forma é possível dar permissões a diferentes usuários de diferentes níveis de acesso.

```sql
-- Cria permissão para o usuario Jose
CREATE USER 'Jose'@'localhost' IDENTIFIED BY '1234';

-- Cria permissão para o usuario Juliana
CREATE USER 'Juliana'@'localhost' IDENTIFIED BY '1313';
```

O comando GRANT concede privilégios para um usuário específico.

```sql
-- Concede ao usuario Jose todos os privilegios da tabela
GRANT ALL ON salao.Pessoa TO 'Jose'@'localhost';

-- Concede ao usuario Jose apenas a seleção na tabela Atendimento
GRANT SELECT ON salao.Atendimento TO 'Jose'@'localhost';

-- Concede ao usuario Juliana apenas o update no atributo razaoSocial na tabela Fornecedor
GRANT UPDATE (razaoSocial) ON salao.Fornecedor TO 'Juliana'@'localhost';

-- Concede ao usuario Juliana todos os privilegios na tabela Caixa
GRANT ALL ON salao.Caixa TO 'Juliana'@'localhost';
```

O comando REVOKE é o contrário do GRANT ele retira os privilégios do usuário.

```sql
-- Revoga do usuario Juliana o direito de alterar a coluna razaoSocial na tabela Fornecedor
REVOKE UPDATE(razaoSocial) ON salao.Fornecedor FROM 'Juliana'@'localhost';

-- Revoga do usuario Jose o direito de fazer seleção na tabela Atendimento
REVOKE SELECT ON salao.Atendimento FROM 'Jose'@'localhost';
```

Exemplos de 3 procedimentos/funções

O comando PROCEDURE é utilizado para guardar consultas que geralmente são usadas repetidamente, tendo fácil acesso.

```sql
/*Mostra as informações de fornecimento do produto "Lola Cosméticos"*/
DELIMITER //
CREATE PROCEDURE entradasProdutos(IN inProduto varchar(30))
BEGIN
    SELECT codigoProduto, idFornecedor, dataDeEntrega, precoPago
        FROM Entrada NATURAL JOIN Fornecedor WHERE razaoSocial= inProduto
    ORDER BY dataDeEntrega desc;
END //
DELIMITER ;
CALL entradasProdutos("Lola Cosméticos");
```

```sql
/*Mostra os "status" dos atendimento de acordo com a nota*/
DELIMITER //
 CREATE PROCEDURE statusDoAtendimento()
 BEGIN
    SELECT nota,
    CASE
        WHEN nota < 10 THEN 'Bom'
        WHEN nota < 6 THEN 'Suficiente'
        WHEN nota < 3 THEN 'Treinamento'
        WHEN nota < 2 THEN 'Demitido'
        ELSE 'Ótimo'
        END AS StatusAtendimento
    FROM Atendimento;
 END //
 DELIMITER ;

call statusDoAtendimento();
```

```sql
/*Aplica Descontos em Atendimento que for realizado em 90 minutos ou menos e acresenta informação nos atributos Descrição
correspondentes*/

DELIMITER //
CREATE PROCEDURE descontoTempoDeAtendimento()
BEGIN
    IF(90 <= ANY(Select tempo from Atendimento
        where descricao = "Escova")) THEN UPDATE Atendimento
            SET descricao = CONCAT (descricao, " ", "(20% de Desconto)"), preco = preco * 0.8
        where tempo <= 90;
    END IF;
END //
DELIMITER ;

call descontoTempoDeAtendimento();
```

Exemplos de 3 triggers

Os Triggers é um comando específico que é disparado por um evento de modificação de dados podendo ser INSERT, UPDATE ou DELETE , em uma tabela pré definida.

```sql
    -- Cria um trigger de auditoria de alterações de salários dos Funcionarios
    --  cria a tabela de registros de auditoria:

CREATE TABLE auditoriaSalario (
    idAuditoria     INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    idFuncionario       INT NOT NULL,
    nome     VARCHAR(50) NOT NULL,
    salarioAnterior FLOAT NOT NULL,
    salarioNovo     FLOAT NOT NULL,
    user VARCHAR(20) NOT NULL,
    dataHora datetime NOT NULL
    );
```

```sql
-- Cria um trigger que registra mudanças de salário na tabela Funcionario

DROP TRIGGER after_Funcionario_update;

DELIMITER $$
CREATE TRIGGER after_Funcionario_update
AFTER UPDATE ON Funcionario
FOR EACH ROW
BEGIN
  IF OLD.salario != NEW.salario THEN
    INSERT INTO auditoriaSalario (idFuncionario, nome, salarioAnterior, salarioNovo, user, dataHora)
    VALUE (NEW.idFuncionario, NEW.nome, OLD.salario, NEW.salario, USER(), NOW());
  END IF;
END $$
DELIMITER ;
```

```sql
-- teste o trigger emitindo alterações (ou não) de salários

SELECT idFuncionario, nome, salario
FROM Funcionario;

UPDATE Funcionario
SET salario = 2000.00
WHERE idFuncionario = 2;

SELECT idFuncionario, nome, salario
FROM Funcionario;
SELECT * FROM auditoriaSalario;
```

```sql
/*
Trigger que verifica o caracter a ser informado no atributo TipoPessoa da tabela Pessoa
É permitidos os valores C,F,N,1,2 ou 3. Se for inserido:
- 1 é convertido para C (Cliente)
- 2 é convertido para F (Funcionario)
- 3 é convertido para N (Fornercedor).
*/

DROP TRIGGER Pessoa_insert;
```

```
DELIMITER $$
CREATE TRIGGER Pessoa_insert
BEFORE INSERT ON Pessoa FOR EACH ROW
BEGIN
   IF (new.TipoPessoa = '1') THEN
      SET new.TipoPessoa = 'C';

   ELSEIF (new.TipoPessoa = '2') THEN
      SET new.TipoPessoa = 'F';

   ELSEIF (new.TipoPessoa = '3') THEN
      SET new.TipoPessoa = 'N';

   ELSEIF (new.TipoPessoa NOT IN ('C','F','N')) THEN
      SIGNAL SQLSTATE '45000' SET message_text = 'Caracter Invalido!';
   END IF;
END;
$$
DELIMITER ;
```

```
/*
Teste o trigger inserindo novos funcionários
*/

INSERT INTO Pessoa (idPessoa,telefone,email,cidade,numero,bairro,estado,logradouro,cep,pais,TipoPessoa)
VALUES (100,'3588231452','ms@gmail.com','Lavras',850,'centro','MG','Av. Das Flores','37200000','Brasil','1');

INSERT INTO Pessoa (idPessoa,telefone,email,cidade,numero,bairro,estado,logradouro,cep,pais,TipoPessoa)
VALUES (101,'3598877710','sa@gmail.com','Tiradentes',02,'centro','MG','Av. Dom Pedro','37201000','Brasil','C');

INSERT INTO Pessoa (idPessoa,telefone,email,cidade,numero,bairro,estado,logradouro,cep,pais,TipoPessoa)
VALUES (103,'3587221400','jp@gmail.com','Lavras',02,'Nova Era','MG','Rua São Jõao ','372000','Brasil','2');

INSERT INTO Pessoa (idPessoa,telefone,email,cidade,numero,bairro,estado,logradouro,cep,pais,TipoPessoa)
VALUES (104,'359872440','wa@gmail.com','Varginha',02,'Centenario','MG','Rua Castelo Branco ','37200001','Brasil','3');

INSERT INTO Pessoa (idPessoa,telefone,email,cidade,numero,bairro,estado,logradouro,cep,pais,TipoPessoa)
VALUES (105,'3587134400','rt@gmail.com','Lavras',02,'Nova Lavras','MG','Rua sete ','37200000','Brasil','9');

SELECT * FROM Pessoa;
```

```sql
-- Cria um trigger que controla o salário dos novos funcionários,
-- sendo que o salário não ultrapasse R$2500,00.


DROP TRIGGER Funcionario_insert;

DELIMITER $$
CREATE TRIGGER Funcionario_insert
BEFORE INSERT ON Funcionario
FOR EACH ROW
BEGIN
   IF NEW.salario > 2500 THEN
      SIGNAL SQLSTATE '45000' SET message_text = 'Salário ultrapassou R$2.500,00';
   END IF;
END $$
DELIMITER ;
```

```sql
-- Teste o trigger inserindo novos funcionários na tabela


INSERT INTO Pessoa
VALUES (200,'357725512','hd@gmail.com','Lavras',75,'Cetro','MG','Rua dez','37200000','Brasil','F');
INSERT INTO Funcionario
VALUES (200,100,'Helena Dias','cabelereira',1500.00);

SELECT * FROM Funcionario;

INSERT INTO Pessoa
VALUES (201,'3598063524','Vi@gmail.com','Lavras',05,'Estação','MG','Rua São Paulo','37200000','Brasil','F');
INSERT INTO Funcionario
VALUES (201,101,'Vini Campos','Cabelereiro',3000.00);

SELECT * FROM Funcionario;
```