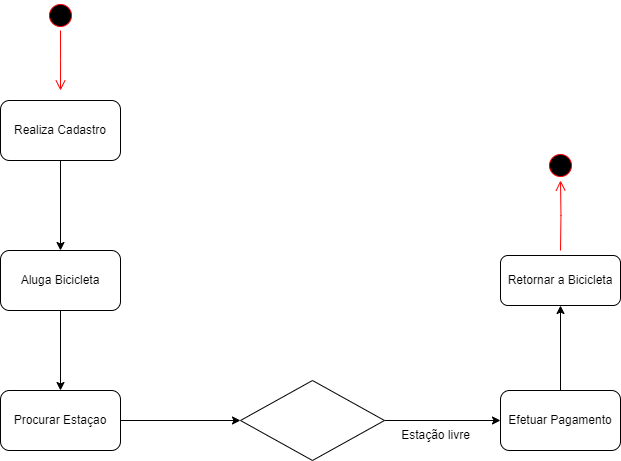
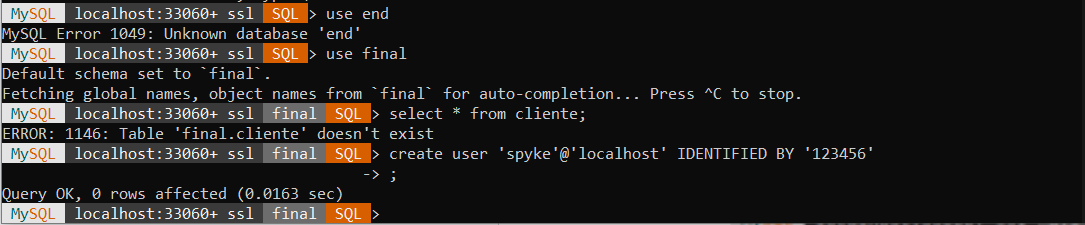
****

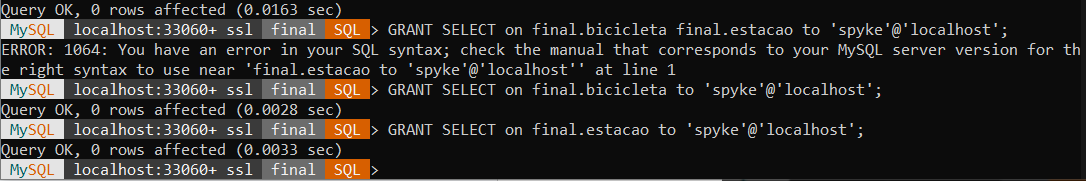
**Criação de Usuário**

**Os usuários precisam de acesso ao sistema para alugar bicicletas**

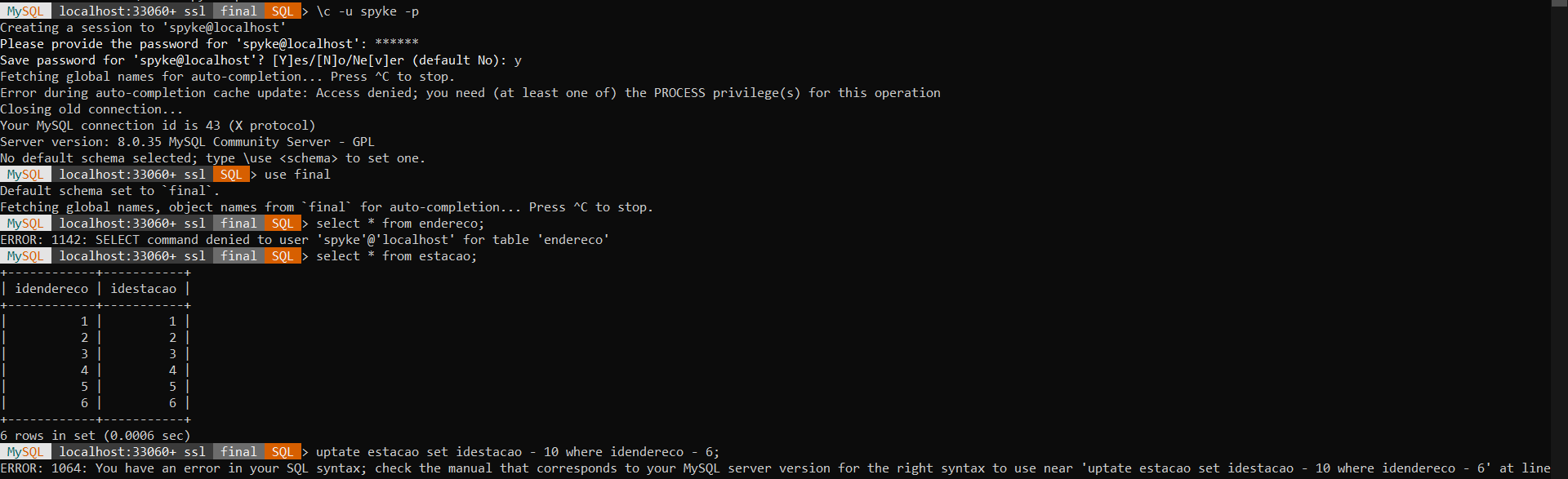
**Criar o usuário:**

****Criando o usuário “Spyke”.

**Atribuído permissões:**

****Atribuir direitos de visualização às tabelas de bicicletas e estações ao usuário "SPYKE"

**Testar usuário:**

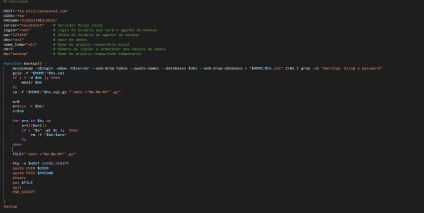
****Na imagem acima, o usuário está tentando visualizar uma tabela que não é permitida. Ele então olhou para a tabela à qual as permissões foram atribuídas.

Ele então tentou atualizar as informações, mas ocorreu um erro devido às suas funções.

**Backup**

Para os backups a melhor forma é usar o crontab, que fará os backup diariamente salvando em um arquivo compactado.

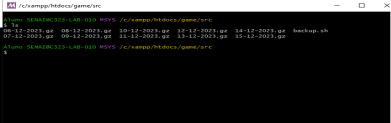
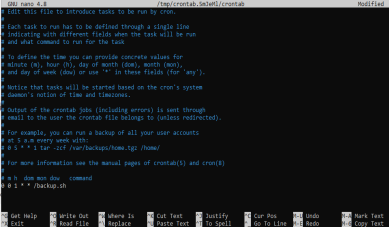
Esses arquivos também são sincronizados com outras máquinas via ftp, feito nativamente nos servidores linux com arquivo shell, usando os comandos ftp e crontab. **Código shell:**

****A imagem acima é um código atribuído no shell para gerar o Backup diariamente.

**Execução:**

****A imagem acima é a execução do backup.

Agendando tarefas com o cron:



**Uso do sistema**

Abaixo está uma breve descrição de procedimentos, funções e atribuições de gatilhos.

Para alugar uma bicicleta, o usuário seleciona o modelo e a estação onde irá retirá-la, e o sistema por sua vez lista a disponibilidade e verifica o aluguel verificando a disponibilidade da estação.

Durante o processo de locação, as informações são salvas na tabela Aluguel, que contém as informações exigidas pelo sistema, como data de início, temporada de início, etc. O SQL também possui mecanismos para evitar inconsistências devido a dados incorretos. Além de atualizar a disponibilidade de cada estação: o saldo da conta e o valor correspondente ao tempo decorrido.

Durante o período de depósito, todo o histórico de depósitos é salvo na tabela “pagamento” e o saldo atual é calculado no próprio sql e atualizado na tabela “usuário”.

Uma adição a esta explicação pode ser encontrada nas linhas de código abaixo..

**Código do Banco de dados**

**Criação do Banco de dados com a tabela:**

CREATE DATABASE final;

USE final;

CREATE TABLE cliente (

idcliente INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

nome\_cliente VARCHAR(12),

telefone\_cliente VARCHAR(20),

data\_nascimento DATE,

saldo\_conta DECIMAL(10, 2),

cpf\_cliente VARCHAR(11)

);

CREATE TABLE estacao\_bike (

id\_endereco INT,

id\_estacao INT PRIMARY KEY

);

CREATE TABLE bicicleta\_info (

id\_bicicleta INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

modelo\_bicicleta VARCHAR(100),

preco\_hora FLOAT,

status\_bicicleta VARCHAR(20),

id\_estacao INT

);

CREATE TABLE locacao (

id\_locacao INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

id\_cliente INT,

id\_bicicleta INT,

estacao\_inicial INT,

estacao\_final INT,

data\_inicio DATETIME,

data\_fim DATETIME,

valor\_total DECIMAL(10, 2),

status\_locacao VARCHAR(20)

);

CREATE TABLE endereco\_local (

id\_endereco INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

bairro\_local VARCHAR(100),

numero\_local INT,

cep\_local INT,

rua\_local VARCHAR(100),

id\_cliente INT,

id\_estacao INT

);

CREATE TABLE transacao\_pagamento (

id\_transacao INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

saldo\_conta DECIMAL(10, 2),

valor\_transacao DECIMAL(10, 2),

metodo\_pagamento VARCHAR(25),

id\_cliente INT

);

CREATE TABLE locacao\_estacao (

id\_locacao INT,

id\_estacao INT,

PRIMARY KEY (id\_locacao, id\_estacao)

);

ALTER TABLE bicicleta\_info

ADD

FOREIGN KEY (id\_estacao) REFERENCES estacao\_bike (id\_estacao);

ALTER TABLE locacao

ADD

FOREIGN KEY (id\_bicicleta) REFERENCES bicicleta\_info (id\_bicicleta);

ALTER TABLE locacao

ADD

FOREIGN KEY (id\_cliente) REFERENCES cliente (idcliente);

ALTER TABLE endereco\_local

ADD

FOREIGN KEY (id\_cliente) REFERENCES cliente (idcliente);

ALTER TABLE endereco\_local

ADD

FOREIGN KEY (id\_estacao) REFERENCES estacao\_bike (id\_estacao);

ALTER TABLE transacao\_pagamento

ADD

FOREIGN KEY (id\_cliente) REFERENCES cliente (idcliente);

ALTER TABLE locacao\_estacao

ADD

FOREIGN KEY (id\_locacao) REFERENCES locacao (id\_locacao);

ALTER TABLE locacao\_estacao

ADD

FOREIGN KEY (id\_estacao) REFERENCES estacao\_bike (id\_estacao);

-- Inserção de dados.

INSERT INTO

cliente (

id\_cliente,

nome\_cliente,

telefone\_cliente,

data\_nascimento,

saldo\_conta,

cpf\_cliente

)

VALUES

(

1,

"Ana",

77999641211,

"1990-02-27",

200,

05055574432

),

(

2,

"João",

77999645411,

"2000-03-27",

50.6,

06045574436

),

(

3,

"Fernanda",

77988941211,

"1992-04-26",

250,

07050574232

),

(

4,

"Lucas",

77998655211,

"1999-05-10",

70,

17405557443

),

(

5,

"Isabela",

77998401211,

"1995-06-27",

185,

12480574432

);

INSERT INTO

estacao\_bike(id\_endereco, id\_estacao)

VALUES

(1, 1),

(2, 2),

(3, 3),

(4, 4),

(5, 5),

(6, 6);

INSERT INTO

bicicleta\_info(

id\_bicicleta,

modelo\_bicicleta,

preco\_hora,

status\_bicicleta,

id\_estacao

)

VALUES

(1, 'Caloi', 15, 'disponivel', 1),

(2, 'Monark', 15, 'disponivel', 1),

(3, 'Oxford', 15, 'disponivel', 1),

(4, 'Scott', 15, 'disponivel', 1),

(5, 'Specialized', 15, 'disponivel', 1),

(

6,

'GT',

15,

'disponivel',

1

);

INSERT INTO

locacao (

id\_locacao,

id\_cliente,

id\_bicicleta,

estacao\_inicial,

estacao\_final,

data\_inicio,

data\_fim,

valor\_total,

status\_locacao

)

VALUES

(

1,

5,

1,

3,

NULL,

"2023-10-10 19:00:00",

NULL,

NULL,

"em andamento"

),

(

2,

1,

2,

2,

3,

"2023-10-1 10:00:00",

"2023-10-1 13:24:20",

11,

2.45,

"concluida"

),

(

3,

2,

2,

1,

4,

"2023-10-110:00:00",

"2023-10-1 13:24:20",

2.45,

"concluida"

),

(

4,

3,

1,

2,

3,

"2023-10-1019:00:00",

"2023-10-10 20:00:00",

10.5,

"concluida"

),

(

5,

4,

1,

2,

2,

"2023-10-1019:00:00",

"2023-10-10 20:00:00",

16.15,

"concluida"

);

INSERT INTO

endereco\_local(

id\_endereco,

bairro\_local,

numero\_local,

cep\_local,

rua\_local,

id\_cliente,

id\_estacao

)

VALUES

(

1,

"Centro",

222,

45000452,

"Rua da barragem",

12,

1,

1

),

(

2,

"Candeias",

641,

45120000,

"Rua A",

2,

2

),

(

3,

"Jurema",

20,

45000785,

"TravessadaConquista",

3,

3

),

(

4,

"MoradadosPássaros",

452,

40000478,

"Av Don Juan",

4,

4

),

(

5,

"Recreio",

14,

42156000,

"Rua torta",

5,

5

);

INSERT INTO

transacao\_pagamento(id\_transacao, saldo\_conta, valor\_transacao, metodo\_pagamento, id\_cliente)

VALUES

(1, 0, 10, "pix", 2),

(2, 0, 20, "cartao-debito", 1),

(3, 0, 30, "pix", 4),

(4, 0, 20, "cartao-credito", 3),

(5, 0, 20, "pix", 5);

INSERT INTO

locacao\_estacao (id\_locacao, id\_estacao)

VALUES

(1, 2),

(2, 3),

(5, 2),

(3, 4),

(2, 5);

**Construção automatizada.**

***/\****

***Atualize automaticamente o status da bicicleta quando o aluguel começar.***

***\*/***

DELIMITER //

CREATE TRIGGER triggerLocacao

AFTER INSERT ON locacao FOR EACH ROW

BEGIN

-- Atualiza o status da bicicleta para "indisponivel" quando a locação começa

IF (NEW.estacao\_final IS NULL) THEN

UPDATE

bicicleta\_info

SET status\_bicicleta = 'indisponivel'

WHERE id\_bicicleta = NEW.id\_bicicleta;

END IF;

***/\****

***Atualize automaticamente o status da bicicleta quando o aluguel começar.***

***\*/***

END; //

DELIMITER ;

DELIMITER //

CREATE TRIGGER triggerLocacao2

AFTER UPDATE ON locacao FOR EACH ROW

BEGIN

-- Atualiza o status da bicicleta para "disponivel" quando a locação é concluída

IF (NEW.estacao\_final IS NOT NULL) THEN

UPDATE bicicleta\_info

SET status\_bicicleta = 'disponivel'

WHERE id\_bicicleta = NEW.id\_bicicleta;

END IF;

END;

//

DELIMITER ;

***/\****

***@param ID do usuário, passado pelo sistema***

***@param id da bicicleta, passado pelo sistema***

***@param ID da estação inicial, passado pelo sistema***

***No momento da locação é realizada uma verificação de disponibilidade e as informações de locação são inseridas no formulário de locação e o estoque de bicicletas da estação é atualizado.***

***NOTA: Término, data final e total são indefinidos, mas serão atualizados no retorno***

***\*/***

DELIMITER //

CREATE PROCEDURE alugar (

IN id\_cliente INT,

IN id\_bicicleta INT,

IN estacao\_inicial INT

)

BEGIN

***-- verificar disponbilidade***

IF (SELECT COUNT(\*) FROM bicicleta\_info WHERE status\_bicicleta = 'disponivel' AND id\_bicicleta = id\_bicicleta AND id\_estacao = estacao\_inicial) > 0 THEN

INSERT INTO locacao (id\_cliente, id\_bicicleta, estacao\_inicial, data\_inicio, status\_locacao)

VALUES (id\_cliente, id\_bicicleta, estacao\_inicial, NOW(), 'em andamento');

UPDATE bicicleta\_info

SET status\_bicicleta = 'indisponivel'

WHERE id\_bicicleta = id\_bicicleta;

END IF;

END;

//

DELIMITER ;

***/\****

***@param ID do usuário, passado pelo sistema***

***@param id da bicicleta, passado pelo sistema***

***@param ID da estação inicial, passado pelo sistema***

***Ao alugar, é realizada uma verificação de disponibilidade e o preço do aluguel é atualizado na tabela de aluguel***

***e inventário atualizado de bicicletas na nova estação.***

***\*/***

DELIMITER //

CREATE PROCEDURE devolver (

IN id\_cliente INT,

IN id\_bicicleta INT,

IN estacao\_final INT

)

BEGIN

DECLARE total DECIMAL(10, 2);

DECLARE data\_inicio DATETIME;

DECLARE data\_final DATETIME;

SET data\_inicio = (SELECT data\_inicio FROM locacao WHERE id\_bicicleta = id\_bicicleta AND id\_cliente = id\_cliente AND status\_locacao = 'em andamento');

SET data\_final = NOW();

SET total = (SELECT preco\_hora FROM bicicleta\_info WHERE id\_bicicleta = id\_bicicleta) \* (HOUR(TIMEDIFF(data\_final, data\_inicio)) + MINUTE(TIMEDIFF(data\_final, data\_inicio)) / 60);

UPDATE locacao

SET estacao\_final = estacao\_final,

data\_fim = data\_final,

valor\_total = total,

status\_locacao = 'concluida'

WHERE id\_cliente = id\_cliente AND id\_bicicleta = id\_bicicleta AND status\_locacao = 'em andamento';

UPDATE bicicleta\_info

SET status\_bicicleta = 'disponivel', id\_estacao = estacao\_final

WHERE id\_bicicleta = id\_bicicleta;

END;

//

DELIMITER ;

***/\****

***@param ID do usuário***

***@param metedo para depósito***

***@param valor do depósito***

***Atualize o novo saldo na tabela de usuários e o saldo do histórico de depósitos na tabela de pagamentos***

\*/

DELIMITER //

CREATE PROCEDURE deposito(

IN id\_cliente INT,

IN metodo VARCHAR(20),

IN valor DECIMAL(10, 2)

)

BEGIN

UPDATE cliente SET saldo\_conta = saldo\_conta + valor WHERE idcliente = id\_cliente;

INSERT INTO transacao\_pagamento (saldo\_conta, valor\_transacao, metodo\_pagamento, id\_cliente)

VALUES ((SELECT saldo\_conta FROM cliente WHERE idcliente = id\_cliente), valor, metodo, id\_cliente);

END;

//

DELIMITER ;

***/\****

***view que facilita a consulta das estações, ja que a tabela estação si são duas colunas! \*/***

CREATE VIEW estacoes AS

SELECT bairro\_local, numero\_local, cep\_local, rua\_local FROM endereco\_local;

***/\****

***@param estacao id***

***@return número de bicicletas***

***Consulta de estoque por temporada***

***\*/***

CREATE FUNCTION estoquePorEstacao (idestacao\_ INT) RETURNS INT DETERMINISTIC

BEGIN

RETURN (SELECT COUNT(\*) FROM bicicleta\_info WHERE id\_estacao = idestacao\_ AND status\_bicicleta = 'disponivel');

END;

//

DELIMITER ;

**Execuções.**

-- alugar e devolver

call alugar(4, 2, 1); -- idusuario, idbicicleta, estacaoinicial,

call alugar(2, 3, 2);

call alugar(3, 4, 1);

call devolver(4, 2, 4);

call devolver(2, 3, 1);

-- fazer um deposito

call deposito(1, "PIX", 100); -- idusuario, metodo, valor

call deposito(1, "PIX", 100);

call deposito(2, "PIX", 10000);

-- consultar estoque

SELECT estoquePorEstacao(2);

-- endereço de cadaestação

select \* from estacoes;

18