MAPA – Material de Avaliação Prática da Aprendizagem

Acadêmico: Robson Cruz Santos	**R.A. 22117001-5**
Curso: Engenharia de Software	
Disciplina: Banco de Dados II	
Valor da atividade: 3,5	**Prazo: 30/06/2024 23:59**

Você foi contratado para criar um sistema de gerenciamento de hospedagens hoteleiras. O sistema deve ser capaz de armazenar informações sobre hotéis, quartos, clientes e hospedagens. Os clientes podem se hospedar em quartos de hotéis diferentes, e é importante manter um registro das reservas/hospedagens.

O sistema deve conter as seguintes tabelas:

Tabela "Hotel":

hotel_id (Chave primária, INT): Identificador único do hotel.
nome (VARCHAR, não nulo): Nome do hotel.
cidade (VARCHAR, não nulo): Cidade onde o hotel está localizado.
uf (VARCHAR, não nulo): Estado onde o hotel está localizado, com dois dígitos.
classificacao (INT, não nulo): Classificação do hotel em estrelas (1 até 5).

Tabela "Quarto":

quarto_id (Chave primária, INT): Identificador único do quarto.
hotel_id (Chave estrangeira não nula para "Hotel"): Identificador do hotel ao qual o quarto pertence.
número (INT, não nulo): Número do quarto.
tipo (VARCHAR, não nulo): Tipo de quarto (por exemplo, "Standard", "Deluxe", "Suíte").
preco_diaria (DECIMAL, não nulo): Preço da diária do quarto.

Tabela "Cliente":

cliente_id (Chave primária, INT): Identificador único do cliente.
nome (VARCHAR, não nulo): Nome do cliente.
email (VARCHAR, não nulo e único): Endereço de e-mail do cliente.
telefone (VARCHAR, não nulo): Número de telefone do cliente.
cpf (VARCHAR, não nulo e único): Número de CPF do cliente.

Tabela "Hospedagem":

hospedagem_id (Chave primária, INT): Identificador único da hospedagem.
cliente_id (Chave estrangeira não nula para "Cliente"): Identificador do cliente que fez a reserva.
quarto_id (Chave estrangeira não nula para "Quarto"): Identificador do quarto reservado.
dt_checkin (DATE): Data de check-in da hospedagem (não nula).
dt_checkout (DATE): Data de check-out da hospedagem (não nula).
Valor_total_hosp(FLOAT, não nulo): Custo total da hospedagem, calculado quando a hospedagem é finalizada.
status_hosp (VARCHAR, não nulo): status_hosp da hospedagem, podendo receber os seguintes valores: “reserva”, reservado pelo cliente; “finalizada”, hospedagem concluida; “hospedado”, o cliente está atualmente hospedado no hotel; “cancelada”, a hospedagem (reserva) foi cancelada.

Para esta atividade mapa você deve criar código SQL, usando MySQL ou PostgreSQL:

1. Crie o esquema de banco de dados para o sistema de gerenciamento e hospedagens hoteleiras nomeado “hospedar_db”.

2. Crie as tabelas "Hotel", "Quarto", "Cliente" e "Hospedagem" com as colunas especificadas anteriormente.

3. Insira dados artificiais nas tabelas "Hotel" (2 hotéis), "Quarto"(5 para cada hotel), "Cliente"(3 clientes) e "Hospedagem" (20 hospedagens, 5 para cada um dos “Status_hosp”) para simular hotéis, quartos, clientes e hospedagens.

4. Escreva as seguintes consultas e comandos SQL:

a. Listar todos os hotéis e seus respectivos quartos, apresentando os seguintes campos: para hotel, nome e cidade; para quarto, tipo e preco_diaria;
b. Listar todos os clientes que já realizaram hospedagens (status_hosp igual á “finalizada”), e os respectivos quartos e hotéis;
c. Mostrar o histórico de hospedagens em ordem cronológica de um determinado cliente;
d. Apresentar o cliente com maior número de hospedagens (não importando o tempo em que ficou hospedado);
e. Apresentar clientes que tiveram hospedagem “cancelada”, os respectivos quartos e hotéis.
f. Calcular a receita de todos os hotéis (hospedagem com status_hosp igual a “finalizada”), ordenado de forma decrescente;
g. Listar todos os clientes que já fizeram uma reserva em um hotel específico;
h. Listar o quanto cada cliente que gastou em hospedagens (status_hosp igual a “finalizada”), em ordem decrescente por valor gasto.
i. Listar todos os quartos que ainda não receberam hóspedes.
j. Apresentar a média de preços de diárias em todos os hotéis, por tipos de quarto.
l. Criar a coluna checkin_realizado do tipo booleano na tabela Hospedagem (via código). E atribuir verdadeiro para as Hospedagens com status_hosp “finalizada” e “hospedado”, e como falso para Hospedagens com status_hosp “reserva” e “cancelada”.
m. Mudar o nome da coluna “classificacao” da tabela Hotel para “ratting” (via código).

5. Efetue a criação dos seguintes procedimentos usando PL/MySQL:

a. Criar uma procedure chamada "RegistrarCheckIn" que aceita hospedagem_id e data_checkin como parâmetros. A procedure deve atualizar a data de check-in na tabela "Hospedagem" e mudar o status_hosp para "hospedado".

b. Criar uma procedure chamada "CalcularTotalHospedagem" que aceita hospedagem_id como parâmetro. A procedure deve calcular o valor total da hospedagem com base na diferença de dias entre check-in e check-out e o preço da diária do quarto reservado. O valor deve ser atualizado na coluna valor_total_hosp.
c. Criar uma procedure chamada "RegistrarCheckout" que aceita hospedagem_id e data_checkout como parâmetros. A procedure deve atualizar a data de check-out na tabela "Hospedagem" e mudar o status_hosp para "finalizada".

6. Efetue a criação das seguintes funções utilizando PL/MySQL:

a. Criar uma function chamada "TotalHospedagensHotel" que aceita hotel_id como parâmetro. A função deve retornar o número total de hospedagens realizadas em um determinado hotel.
b. Criar uma function chamada "ValorMedioDiariasHotel" que aceita hotel_id como parâmetro. A função deve calcular e retornar o valor médio das diárias dos quartos deste hotel.
c. Criar uma function chamada "VerificarDisponibilidadeQuarto" que aceita quarto_id e data como parâmetros. A função deve retornar um valor booleano indicando se o quarto está disponível ou não para reserva na data especificada.

7. Efetue a criação das seguintes triggers utilizando PL/MySQL:

a. Criar um trigger chamado "AntesDeInserirHospedagem" que é acionado antes de uma inserção na tabela "Hospedagem". O trigger deve verificar se o quarto está disponível na data de check-in. Se não estiver, a inserção deve ser cancelada.

b. Cria um trigger chamado "AposDeletarCliente" que é acionado após a exclusão de um cliente na tabela "Cliente". O trigger deve registrar a exclusão em uma tabela de log.

Observações:
1. Apresentar os códigos SQL utilizados para a resolução de todas as questões em ordem cronológica.

2. Todas os items dever ser feitos utilizando instruções SQL, sem a modificação dos dados diretamente nas tabelas.

Resposta aqui!!

1. Crie o esquema de banco de dados para o sistema de gerenciamento e hospedagens hoteleiras nomeado “hospedar_db”.

```
CREATE SCHEMA hospedar_db;
```

2. Crie as tabelas "Hotel", "Quarto", "Cliente" e "Hospedagem" com as colunas especificadas anteriormente.

```
CREATE TABLE Hotel (
        hotel_id  INT AUTO_INCREMENT,
   nome VARCHAR(100) NOT NULL,
   cidade VARCHAR(100) NOT NULL,
   uf VARCHAR(2) NOT NULL,
   classificacao INT NOT NULL COMMENT 'Classificação do hotel em estrelas (1 até 5).',

   CONSTRAINT hotel_id_pk PRIMARY KEY(hotel_id )
);

CREATE TABLE Quarto (
        quarto_id INT AUTO_INCREMENT,
   hotel_id INT NOT NULL,
   numero INT NOT NULL COMMENT 'Número do quarto.',
   tipo VARCHAR(50) NOT NULL COMMENT 'Tipo de quarto: Standard, Deluxe, Suíte',
   preco_diaria  DECIMAL(10, 2) NOT NULL,
```

```
    CONSTRAINT quarto_id_pk PRIMARY KEY(quarto_id),
    CONSTRAINT hotel_id_fk FOREIGN KEY(hotel_id) REFERENCES Hotel(hotel_id),
    CONSTRAINT tipo_ck CHECK(tipo IN('Standard', 'Deluxe', 'Suíte')),
    CONSTRAINT preco_diaria_ck CHECK(preco_diaria > 0)
);

CREATE TABLE Cliente (
        cliente_id INT AUTO_INCREMENT,
    nome VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL,
    telefone VARCHAR(20) NOT NULL,
    cpf VARCHAR(11) NOT NULL,

    CONSTRAINT cliente_id_pk PRIMARY KEY(cliente_id),
    CONSTRAINT cpf_un UNIQUE(cpf),
    CONSTRAINT cpf_ck CHECK(CHAR_LENGTH(cpf) = 11)
);

CREATE TABLE Hospedagem (
        hospedagem_id INT AUTO_INCREMENT,
    cliente_id INT NOT NULL,
    quarto_id INT NOT NULL,
    dt_checkin DATE NOT NULL,
    dt_checkout DATE NOT NULL,
    Valor_total_hosp FLOAT NOT NULL,
    status_hosp VARCHAR(10) NOT NULL,

    CONSTRAINT hospedagem_id_pk PRIMARY KEY(hospedagem_id ),
    CONSTRAINT cliente_id_fk FOREIGN KEY(cliente_id) REFERENCES
Cliente(cliente_id)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
```

```
    CONSTRAINT quarto_id_fk FOREIGN KEY(quarto_id) REFERENCES Quarto(quarto_id),
    CONSTRAINT Valor_total_hosp_ck CHECK(Valor_total_hosp > 0),
    CONSTRAINT status_hosp_ck CHECK(status_hosp IN('reserva', 'finalizada', 'hospedado', 'cancelada'))
);
-- Adiciona Comentário ao atributo 'status_hosp' da tabela 'Hospedagem'
ALTER TABLE Hospedagem
        MODIFY COLUMN status_hosp VARCHAR(50) NOT NULL
                COMMENT 'reserva = Reservado pelo Cliente;
                                finalizada = Hospedagem Concluida;
                                hospedado = o Cliente está Hospedado;
                                cancelada = A Reserva foi Cancelada.';
```

3. Insira dados artificiais nas tabelas "Hotel" (2 hotéis), "Quarto"(5 para cada hotel), "Cliente"(3 clientes) e "Hospedagem" (20 hospedagens, 5 para cada um dos “Status_hosp”) para simular hotéis, quartos, clientes e hospedagens.

```
USE hospedar_db;

INSERT INTO Hotel (nome, cidade, uf, classificacao)
VALUES
        ('Hotel Beira Mar', 'Natal', 'RN', 5),
    ('Hotel Mirante do Lago', 'Natal', 'RN', 4);

INSERT INTO Quarto (hotel_id, numero, tipo, preco_diaria)
VALUES
        (1, 100, 'Standard', 178.50),
        (1, 101, 'Deluxe', 278.00),
    (1, 102, 'Standard', 178.50),
    (1, 103, 'Suíte', 290.50),
    (1, 104, 'Suíte', 290.50),
    (2, 10, 'Standard', 290.50),
```

```
    (2, 12, 'Standard', 290.50),
    (2, 14, 'Standard', 290.50),
    (2, 16, 'Standard', 290.50),
    (2, 18, 'Standard', 290.50);

INSERT INTO Cliente (nome, email, telefone, cpf)
VALUES
        ('Antonio Zimermax', 'antzim23@hostname.com', '31 98487-3251', '32174147098'),
    ('Ana Maria Oliveria', 'olivasana@hostname.com', '11 99885-7925', '21741470981'),
    ('Jonas Baleeiro', 'jonasbal@hostname.com', '31 98875-2122', '67784220111');

INSERT INTO Hospedagem (cliente_id, quarto_id, dt_checkin, dt_checkout,
Valor_total_hosp, status_hosp)
VALUES
        (1, 1, '2023-11-20', '2023-11-25', 892.50, 'finalizada'),
    (1, 2, '2023-12-01', '2023-12-02', 278.00, 'finalizada'),
    (1, 1, '2024-01-20', '2024-01-22', 357.00, 'finalizada'),
    (2, 1, '2023-11-25', '2023-11-26', 178.50, 'finalizada'),
    (2, 6, '2023-12-20', '2023-12-22', 581.00, 'finalizada'),
    (2, 1, '2023-12-22', '2023-12-23', 178.50, 'cancelada'),
    (2, 3, '2024-01-05', '2024-01-07', 357.00, 'cancelada'),
    (1, 1, '2024-01-05', '2024-01-07', 357.00, 'cancelada'),
    (3, 7, '2024-01-05', '2024-01-07', 581.00, 'cancelada'),
    (3, 8, '2024-02-20', '2024-02-21', 290.50, 'cancelada'),
    (2, 10, '2024-05-20', '2024-05-30', 2905.00, 'hospedado'),
    (1, 3, '2024-05-20', '2024-05-30', 892.50, 'hospedado'),
    (1, 4, '2024-05-20', '2024-05-30', 892.50, 'hospedado'),
    (3, 6, '2024-05-20', '2024-05-30', 892.50, 'hospedado'),
    (3, 7, '2024-05-20', '2024-05-30', 892.50, 'hospedado'),
    (1, 1, '2024-06-01', '2024-06-05', 892.50, 'reserva'),
    (3, 2, '2024-07-20', '2024-07-25', 556.00, 'reserva'),
    (3, 3, '2024-07-20', '2024-07-25', 892.50, 'reserva'),
```

```
(2, 8, '2024-07-20', '2024-07-25', 1452.50, 'reserva'),
(2, 7, '2024-07-20', '2024-07-25', 1452.5, 'reserva');
```

4. Escreva as seguintes consultas e comandos SQL:

a. Listar todos os hotéis e seus respectivos quartos, apresentando os seguintes campos: para hotel, nome e cidade; para quarto, tipo e preco_diaria;

```
SELECT DISTINCT
        nome AS Nome,
    cidade AS Cidade,
    tipo AS Tipo,
    preco_diaria AS 'Preço Diária'
FROM Hotel
INNER JOIN Quarto ON Hotel.hotel_id = Quarto.hotel_id;
```

b. Listar todos os clientes que já realizaram hospedagens (status_hosp igual á “finalizada”), e os respectivos quartos e hotéis;

```
SELECT
        Cliente.nome AS 'Nome do Cliente',
    Quarto.numero AS 'Número do Quarto',
    Hotel.nome AS Hotel
FROM Cliente
INNER JOIN Hospedagem ON Cliente.cliente_id = Hospedagem.cliente_id
INNER JOIN Quarto ON Hospedagem.quarto_id = Quarto.quarto_id
INNER JOIN Hotel ON Quarto.hotel_id = Hotel.hotel_id
WHERE status_hosp = 'finalizada';
```

c. Mostrar o histórico de hospedagens em ordem cronológica de um determinado cliente;

```
SELECT
        nome AS Cliente,
   dt_checkin AS 'Check In',
   dt_checkout AS 'Check Out'
FROM Cliente
INNER JOIN Hospedagem ON Cliente.cliente_id = Hospedagem.cliente_id
WHERE nome = 'Antonio Zimermax'
ORDER BY dt_checkin;
```

d. Apresentar o cliente com maior número de hospedagens (não importando o tempo em que ficou hospedado);

```
SELECT
        Cliente.nome AS Cliente,
   COUNT(*) AS Hospedagens
FROM Hospedagem
INNER JOIN Cliente ON Hospedagem.cliente_id = Cliente.cliente_id
WHERE Hospedagem.status_hosp = 'finalizada'
GROUP BY Cliente.nome
LIMIT 1;
```

e. Apresentar clientes que tiveram hospedagem “cancelada”, os respectivos quartos e hotéis.

```
SELECT
        Cliente.nome AS Cliente,
        Quarto.numero AS 'Número Quarto',
        Hotel.nome AS Hotel
FROM Cliente
INNER JOIN Hospedagem ON Cliente.cliente_id = Hospedagem.cliente_id
INNER JOIN Quarto ON Hospedagem.quarto_id = Quarto.quarto_id
INNER JOIN Hotel ON Quarto.hotel_id = Hotel.hotel_id
WHERE status_hosp = 'cancelada';
```

f. Calcular a receita de todos os hotéis (hospedagem com status_hosp igual a “finalizada”), ordenado de forma decrescente;

```
SELECT
        nome AS Hotel,
   SUM(Valor_total_hosp) AS Receita
FROM Hotel
INNER JOIN Quarto ON Hotel.hotel_id = Quarto.hotel_id
INNER JOIN Hospedagem ON Quarto.quarto_id = Hospedagem.quarto_id
WHERE status_hosp = 'finalizada'
GROUP BY nome;
```

g. Listar todos os clientes que já fizeram uma reserva em um hotel específico;

```
SELECT
        Cliente.nome AS Cliente,
   Hotel.nome AS Hotel,
   status_hosp AS 'Status Hospedagem'
FROM Cliente
INNER JOIN Hospedagem ON Cliente.cliente_id = Hospedagem.cliente_id
INNER JOIN Quarto ON Hospedagem.quarto_id = Quarto.quarto_id
INNER JOIN Hotel ON Quarto.hotel_id = Hotel.hotel_id
WHERE Hotel.nome = 'Hotel Mirante do Lago';
```

h. Listar o quanto cada cliente que gastou em hospedagens (status_hosp igual a “finalizada”), em ordem decrescente por valor gasto.

```
SELECT
        nome AS Cliente,
   SUM(Valor_total_hosp) AS 'Valor Hospedagem'
FROM Cliente
INNER JOIN Hospedagem ON Cliente.cliente_id = Hospedagem.cliente_id
WHERE status_hosp = 'finalizada'
GROUP BY nome
```

```
ORDER BY SUM(Valor_total_hosp) DESC;
```

i. Listar todos os quartos que ainda não receberam hóspedes.

```
SELECT
        Quarto.numero AS Quarto,
   Hotel.nome AS Hotel
FROM Quarto
INNER JOIN Hotel ON Quarto.hotel_id = Hotel.hotel_id
WHERE quarto_id NOT IN (SELECT quarto_id FROM Hospedagem);
```

j. Apresentar a média de preços de diárias em todos os hotéis, por tipos de quarto.

```
SELECT
        nome AS Hotel,
   tipo AS 'Tipo Quarto',
   ROUND(AVG(preco_diaria), 2) AS 'Valor Médio Diária'
FROM Hotel
INNER JOIN Quarto ON Hotel.hotel_id = Quarto.hotel_id
GROUP BY nome, tipo;
```

l. Criar a coluna checkin_realizado do tipo booleano na tabela Hospedagem (via código). E atribuir verdadeiro para as Hospedagens com status_hosp “finalizada” e “hospedado”, e como falso para Hospedagens com status_hosp “reserva” e “cancelada”.

```
USE hospedar_db;

ALTER TABLE Hospedagem ADD COLUMN checkin_realizado BOOLEAN;



UPDATE Hospedagem
SET checkin_realizado = TRUE
WHERE status_hosp IN ('finalizada', 'hospedado') AND hospedagem_id IS NOT NULL AND hospedagem_id > 0;

UPDATE Hospedagem
SET checkin_realizado = FALSE
WHERE status_hosp IN ('reserva', 'cancelada') AND hospedagem_id IS NOT NULL AND hospedagem_id > 0;
```

m. Mudar o nome da coluna “classificacao” da tabela Hotel para “ratting” (via código).

```
USE hospedar_db;

ALTER TABLE Hotel
RENAME COLUMN classificacao TO ratting;
```

5. Efetue a criação dos seguintes procedimentos usando PL/MySQL:

a. Criar uma procedure chamada "RegistrarCheckIn" que aceita hospedagem_id e data_checkin como parâmetros. A procedure deve atualizar a data de check-in na tabela "Hospedagem" e mudar o status_hosp para "hospedado".

```
DELIMITER $$
CREATE PROCEDURE RegistrarCheckIn(IN hosp_id INT, IN data_checkin DATE)
BEGIN
        DECLARE conta_hospedagem_id INT;
  DECLARE status_hosp_novo VARCHAR(50);

  -- Confere se a hospedagem existe
  SELECT COUNT(*) INTO conta_hospedagem_id
  FROM Hospedagem WHERE hospedagem_id = hosp_id;

  IF conta_hospedagem_id = 1 THEN
    -- Desativa temporariamente o modo de atualização segura, caso esteja habilitado
    SET @@SESSION.sql_safe_updates = 0;

    -- Atualiza a tabela com a nova da ta de checkin
    BEGIN
                    UPDATE Hospedagem
      SET dt_checkin = data_checkin, status_hosp = 'hospedado'
      WHERE hospedagem_id = hosp_id;
    END;

        -- Ativa novamente o modo de atualização segura
    SET @@SESSION.sql_safe_updates = 1;

      ELSE
        SELECT CONCAT('Id de Hospedagem ', hosp_id, ' não encontrado!') AS
Error_Message;
  END IF;
END $$
DELIMITER ;;
```

b. Criar uma procedure chamada "CalcularTotalHospedagem" que aceita hospedagem_id como parâmetro. A procedure deve calcular o valor total da hospedagem com base na diferença de dias entre check-in e check-out e o preço da diária do quarto reservado. O valor deve ser atualizado na coluna valor_total_hosp.

```
DELIMITER $$
CREATE PROCEDURE CalcularTotalHospedagem(IN hosp_id INT)
BEGIN
   -- Total de dias hospedado
   DECLARE diarias INT;
   -- Valor da diária
   DECLARE custo_diaria FLOAT;
   -- Valor total a ser pago após check out
   DECLARE valor_total_hosp_final DECIMAL(10, 2);
   -- Data de entrada definitiva
   DECLARE dt_checkin_fato DATE;
   -- Data de saída definitiva
   DECLARE dt_checkout_fato DATE;
   -- Contador para id de hospedagem
   DECLARE conta_id_hosp INT;

   -- Confere se o id de hospedagem existe na tabela Hospedagem
   SELECT COUNT(*) INTO conta_id_hosp
   FROM Hospedagem WHERE hospedagem_id = hosp_id;

   IF conta_id_hosp = 1 THEN
      -- Desativa temporariamente o modo de atualização segura, caso esteja habilitado
      SET @@SESSION.sql_safe_updates = 0;

      SELECT
         preco_diaria,
         dt_checkin,
         dt_checkout
```

```
        INTO
            custo_diaria,
            dt_checkin_fato,
            dt_checkout_fato
        FROM Hospedagem
        INNER JOIN Quarto ON Hospedagem.quarto_id = Quarto.quarto_id
        WHERE Hospedagem.hospedagem_id = hosp_id;

        -- Cálculo da conta a ser paga pelo cliente
        SET diarias = DATEDIFF(dt_checkout_fato, dt_checkin_fato);
        SET valor_total_hosp_final = diarias * custo_diaria;

        -- Atualiza o valor total de hospedagem na tabela Hospedagem
        UPDATE Hospedagem
        SET valor_total_hosp = valor_total_hosp_final
        WHERE hospedagem_id = hosp_id;

        -- Ativa novamente o modo de atualização segura
        SET @@SESSION.sql_safe_updates = 1;

    ELSE
        SELECT CONCAT('ID de hospedagem ', hosp_id, ' não encontrado!') AS
Error_Message;
    END IF;

END $$
DELIMITER ;
```

c. Criar uma procedure chamada "RegistrarCheckout" que aceita hospedagem_id e data_checkout como parâmetros. A procedure deve atualizar a data de check-out na tabela "Hospedagem" e mudar o status_hosp para "finalizada".

```
DELIMITER $$
CREATE PROCEDURE RegistrarCheckout(IN hosp_id INT, IN data_checkout DATE)
BEGIN
        DECLARE conta_hospedagem_id INT;

  -- Conferir se a hospedagem é válida
  SELECT COUNT(*) INTO conta_hospedagem_id
  FROM Hospedagem
  WHERE hospedagem_id = hosp_id;

  IF conta_hospedagem_id = 1 THEN
    -- Desativa temporariamente o modo de atualização segura, caso esteja habilitado
    SET @@SESSION.sql_safe_updates = 0;

    -- Atualiza a tabela Hospedagem com a data de check out
    BEGIN
                    UPDATE Hospedagem
      SET dt_checkout = data_checkout, status_hosp = 'finalizada'
      WHERE hospedagem_id = hosp_id;
    END;

    -- Ativa novamente o modo de atualização segura
    SET @@SESSION.sql_safe_updates = 1;
  ELSE
            SELECT CONCAT('Id de Hospedagem ', hosp_id, ' não encontrado!') AS
Error_Message;
  END IF;
END $$
DELIMITER ;
```

6. Efetue a criação das seguintes funções utilizando PL/MySQL:

a. Criar uma function chamada "TotalHospedagensHotel" que aceita hotel_id como parâmetro. A função deve retornar o número total de hospedagens realizadas em um determinado hotel.

```
DELIMITER $$
CREATE FUNCTION TotalHospedagensHotel(id INT)
RETURNS INT DETERMINISTIC

BEGIN
        DECLARE num_hospedagem INT;
        SELECT COUNT(*) INTO num_hospedagem
        FROM hospedar_db.Hospedagem
   INNER JOIN hospedar_db.Quarto ON hospedar_db.Hospedagem.quarto_id = hospedar_db.Quarto.quarto_id
        WHERE hotel_id = id AND status_hosp = 'finalizada';

        RETURN num_hospedagem;
END $$

DELIMITER ;
```

b. Criar uma function chamada "ValorMedioDiariasHotel" que aceita hotel_id como parâmetro. A função deve calcular e retornar o valor médio das diárias dos quartos deste hotel.

```
DELIMITER $$
CREATE FUNCTION ValorMedioDiariasHotel(id INT)
RETURNS DECIMAL(10, 2) DETERMINISTIC

BEGIN
        DECLARE valor_medio_diaria DECIMAL(10, 2);

   SELECT AVG(Quarto.preco_diaria) INTO valor_medio_diaria
        FROM hospedar_db.Hospedagem
```

```
   INNER JOIN hospedar_db.Quarto ON hospedar_db.Hospedagem.quarto_id =
hospedar_db.Quarto.quarto_id
        WHERE hotel_id = id;


        RETURN valor_medio_diaria;
END $$
DELIMITER ;
```

c. Criar uma function chamada "VerificarDisponibilidadeQuarto" que aceita quarto_id e data como parâmetros. A função deve retornar um valor booleano indicando se o quarto está disponível ou não para reserva na data especificada.

```
DELIMITER $$
CREATE FUNCTION VerificarDisponibilidadeQuarto(id INT, data_busca DATE)
RETURNS BOOLEAN DETERMINISTIC

COMMENT 'A VerificarDisponibilidadeQuarto deve receber a data no formato YYYY-
MM-DD'

BEGIN
        DECLARE busca BOOLEAN;
        DECLARE ocupado INT;

   SELECT COUNT(*) INTO ocupado
        FROM hospedar_db.Hospedagem
   INNER JOIN hospedar_db.Quarto ON hospedar_db.Hospedagem.quarto_id =
hospedar_db.Quarto.quarto_id
        WHERE Quarto.quarto_id = id
                AND status_hosp NOT IN ( 'finalizada', 'cancelada')
                        AND data_busca NOT BETWEEN dt_checkin AND
dt_checkout;

   IF ocupado = 0 THEN
                SET busca = TRUE;
        ELSE
                SET busca = FALSE;
        END IF;

   RETURN busca;

END $$
DELIMITER ;
```

7. Efetue a criação das seguintes triggers utilizando PL/MySQL:

a. Criar um trigger chamado "AntesDeInserirHospedagem" que é acionado antes de uma inserção na tabela "Hospedagem". O trigger deve verificar se o quarto está disponível na data de check-in. Se não estiver, a inserção deve ser cancelada.

```
DELIMITER $$
CREATE TRIGGER AntesDeInserirHospedagem
BEFORE INSERT ON Hospedagem
FOR EACH ROW
BEGIN
        IF NOT VerificarDisponibilidadeQuarto(NEW.quarto_id, NEW.dt_checkin) THEN
                SIGNAL SQLSTATE '45000'
                        SET MESSAGE_TEXT = 'O quarto não está disponível na data de
check-in.';
        END IF;
END $$
DELIMITER;
```

b. Cria um trigger chamado "AposDeletarCliente" que é acionado após a exclusão de um cliente na tabela "Cliente". O trigger deve registrar a exclusão em uma tabela de log.

```
-- Criar a tabela de log
CREATE TABLE LogExclusaoCliente (
    log_id INT AUTO_INCREMENT,
    cliente_id INT,
    nome VARCHAR(100),
    cpf VARCHAR(11),
    data_hora DATETIME,
    usuario VARCHAR(100),

    CONSTRAINT log_id_pk PRIMARY KEY(log_id)
);

-- Criar o trigger AposDeletarCliente
USE hospedar_db;

DELIMITER $$
CREATE TRIGGER AposDeletarCliente
```

```
AFTER DELETE ON Cliente
FOR EACH ROW
BEGIN
        INSERT INTO LogExclusaoCliente (cliente_id, nome, cpf, data_hora)
   VALUES (OLD.cliente_id, OLD.nome, OLD.cpf, NOW(), SYSTEM_USER());
END $$
DELIMITER ;
```