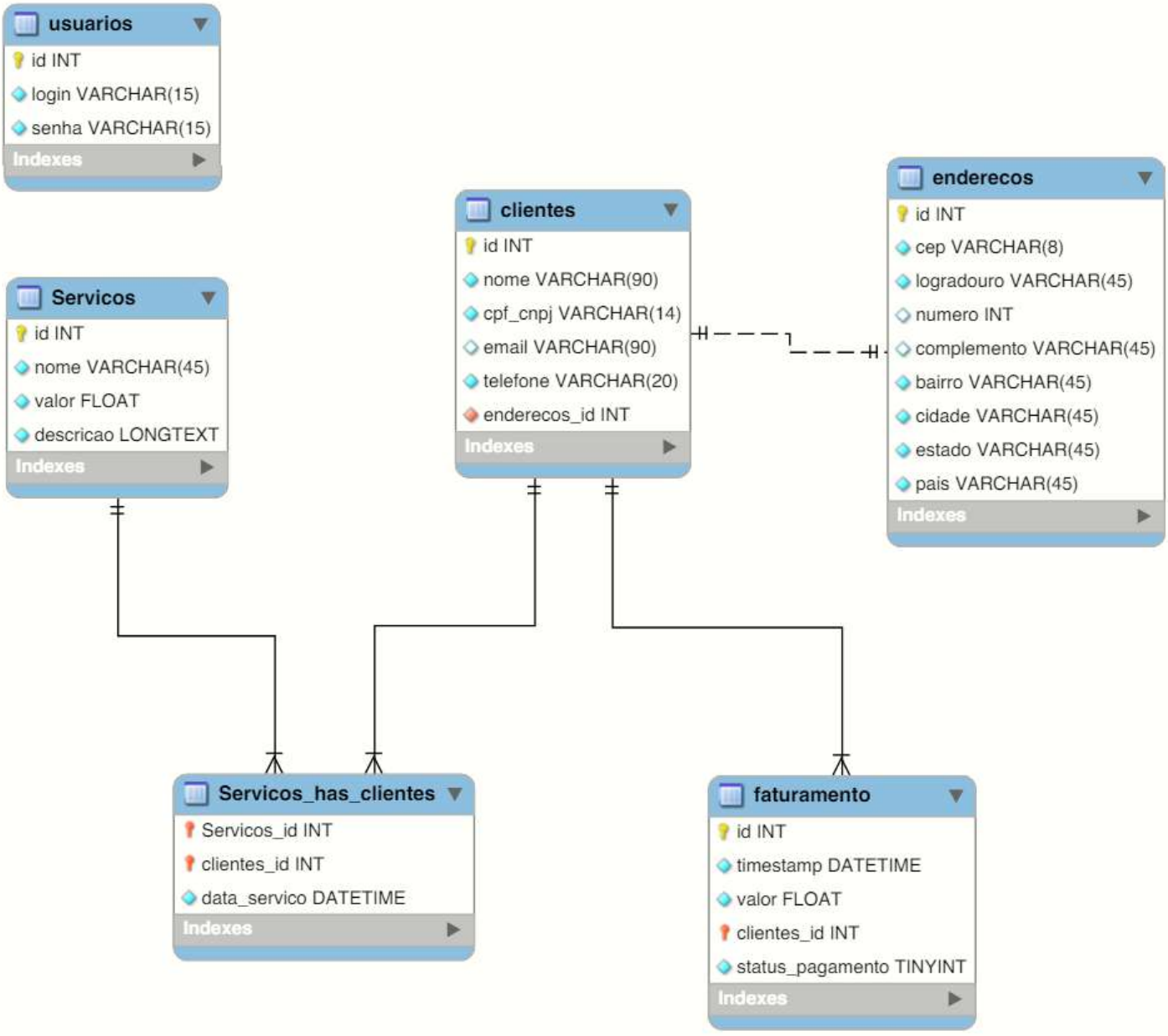
usuarios
id INT
login VARCHAR(15)
senha VARCHAR(15)
Indexes
clientes
id INT
nome VARCHAR(90)
cpf_cnpj VARCHAR(14)
email VARCHAR(90)
telefone VARCHAR(20)
enderecos_id INT
Indexes
enderecos
id INT
cep VARCHAR(8)
logradouro VARCHAR(45)
numero INT
complemento VARCHAR(45)
bairro VARCHAR(45)
cidade VARCHAR(45)
estado VARCHAR(45)
pais VARCHAR(45)
Indexes
Servicos
id INT
nome VARCHAR(45)
valor FLOAT
descricao LONGTEXT
Indexes
Servicos_has_clientes
Servicos_id INT
clientes_id INT
data_servico DATETIME
Indexes
faturamento
id INT
timestamp DATETIME
valor FLOAT
clientes_id INT
status_pagamento TINYINT
Indexes

```
create database jeting_db;
use jeting_db;

CREATE TABLE IF NOT EXISTS usuarios (
    id BIGINT PRIMARY KEY NOT NULL AUTO_INCREMENT,
    login VARCHAR(15) NOT NULL,
    senha VARCHAR(15) NOT NULL
);

CREATE TABLE IF NOT EXISTS enderecos (
    id BIGINT PRIMARY KEY NOT NULL AUTO_INCREMENT,
    cep VARCHAR(8) NOT NULL,
    logradouro VARCHAR(45) NOT NULL,
    numero INT,
    complemento VARCHAR(45),
    bairro VARCHAR(45) NOT NULL,
    cidade VARCHAR(45) NOT NULL,
    estado VARCHAR(45) NOT NULL,
    pais VARCHAR(45) NOT NULL
);

CREATE TABLE IF NOT EXISTS clientes (
    id BIGINT PRIMARY KEY NOT NULL AUTO_INCREMENT,
    nome VARCHAR(90) NOT NULL,
    cpf_cnpj VARCHAR(14) NOT NULL,
    email VARCHAR(90),
    telefone VARCHAR(20) NOT NULL,
    enderecos_id_fk BIGINT NOT NULL,
    FOREIGN KEY (enderecos_id_fk) REFERENCES enderecos (id),
    INDEX (nome)
);
```

```
CREATE TABLE IF NOT EXISTS servicos (
    id BIGINT PRIMARY KEY NOT NULL AUTO_INCREMENT,
    nome_servico VARCHAR(45) NOT NULL,
    valor DECIMAL(10 , 2 ) NOT NULL,
    descricao LONGTEXT NOT NULL
);

CREATE TABLE IF NOT EXISTS faturamentos (
    id BIGINT PRIMARY KEY NOT NULL AUTO_INCREMENT,
    clientes_id_fk BIGINT NOT NULL,
    FOREIGN KEY (clientes_id_fk) REFERENCES clientes (id),
    servicos_id_fk BIGINT NOT NULL,
    FOREIGN KEY (servicos_id_fk) REFERENCES servicos (id),
    data_pagamento TIMESTAMP NOT NULL,
    valor DECIMAL(10 , 2 ) NOT NULL,
    status_pagamento BOOLEAN NOT NULL DEFAULT FALSE
);

CREATE TABLE IF NOT EXISTS servicos_has_clientes (
    servicos_id_fk BIGINT NOT NULL,
    FOREIGN KEY (servicos_id_fk) REFERENCES servicos (id),
    clientes_id_fk BIGINT NOT NULL,
    FOREIGN KEY (clientes_id_fk) REFERENCES clientes (id),
    data_servico TIMESTAMP
);

-- Inserir registro único na tabela usuarios
INSERT INTO usuarios (login, senha)
VALUES
('jetinADM', 'ADM112233');
```

```
-- Inserir registros na tabela enderecos

INSERT INTO enderecos (cep, logradouro, numero, complemento, bairro, cidade, estado, pais)

VALUES

('85852220', 'Rua dos Lírios', 456, null, 'Jardim São Paulo', 'Foz do Iguaçu', 'Paraná', 'Brasil'),

('85853400', 'Avenida das Acácias', 789, 'Sala 201', 'Vila Borges', 'Foz do Iguaçu', 'Paraná',
'Brasil');


-- Inserir registros na tabela clientes

INSERT INTO clientes (nome, cpf_cnpj, email, telefone, enderecos_id_fk)

VALUES

('João da Silva', '123.456.789-00', 'joao@example.com', '(11) 1234-5678', 1),

('Maria Oliveira', '987.654.321-00', 'maria@example.com', '(22) 9876-5432', 2);


-- Inserir registros na tabela servicos

INSERT INTO servicos (nome_servico, valor, descricao)

VALUES

('Tráfego pago', 50.00, 'Serviço de tráfego pago, inclui blablabla'),

('Posicionamento digital', 100.00, 'Serviço de posicionamento digital, inclui tananan');


-- Inserir registros na tabela servicos_has_clientes

INSERT INTO servicos_has_clientes (servicos_id_fk, clientes_id_fk, data_servico)

VALUES

(1, 2, NOW()),

(2, 2, NOW());


-- Seleciona id, nome e cpf ou cnpj do cliente com o sobrenome Oliveira (seleção com filtro)

SELECT c.id, c.nome, c.cpf_cnpj FROM clientes AS c WHERE c.nome LIKE '%Oliveira%';


-- Seleção controlada de dados com inner join

SELECT c.id, c.nome, c.cpf_cnpj, s.id, s.nome_servico, sc.data_servico FROM clientes AS c
```

```
INNER JOIN servicos_has_clientes AS sc ON sc.clientes_id_fk = c.id

INNER JOIN servicos AS s ON s.id = sc.servicos_id_fk;


-- Automação, a cada registro inserido na tabela servicos_has_clientes, será criado um novo registro na tabela faturamentos.

DELIMITER //

CREATE TRIGGER novo_faturamento

AFTER INSERT ON servicos_has_clientes

FOR EACH ROW

BEGIN

DECLARE valor_servico decimal(10,2);

SELECT valor INTO valor_servico FROM servicos WHERE servicos.id = new.servicos_id_fk;

INSERT INTO faturamentos(clientes_id_fk, servicos_id_fk, data_pagamento, valor, status_pagamento) VALUES

(new.clientes_id_fk, new.servicos_id_fk, now(), valor_servico, false);

END //

DELIMITER ;


-- Automação, chamando um procedure para mudar o status do pagamento para verdadeiro, ou seja, já foi pago.

DELIMITER //

CREATE PROCEDURE pagamento_realizado(IN id_faturamento BIGINT)

BEGIN

	DECLARE pagamento_atualizado BOOLEAN;

	SET @IDFATURAMENTO = id_faturamento;

	SELECT status_pagamento INTO pagamento_atualizado FROM faturamentos WHERE id = @IDFATURAMENTO;


   UPDATE faturamentos SET status_pagamento = TRUE WHERE id = @IDFATURAMENTO;

END //

DELIMITER ;
```

```
-- Chamando o procedure para atualizar o status de pagamento do faturamento cujo id = 1

call pagamento_realizado(1);


describe faturamentos;


-- Selecionando os clientes e detalhes do pagamento onde status_pagamento = true, ou seja, onde o pagamento já foi realizado.

SELECT c.id, c.nome, c.cpf_cnpj, c.email, c.telefone, c.enderecos_id_fk, f.data_pagamento, f.valor, f.status_pagamento FROM clientes AS c

INNER JOIN faturamentos AS f ON f.clientes_id_fk = c.id WHERE f.status_pagamento = true;
```