Universidade de Fortaleza

	AV1	AV2	AV3
1ª ch			
2ª ch			

Curso:	Disciplina:	Código/Turma:
Professor/a:		Data:
Aluno/a:		Matrícula:

INSTRUÇÕES PARA O DESENVOLVIMENTO DO TRABALHO

- Trabalho é em Equipe
- O Trabalho vale 30% da nota da disciplina.

Implementar um Processador de Consultas

1. **Interface gráfica obrigatória** mostrando o funcionamento do processador de consultas.

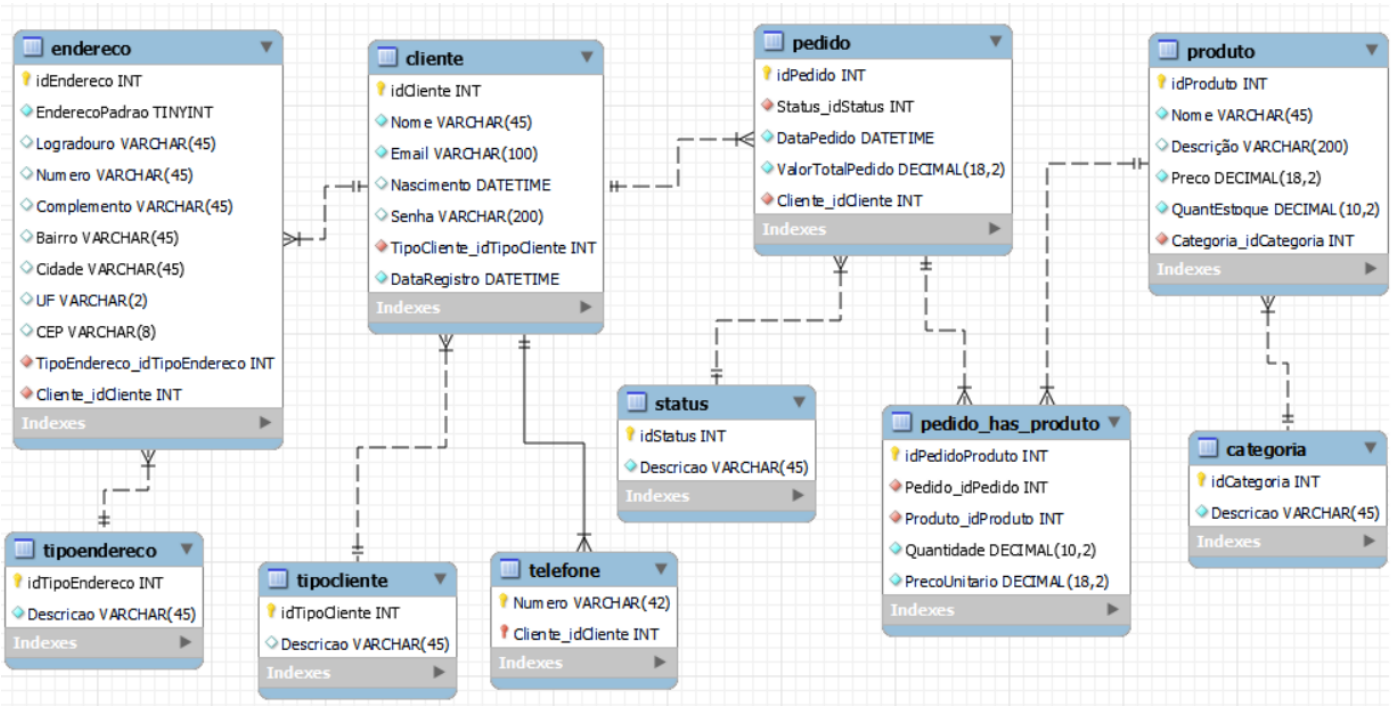
2. **Funcionalidades** principais:
   a. Parser (Análise) de uma consulta SQL;
   b. Geração do grafo de operadores da consulta;
   c. Ordem de execução da consulta;
   d. Exibição dos resultados na interface gráfica;

3. **Entidades/estruturas e algoritmos** a serem implementadas (sugestão -> usando POO como padrão):
   a. Parser**:** componente que analisa e separa os componentes principais da string com a consulta SQL. **Pode** ser implementado por meio de expressões regulares. Limitaremos o parse a:
      i. Select, From, Where, Join On (possibilidade de **vários** joins);
      ii. Operadores =, >, <, <=, >=, <>, And, (,) ;

   b. Grafo de operadores: estrutura de dados grafo com a representação interna da consulta, onde os nós são os operadores/tabelas/constantes e as arestas representam a direção do fluxo dos resultados intermediários, assim como os predicados que os objetos devem satisfazer. Representa a estratégia básica de execução da consulta para gerar o resultado.

   c. Processador de consultas simples: recebe o grafo de operadores como entrada, percorre o grafo e gera a ordem de execução correspondente.

4. **Banco de Dados:**
   a. O banco de dados exemplo a ser representado deve se basear no modelo que segue;
   b. O banco exemplo servirá como base na validação dos nomes de tabelas e campos no momento da validação da cláusula SQL.
   c. Se desejarem criar o banco, no final desde documento foram fornecidos os Scripts para criação do banco de dados para o MySQL.



5. **Heurísticas Básicas a serem usadas**
    a. Aplicar primeiro as operações que reduzem o tamanho dos resultados intermediários
        i. operações de seleção - reduzem o número de tuplas
        ii. operações de projeção - reduzem o número de atributos

    b. Aplicar primeiro as operações de seleção e de junção mais restritivas
        i. reordenar os nós folha da árvore de consulta
        ii. evitar a operação de produto cartesiano
        iii. ajustar o restante da árvore de forma apropriada

6. **Funcionamento:**
    a. A string com a consulta SQL é entrada na interface gráfica;
    b. A string é parseada e o comando SQL é validado;
    c. O comando SQL é convertido para álgebra relacional;
    d. A álgebra relacional é otimizada conforme as heurísticas solicitadas;
    e. O grafo de operadores é construído em memória;
    f. O grafo de operadores deve ser mostrado na Interface gráfica;
    g. O resultado da consulta mostrando cada operação e a ordem que será executada, é exibido na interface gráfica.

7. **Critérios de Avaliação:**

Critério	Nota
Interface Gráfica Existente e funcional	1,5
Existe local para inserir a string com a consulta SQL:	1,0
A string é parseada	1,0
Campos e comandos errados são validados	1,0
O grafo Otimizado de operadores deve ser mostrado na GUI	1,0
Ordem de execução da consulta deve ser mostrada (plano de execução)	1,5
Heurística Redução de Tuplas apresentada no grafo	1,0
Heurística Redução de Atributos apresentada no grafo	1,0
Demais Heurísticas	1,0
Total	10,0

Scripts SQL para criação do Banco Exemplo (Opcional)

```
-- Criando o Schema BD_Vendas
-- -----------------------------------------------------
CREATE SCHEMA BD_Vendas;
USE BD_Vendas ;

-- Criando Tabelas e PKs
-- -----------------------------------------------------
-- Table Categoria
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS Categoria (
  idCategoria INT NOT NULL,
  Descricao VARCHAR(45) NOT NULL,
  PRIMARY KEY (idCategoria))
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table Produto
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS Produto (
  idProduto INT NOT NULL,
  Nome VARCHAR(45) NOT NULL,
  Descricao VARCHAR(200) NULL,
  Preco DECIMAL(18,2) NOT NULL DEFAULT 0,
  QuantEstoque DECIMAL(10,2) NOT NULL DEFAULT 0,
  Categoria_idCategoria INT NOT NULL,
  PRIMARY KEY (idProduto)
  )
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table TipoCliente
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS TipoCliente (
  idTipoCliente INT NOT NULL,
  Descricao VARCHAR(45) NULL,
  PRIMARY KEY (idTipoCliente))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table Cliente
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS Cliente (
  idCliente INT NOT NULL,
  Nome VARCHAR(45) NOT NULL,
  Email VARCHAR(100) NOT NULL,
  Nascimento DATETIME NULL,
  Senha VARCHAR(200) NULL,
  TipoCliente_idTipoCliente INT NOT NULL,
  DataRegistro DATETIME NOT NULL DEFAULT Now(),
  PRIMARY KEY (idCliente))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table TipoEndereco
-- -----------------------------------------------------
```

```
CREATE TABLE IF NOT EXISTS TipoEndereco (
  idTipoEndereco INT NOT NULL,
  Descricao VARCHAR(45) NOT NULL,
  PRIMARY KEY (idTipoEndereco))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table Endereco
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS Endereco (
  idEndereco INT NOT NULL,
  EnderecoPadrao TINYINT NOT NULL DEFAULT 0,
  Logradouro VARCHAR(45) NULL,
  Numero VARCHAR(45) NULL,
  Complemento VARCHAR(45) NULL,
  Bairro VARCHAR(45) NULL,
  Cidade VARCHAR(45) NULL,
  UF VARCHAR(2) NULL,
  CEP VARCHAR(8) NULL,
  TipoEndereco_idTipoEndereco INT NOT NULL,
  Cliente_idCliente INT NOT NULL,
  PRIMARY KEY (idEndereco))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table Telefone
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS Telefone (
  Numero VARCHAR(42) NOT NULL,
  Cliente_idCliente INT NOT NULL,
  PRIMARY KEY (Numero, Cliente_idCliente))
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table Status
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS Status (
  idStatus INT NOT NULL,
  Descricao VARCHAR(45) NOT NULL,
  PRIMARY KEY (idStatus))
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table Pedido
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS Pedido (
  idPedido INT NOT NULL,
  Status_idStatus INT NOT NULL,
  DataPedido DATETIME NOT NULL DEFAULT Now(),
  ValorTotalPedido DECIMAL(18,2) NOT NULL DEFAULT 0,
  Cliente_idCliente INT NOT NULL,
  PRIMARY KEY (idPedido)  )
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table Pedido_has_Produto
```

```
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS Pedido_has_Produto (
  idPedidoProduto INT NOT NULL AUTO_INCREMENT,
  Pedido_idPedido INT NOT NULL,
  Produto_idProduto INT NOT NULL,
  Quantidade DECIMAL(10,2) NOT NULL,
  PrecoUnitario DECIMAL(18,2) NOT NULL,
  PRIMARY KEY (idPedidoProduto))
ENGINE = InnoDB;

-- Criando FKs.

CREATE INDEX fk_Produto_Categoria_idx ON Produto (Categoria_idCategoria ASC) ;

Alter Table Produto
add CONSTRAINT fk_Produto_Categoria
    FOREIGN KEY (Categoria_idCategoria)
    REFERENCES Categoria (idCategoria)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION;


CREATE INDEX fk_Cliente_TipoCliente_idx ON Cliente (TipoCliente_idTipoCliente ASC) ;
Alter Table Cliente
ADD CONSTRAINT fk_Cliente_TipoCliente
    FOREIGN KEY (TipoCliente_idTipoCliente)
    REFERENCES TipoCliente (idTipoCliente)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION;


CREATE INDEX fk_Endereco_TipoEndereco1_idx ON Endereco (TipoEndereco_idTipoEndereco
ASC) ;

CREATE INDEX fk_Endereco_Cliente1_idx ON Endereco (Cliente_idCliente ASC) ;

Alter Table Endereco
ADD  CONSTRAINT fk_Endereco_TipoEndereco
    FOREIGN KEY (TipoEndereco_idTipoEndereco)
    REFERENCES TipoEndereco (idTipoEndereco)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
ADD  CONSTRAINT fk_Endereco_Cliente
    FOREIGN KEY (Cliente_idCliente)
    REFERENCES Cliente (idCliente)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION;


CREATE INDEX fk_Telefone_Cliente_idx ON Telefone (Cliente_idCliente ASC) ;

Alter Table Telefone
ADD  CONSTRAINT fk_Telefone_Cliente
    FOREIGN KEY (Cliente_idCliente)
    REFERENCES Cliente (idCliente)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION;
```

```
    CREATE INDEX fk_Pedido_Status1_idx ON Pedido (Status_idStatus ASC) ;

CREATE INDEX fk_Pedido_Cliente1_idx ON Pedido (Cliente_idCliente ASC) ;

Alter Table Pedido
ADD CONSTRAINT fk_Pedido_Status
   FOREIGN KEY (Status_idStatus)
   REFERENCES Status (idStatus)
   ON DELETE NO ACTION
   ON UPDATE NO ACTION,
ADD CONSTRAINT fk_Pedido_Cliente
   FOREIGN KEY (Cliente_idCliente)
   REFERENCES Cliente (idCliente)
   ON DELETE NO ACTION
   ON UPDATE NO ACTION;

CREATE INDEX fk_Pedido_has_Produto_Produto_idx ON Pedido_has_Produto
(Produto_idProduto ASC) ;

CREATE INDEX fk_Pedido_has_Produto_Pedido_idx ON Pedido_has_Produto (Pedido_idPedido
ASC) ;


Alter Table Pedido_has_Produto
ADD CONSTRAINT fk_Pedido_has_Produto_Pedido
   FOREIGN KEY (Pedido_idPedido)
   REFERENCES Pedido (idPedido)
   ON DELETE NO ACTION
   ON UPDATE NO ACTION,
ADD  CONSTRAINT fk_Pedido_has_Produto_Produto
   FOREIGN KEY (Produto_idProduto)
   REFERENCES Produto (idProduto)
   ON DELETE NO ACTION
   ON UPDATE NO ACTION;
```