

Universidade Federal de Viçosa  
Campus UFV-Florestal  
Instituto de Ciências Exatas e Tecnológicas  
Ciência da Computação



## Relatório do Trabalho Prático II

**Disciplina:** Banco de dados

**Professor:** Daniel Mendes Barbosa

**Aluno:**

Matheus Júnio da Silva - 5382

# Sumário

1. Introdução
2. Estrutura do Projeto
3. Diagrama do Banco de Dados
  1. Versão Original
  2. Versão Expandida
4. Consultas SQL
  1. Questão 1
  2. Questão 2
  3. Questão 3
  4. Questão 4
  5. Questão 5
  6. Questão 6
  7. Questão 7
  8. Questão 8
  9. Questão 9
  10. Questão 10
  11. Questão 11
  12. Questão 12
  13. Questão 13
5. Script de criação dos bancos de dados

# 1 Introdução

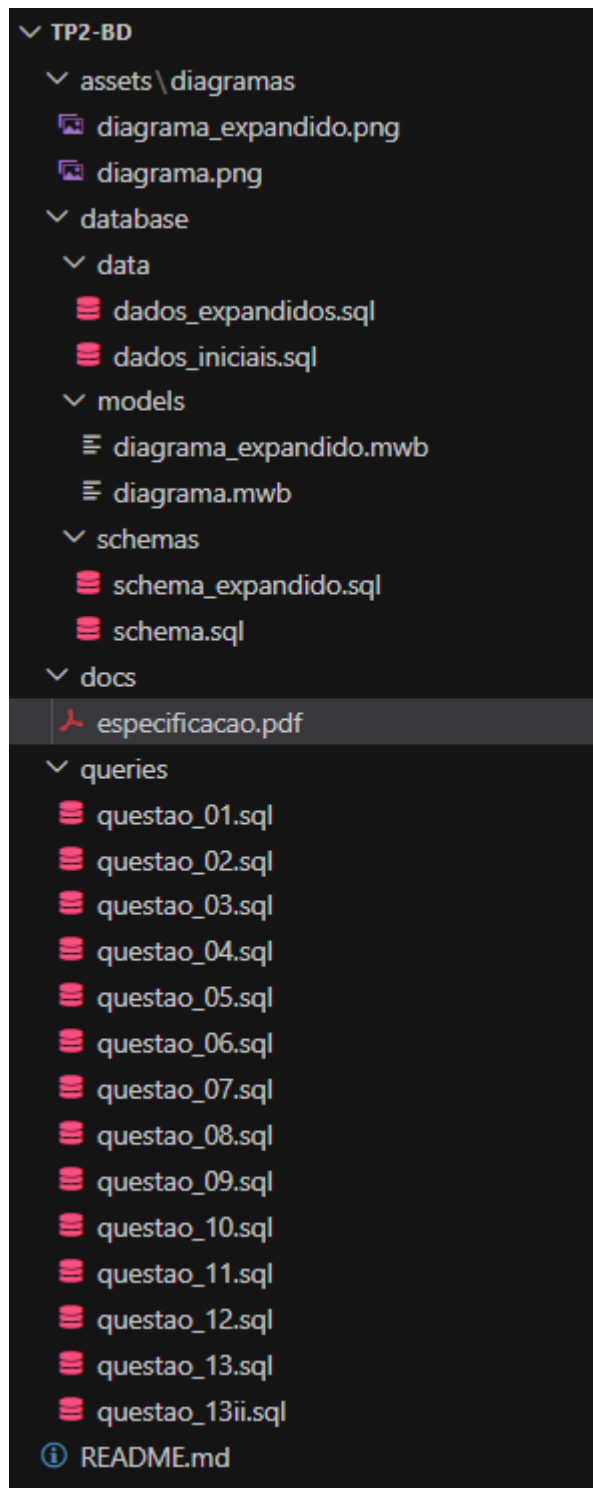
Este relatório apresenta o desenvolvimento e resultados do Trabalho Prático II, proposto no contexto da disciplina de Banco de Dados. O objetivo é projetar, implementar e explorar um banco de dados relacional para um colecionador de videogames, utilizando MySQL e MySQL Workbench. Todas as questões propostas foram resolvidas de forma incremental, seguindo a ordem obrigatória, conforme solicitado.

---

## 2 Estrutura do Projeto

A estrutura do projeto foi organizada da seguinte forma:

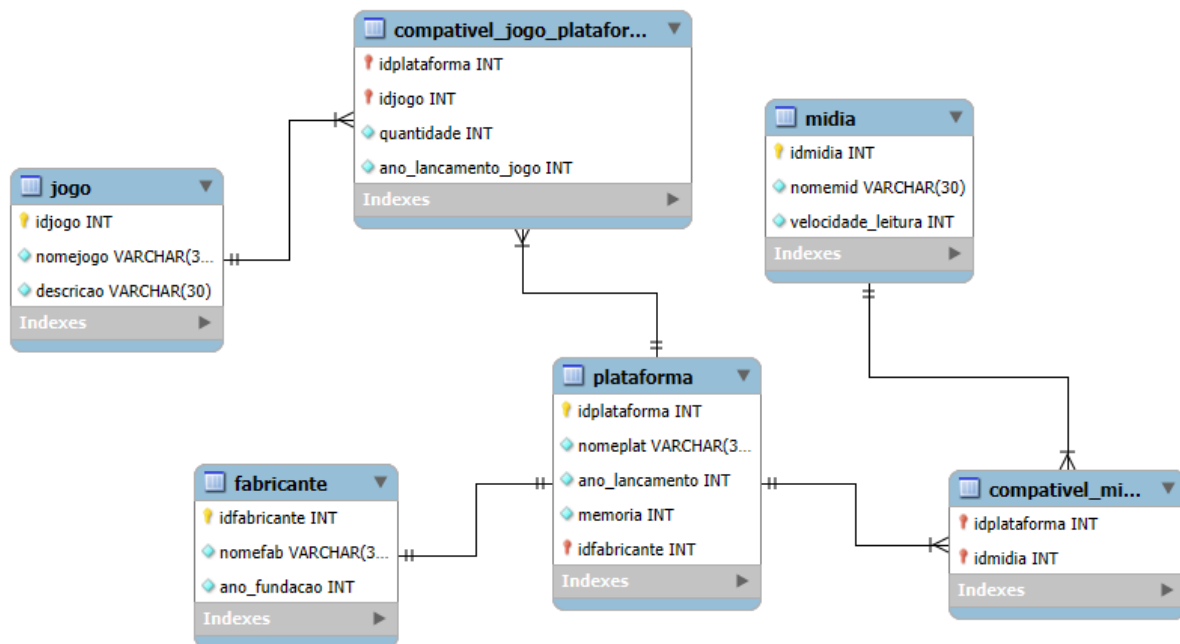
```
① README.md
1  Trabalho_Pratico_II
2  |— docs/                      # Documentação e relatórios
3  |   |— relatorio.pdf          # Relatório principal em PDF
4  |   |— especificacao.pdf      # especificação do que foi pedido para o trabalho
5  |
6  |— database/                  # Arquivos relacionados ao banco de dados
7  |   |— models/                # Modelos e diagramas do banco de dados
8  |   |   |— diagrama.mwb       # Arquivo do MySQL Workbench
9  |   |   |— diagrama_expandido.mwb # Versão expandida (questões 11-13)
10 |
11 |   |— schemas/                # Scripts de criação do banco de dados
12 |   |   |— schema.sql          # Script principal
13 |   |   |— schema_expandido.sql # Versão expandida (questões 11-13)
14 |
15 |   |— data/                    # Dados para popular o banco
16 |   |   |— dados_iniciais.sql  # Script para dados iniciais
17 |   |   |— dados_expandido.sql # Dados adicionais para expansão (questao 12)
18 |
19 |— queries/                     # Consultas SQL para responder às questões
20 |   |— questao_01.sql          # Consulta para a questão 1
21 |   |— questao_02.sql          # Consulta para a questão 2
22 |   |— ...                      # Outras consultas
23 |   |— questoes_expansao.sql   # Consultas das questões 12 e 13
24 |
25 |— assets/                      # Arquivos auxiliares (imagens)
26 |   |— diagramas/              # Diagramas e visuais adicionais
27 |       |— diagrama.png        # Diagrama original
28 |       |— diagrama_expandido.png # Diagrama expandido
29
```



### 3 Diagrama do Banco de Dados

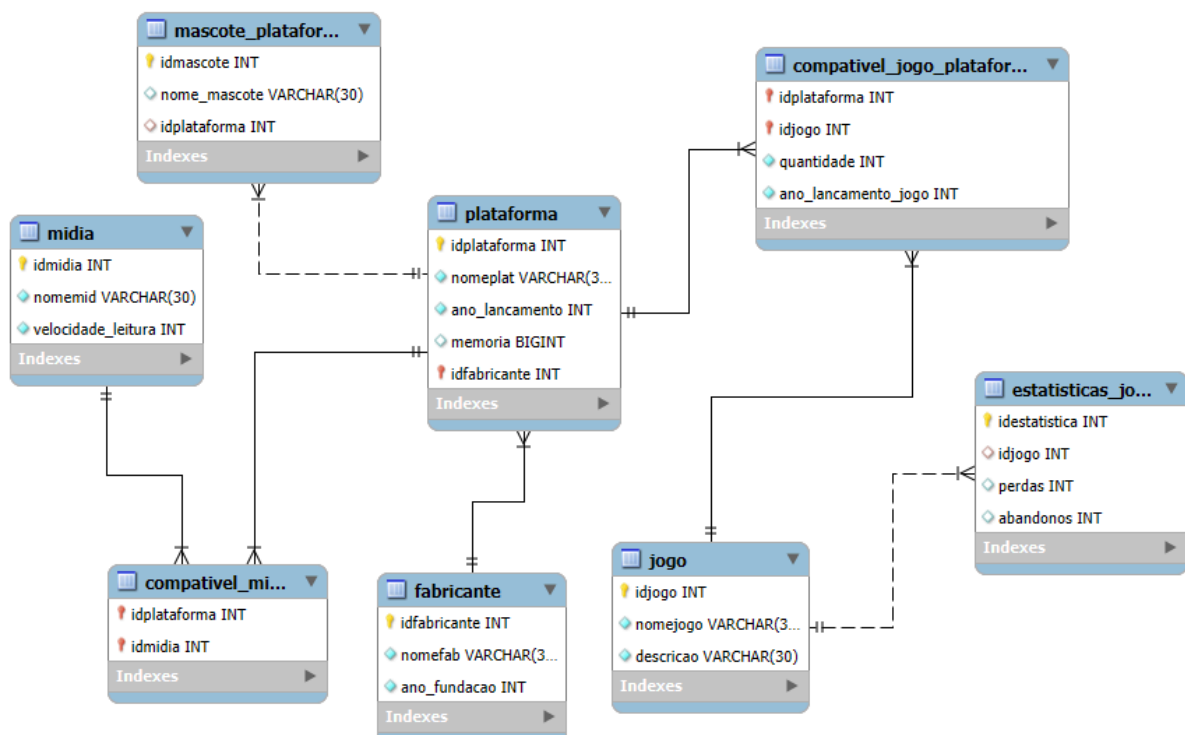
#### Versão Original

O diagrama a seguir representa o banco de dados original, projetado para atender às primeiras 10 questões. Ele foi desenvolvido no MySQL Workbench e exportado em formato PNG.



## Versão Expandida

A versão expandida do banco de dados inclui novas tabelas e atributos, desenvolvidos para atender às questões 11 a 13. O diagrama atualizado é apresentado abaixo:

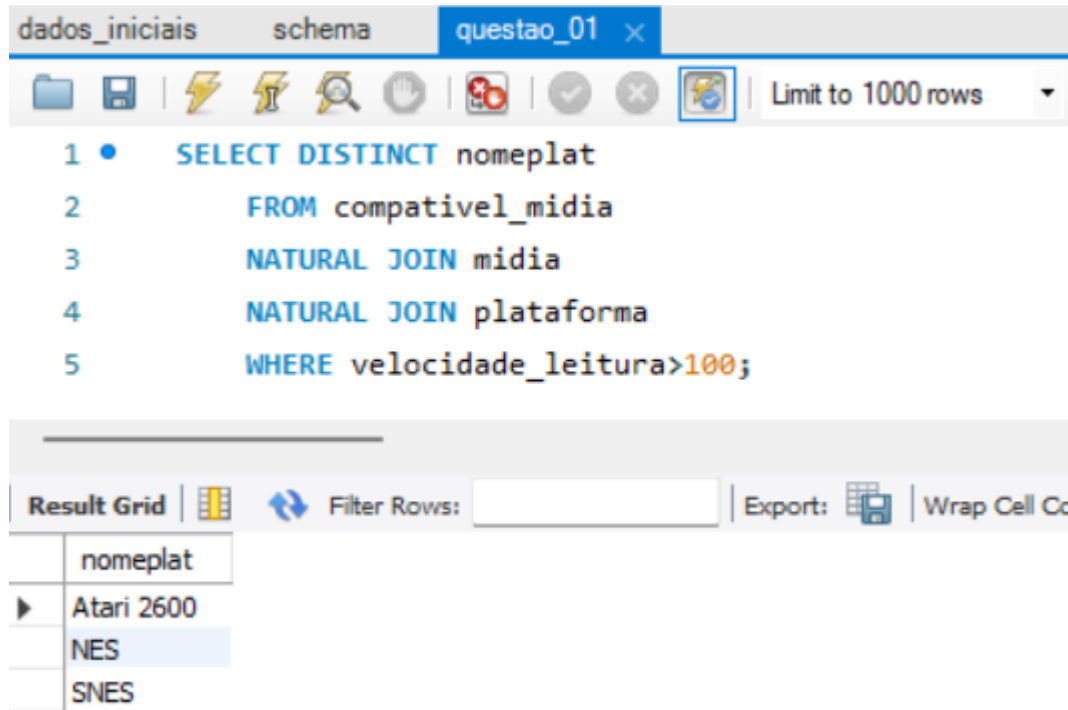


## 4 Consultas SQL

### Questão 1

**Descrição:** Recuperar os nomes das plataformas que possuam pelo menos uma mídia com velocidade de leitura superior a 100. A tabela resultante da execução desta consulta não deverá ter repetição de nomes das plataformas.

**Screenshot da consulta SQL e de seu resultado:**



The screenshot shows a SQL query editor with a toolbar and a results pane. The query is as follows:

```
1 • SELECT DISTINCT nomeplat
2     FROM compativel_midia
3     NATURAL JOIN midia
4     NATURAL JOIN plataforma
5     WHERE velocidade_leitura>100;
```

The results pane shows a table with the following data:

| nomeplat   |
|------------|
| Atari 2600 |
| NES        |
| SNES       |

## Questão 2

**Descrição:** Retornar os nomes das plataformas e seus anos de lançamento, desde que o fabricante tenha sido fundado após o ano de 1970.

**Screenshot da consulta SQL e de seu resultado:**

|                |        |            |            |   |
|----------------|--------|------------|------------|---|
| dados_iniciais | schema | questao_01 | questao_02 | x |
|----------------|--------|------------|------------|---|

Limit to 1000 rows

```

1 • SELECT nomeplat,ano_lancamento
2     FROM plataforma
3     JOIN fabricante
4     ON plataforma.idfabricante=fabricante.idfabricante
5     WHERE ano_fundacao>1970;

```

---

Result Grid
Filter Rows:
Export:
Wrap Cell Content:

|   | nomeplat   | ano_lancamento |
|---|------------|----------------|
| ▶ | Xbox360    | 2005           |
|   | Atari 2600 | 1977           |

### Questão 3

**Descrição:** Recuperar os nomes das mídias e suas respectivas velocidades de leitura, desde que suas velocidades de leitura estejam entre 10 e 30. Os resultados devem aparecer em ordem decrescente de velocidades de leitura e, em caso de empate, em ordem alfabética dos nomes.

**Screenshot da consulta SQL e de seu resultado:**

|                |        |            |            |            |   |
|----------------|--------|------------|------------|------------|---|
| dados_iniciais | schema | questao_01 | questao_02 | questao_03 | x |
|----------------|--------|------------|------------|------------|---|

Limit to 1000 rows

```

1 • SELECT nomemid,velocidade_leitura
2     FROM midia
3     WHERE velocidade_leitura BETWEEN 10 AND 30
4     ORDER BY velocidade_leitura DESC,nomemid;

```

---

Result Grid
Filter Rows:
Export:
Wrap Cell Content:

|   | nomemid | velocidade_leitura |
|---|---------|--------------------|
| ▶ | DVD     | 20                 |
|   | CD      | 10                 |

#### Questão 4

**Descrição:** Inserir a plataforma "WiiU" com os seguintes atributos: chave primária 7, ano de lançamento 2012, memória 2147483648 bytes, e fabricante "Nintendo". A inserção deve usar uma subconsulta para determinar o id do fabricante.

**Screenshot da consulta SQL e de seu resultado:**

```
1  •  INSERT INTO `mydb`.`plataforma` (  
2      `idplataforma`,  
3      `nomeplat`,  
4      `ano_lancamento`,  
5      `memoria`,  
6      `idfabricante`  
7  )  
8  VALUES (  
9      7,  
10     'WiiU',  
11     2012,  
12     '2147483648',  
13     (SELECT idfabricante FROM fabricante  
14         WHERE nomefab="Nintendo"  
15     )  
16     );
```

#### Questão 5

**Descrição:** Recuperar o nome da plataforma, o nome do fabricante e a quantidade total de jogos disponíveis por plataforma. A coluna de quantidade total deve ser exibida como "numero\_jogos". Ordenar os resultados por nome da plataforma.

**Screenshot da consulta SQL e de seu resultado:**



```

1 • SELECT nomeplat,nomefab,SUM(quantidade) AS numero_jogos
2     FROM compativel_jogo_plataforma
3     NATURAL JOIN plataforma
4     NATURAL JOIN fabricante
5     NATURAL JOIN jogo
6     GROUP BY nomeplat
7     ORDER BY nomeplat;

```

| Result Grid  | Filter Rows: | Export:      | Wrap Cell Content: |
|--------------|--------------|--------------|--------------------|
| nomeplat     | nomefab      | numero_jogos |                    |
| Atari 2600   | Atari        | 4            |                    |
| NES          | Nintendo     | 2            |                    |
| PlayStation3 | Sony         | 5            |                    |
| SNES         | Nintendo     | 1            |                    |
| Wii          | Nintendo     | 4            |                    |
| Xbox360      | Microsoft    | 4            |                    |

## Questão 6

**Descrição:** Recuperar os nomes dos jogos que foram lançados para mais de uma plataforma.

**Screenshot da consulta SQL e de seu resultado:**

```

1 • SELECT nomejogo
2     NATURAL JOIN jogo
3     GROUP BY nomejogo
4     HAVING COUNT(*)>1;

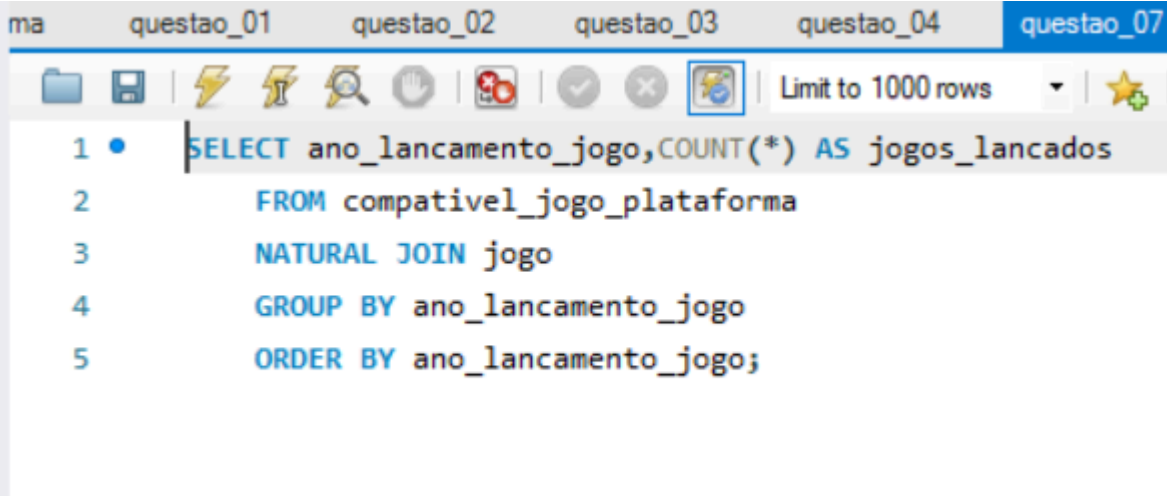
```

| Result Grid       | Filter Rows: | Export: | Wrap Cell Content: |
|-------------------|--------------|---------|--------------------|
| nomejogo          |              |         |                    |
| Mario Bros        |              |         |                    |
| F-Zero            |              |         |                    |
| Super Mario Bros  |              |         |                    |
| FIFA12            |              |         |                    |
| Street Fighter IV |              |         |                    |

## Questão 7

**Descrição:** Recuperar o número de jogos lançados em cada ano. A contagem deve considerar lançamentos únicos por ano e plataforma. A coluna com o número de jogos deve ser nomeada como "jogos\_lancados" e os resultados devem ser ordenados do ano mais antigo para o mais recente.

**Screenshot da consulta SQL e de seu resultado:**



The screenshot shows a SQL query editor with a toolbar at the top containing icons for file operations, execution, and settings. The query is as follows:

```
1 • SELECT ano_lancamento_jogo, COUNT(*) AS jogos_lancados
2     FROM compativel_jogo_plataforma
3     NATURAL JOIN jogo
4     GROUP BY ano_lancamento_jogo
5     ORDER BY ano_lancamento_jogo;
```

Below the query editor, the 'Result Grid' tab is active, displaying the query results in a table. The table has two columns: 'ano\_lancamento\_jogo' and 'jogos\_lancados'. The results are ordered by year from oldest to newest.

| ano_lancamento_jogo | jogos_lancados |
|---------------------|----------------|
| 1982                | 1              |
| 1983                | 2              |
| 1985                | 1              |
| 1990                | 1              |
| 2006                | 1              |
| 2007                | 2              |
| 2008                | 2              |
| 2010                | 1              |
| 2011                | 4              |
| 2012                | 1              |

## Questão 8

**Descrição:** Recuperar o nome do jogo com o ano de lançamento mais antigo, juntamente com o próprio ano e o nome da plataforma em que foi lançado.

**Screenshot da consulta SQL e de seu resultado:**

ma    questao\_01    questao\_02    questao\_03    questao\_04    **questao\_08**

Limit to 1000 rows

```

1 • SELECT nomejogo, ano_lancamento_jogo, nomeplat
2     FROM compativel_jogo_plataforma
3     NATURAL JOIN jogo
4     NATURAL JOIN plataforma
5     WHERE ano_lancamento_jogo=(
6         SELECT MIN(ano_lancamento_jogo)
7         FROM compativel_jogo_plataforma
8     );

```

Result Grid    Filter Rows:    Export:    Wrap Cell Content:

|   | nomejogo   | ano_lancamento_jogo | nomeplat   |
|---|------------|---------------------|------------|
| ▶ | River Raid | 1982                | Atari 2600 |

## Questão 9

**Descrição:** Calcular a média de memória utilizada pelos jogos, considerando jogos únicos (excluindo duplicatas de mesma plataforma, mas considerando múltiplas plataformas). A média deve ser baseada no total de memória das plataformas.

**Screenshot da consulta SQL e de seu resultado:**

Limit to 1000 rows

```

1 • SELECT AVG(memoria) AS media_memoria_utilizada
2     FROM (
3         SELECT DISTINCT jogo.idjogo, plataforma.memoria
4         FROM jogo
5         JOIN compativel_jogo_plataforma
6         ON jogo.idjogo = compativel_jogo_plataforma.idjogo
7         JOIN plataforma
8         ON compativel_jogo_plataforma.idplataforma = plataforma.idplataforma
9     ) AS jogos_unicos;

```

Result Grid    Filter Rows:    Export:    Wrap Cell Content:

|   | media_memoria_utilizada |
|---|-------------------------|
| ▶ | 218113481.1429          |

## Questão 10

**Descrição:** Recuperar os nomes dos fabricantes que já utilizaram mais de um tipo de mídia em suas plataformas. Os resultados devem ser ordenados alfabeticamente.

**Screenshot da consulta SQL e de seu resultado:**

The screenshot shows a SQL query editor with the following query:

```
1 • SELECT DISTINCT nomefab
2     FROM compativel_midia
3     NATURAL JOIN plataforma
4     NATURAL JOIN fabricante
5     GROUP BY nomefab
6     HAVING COUNT(idmidia)>1
7     ORDER BY nomefab;
```

Below the query editor, the results are displayed in a table:

| nomefab   |
|-----------|
| Microsoft |
| Nintendo  |
| Sony      |

## Questão 11: Expansão do Esquema

**Descrição:** Foram criadas duas tabelas adicionais:

**Tabela 1: mascote\_plataforma**

- **Descrição:** Cada plataforma pode ter um mascote associado, um personagem fictício ou símbolo que represente a marca ou a plataforma em eventos ou campanhas.
- **Atributos:**
  - **idmascote** (INT, PK): Identificador único do mascote.
  - **nome\_mascote** (VARCHAR(30)): Nome do mascote.

- **idplataforma** (INT, FK): Referência à plataforma associada.

```
queries > questao_11.sql
1  -- Tabela Mascote_Plataforma
2  CREATE TABLE mascote_plataforma (
3      idmascote INT PRIMARY KEY AUTO_INCREMENT,
4      nome_mascote VARCHAR(30),
5      idplataforma INT,
6      FOREIGN KEY (idplataforma) REFERENCES plataforma(idplataforma)
7  );
```

## Tabela 2: estatisticas\_jogo

- **Descrição:** Estatísticas engraçadas e curiosas sobre cada jogo, como quantas vezes os jogadores "perderam" no jogo ou quantas cópias foram deixadas de lado em coleções.
- **Atributos:**
  - **idestatistica** (INT, PK): Identificador único da estatística.
  - **idjogo** (INT, FK): Referência ao jogo.
  - **perdas** (INT): Número de vezes que os jogadores perderam o jogo.
  - **abandonos** (INT): Quantidade de vezes que o jogo foi deixado de lado.

```
9  -- Tabela Estatisticas_Jogo
10 CREATE TABLE estatisticas_jogo (
11     idestatistica INT PRIMARY KEY AUTO_INCREMENT,
12     idjogo INT,
13     perdas INT DEFAULT 0,
14     abandonos INT DEFAULT 0,
15     FOREIGN KEY (idjogo) REFERENCES jogo(idjogo)
16 );
17
```

## Questão 12: Inserção de Dados

Comandos de Inserção:

```

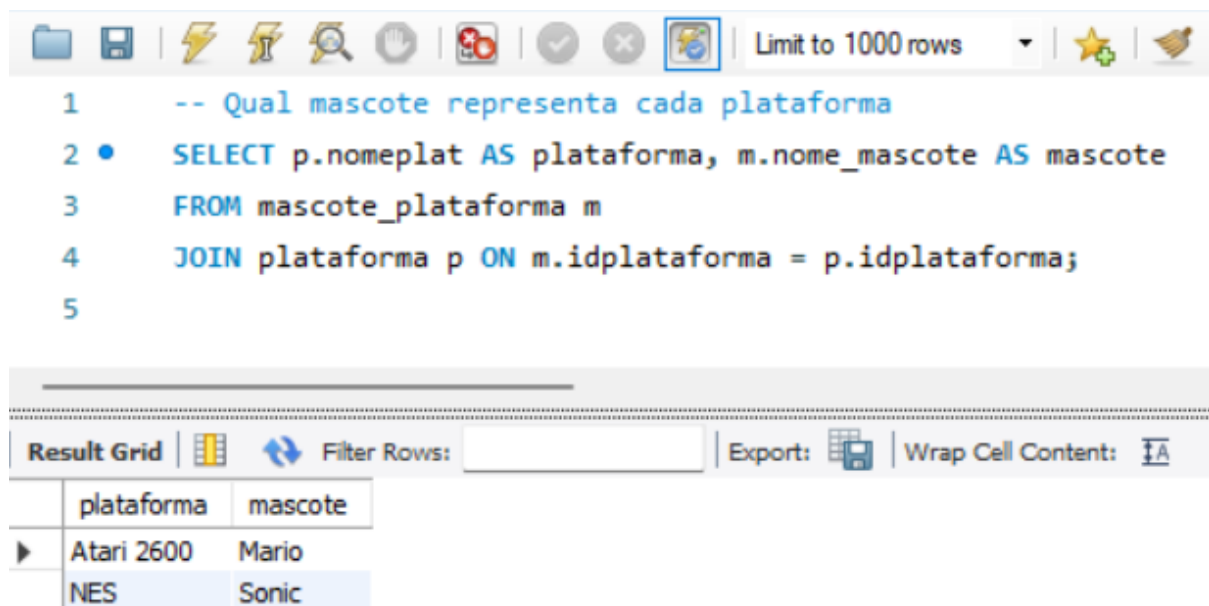
queries > questao_12.sql
1  -- Inserção de mascotes associados às plataformas
2  INSERT INTO mascote_plataforma (nome_mascote, idplataforma)
3  VALUES
4      ('Mario', 1), -- Mascote da Nintendo
5      ('Sonic', 2); -- Mascote da Sega
6
7  -- Inserção de estatísticas dos jogos
8  INSERT INTO estatisticas_jogo (idjogo, perdas, abandonos)
9  VALUES
10     (1, 500, 50), -- Jogo 1: 500 perdas, 50 abandonos
11     (2, 100, 20); -- Jogo 2: 100 perdas, 20 abandonos
12

```

### Questão 13: Perguntas Baseadas nas Novas Tabelas

Pergunta 1: Qual mascote representa cada plataforma?

Consulta SQL:



```

1  -- Qual mascote representa cada plataforma
2  • SELECT p.nomeplat AS plataforma, m.nome_mascote AS mascote
3     FROM mascote_plataforma m
4     JOIN plataforma p ON m.idplataforma = p.idplataforma;
5

```

|   | plataforma | mascote |
|---|------------|---------|
| ▶ | Atari 2600 | Mario   |
|   | NES        | Sonic   |

**Descrição:** Essa consulta retorna o nome das plataformas e os mascotes associados a elas.

Pergunta 2: Quais são os jogos mais abandonados pelos colecionadores?

Consulta SQL:

```
1  -- Quais são os jogos mais abandonados pelos colecionadores?
2  • SELECT j.nomejogo AS jogo, e.abandonos AS total_abandonos
3     FROM estatisticas_jogo e
4     JOIN jogo j ON e.idjogo = j.idjogo
5     ORDER BY e.abandonos DESC
6     LIMIT 5;
7
```

Result Grid | Filter Rows:  | Export: | Wrap Cell Content:

|   | jogo       | total_abandonos |
|---|------------|-----------------|
| ▶ | River Raid | 50              |
|   | Mario Bros | 20              |

**Descrição:** Lista os jogos com mais abandonos registrados, em ordem decrescente.

## 5 Scripts de criação dos bancos de dados

### 1 schema.sql (original)

```
Unset
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----
-- Schema mydb
-- -----

-- -----
-- Schema mydb
-- -----

CREATE SCHEMA IF NOT EXISTS `mydb` ;
-- -----
```

```

-- Schema new_schema1
-----

USE `mydb` ;

-----

-- Table `mydb`.`fabricante`
-----

CREATE TABLE IF NOT EXISTS `mydb`.`fabricante` (
  `idfabricante` INT NOT NULL AUTO_INCREMENT,
  `nomefab` VARCHAR(30) NOT NULL,
  `ano_fundacao` INT NOT NULL,
  PRIMARY KEY (`idfabricante`),
  UNIQUE INDEX `nomefab_UNIQUE` (`nomefab` ASC) VISIBLE)
ENGINE = InnoDB;

-----

-- Table `mydb`.`plataforma`
-----

CREATE TABLE IF NOT EXISTS `mydb`.`plataforma` (
  `idplataforma` INT NOT NULL AUTO_INCREMENT,
  `nomeplat` VARCHAR(30) NOT NULL,
  `ano_lancamento` INT NOT NULL,
  `memoria` INT NOT NULL,
  `idfabricante` INT NOT NULL,
  PRIMARY KEY (`idplataforma`, `idfabricante`),
  UNIQUE INDEX `nomeplat_UNIQUE` (`nomeplat` ASC) VISIBLE,
  INDEX `fk_plataforma_fabricante_idx` (`idfabricante` ASC) VISIBLE,
  CONSTRAINT `fk_plataforma_fabricante`
    FOREIGN KEY (`idfabricante`)
      REFERENCES `mydb`.`fabricante` (`idfabricante`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;

-----

-- Table `mydb`.`midia`
-----

CREATE TABLE IF NOT EXISTS `mydb`.`midia` (
  `idmidia` INT NOT NULL AUTO_INCREMENT,
  `nomemid` VARCHAR(30) NOT NULL,
  `velocidade_leitura` INT NOT NULL,
  PRIMARY KEY (`idmidia`),
  UNIQUE INDEX `nomemid_UNIQUE` (`nomemid` ASC) VISIBLE)
ENGINE = InnoDB;

```



```

-----
-- Table `mydb`.`jogo`
-----
CREATE TABLE IF NOT EXISTS `mydb`.`jogo` (
  `idjogo` INT NOT NULL AUTO_INCREMENT,
  `nomejogo` VARCHAR(30) NOT NULL,
  `descricao` VARCHAR(30) NOT NULL,
  PRIMARY KEY (`idjogo`))
ENGINE = InnoDB;

-----

-- Table `mydb`.`compativel_midia`
-----
CREATE TABLE IF NOT EXISTS `mydb`.`compativel_midia` (
  `idplataforma` INT NOT NULL,
  `idmidia` INT NOT NULL,
  PRIMARY KEY (`idplataforma`, `idmidia`),
  INDEX `fk_compativel_midia_plataforma1_idx` (`idplataforma` ASC) VISIBLE,
  INDEX `fk_compativel_midia_midia1_idx` (`idmidia` ASC) VISIBLE,
  CONSTRAINT `fk_compativel_midia_plataforma1`
    FOREIGN KEY (`idplataforma`)
      REFERENCES `mydb`.`plataforma` (`idplataforma`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  CONSTRAINT `fk_compativel_midia_midia1`
    FOREIGN KEY (`idmidia`)
      REFERENCES `mydb`.`midia` (`idmidia`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;

-----

-- Table `mydb`.`compativel_jogo_plataforma`
-----
CREATE TABLE IF NOT EXISTS `mydb`.`compativel_jogo_plataforma` (
  `idplataforma` INT NOT NULL,
  `idjogo` INT NOT NULL,
  `quantidade` INT NOT NULL,
  `ano_lancamento_jogo` INT NOT NULL,
  PRIMARY KEY (`idplataforma`, `idjogo`),
  INDEX `fk_compativel_jogo_plataforma_jogo1_idx` (`idjogo` ASC) VISIBLE,
  CONSTRAINT `fk_compativel_jogo_plataforma_plataforma1`
    FOREIGN KEY (`idplataforma`)
      REFERENCES `mydb`.`plataforma` (`idplataforma`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,

```

```

        CONSTRAINT `fk_compativel_jogo_plataforma_jogo1`
        FOREIGN KEY (`idjogo`)
        REFERENCES `mydb`.`jogo` (`idjogo`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

```

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```

## 2 schema\_expandido.sql (para atender às questões 11,12 e 13)

Unset

```
-- MySQL Workbench Forward Engineering
```

```

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

```

```
-- Schema mydb
```

```
-- Schema mydb
```

```

CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci ;
USE `mydb` ;

```

```
-- Table `mydb`.`jogo`
```

```

CREATE TABLE IF NOT EXISTS `mydb`.`jogo` (
  `idjogo` INT NOT NULL AUTO_INCREMENT,
  `nomejogo` VARCHAR(30) NOT NULL,
  `descricao` VARCHAR(30) NOT NULL,
  PRIMARY KEY (`idjogo`))

```

```

ENGINE = InnoDB
AUTO_INCREMENT = 11
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-----
-- Table `mydb`.`fabricante`
-----

CREATE TABLE IF NOT EXISTS `mydb`.`fabricante` (
  `idfabricante` INT NOT NULL AUTO_INCREMENT,
  `nomefab` VARCHAR(30) NOT NULL,
  `ano_fundacao` INT NOT NULL,
  PRIMARY KEY (`idfabricante`),
  UNIQUE INDEX `nomefab_UNIQUE` (`nomefab` ASC) VISIBLE)
ENGINE = InnoDB
AUTO_INCREMENT = 5
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-----
-- Table `mydb`.`plataforma`
-----

CREATE TABLE IF NOT EXISTS `mydb`.`plataforma` (
  `idplataforma` INT NOT NULL AUTO_INCREMENT,
  `nomeplat` VARCHAR(30) NOT NULL,
  `ano_lancamento` INT NOT NULL,
  `memoria` BIGINT NULL DEFAULT NULL,
  `idfabricante` INT NOT NULL,
  PRIMARY KEY (`idplataforma`, `idfabricante`),
  UNIQUE INDEX `nomeplat_UNIQUE` (`nomeplat` ASC) VISIBLE,
  INDEX `fk_plataforma_fabricante_idx` (`idfabricante` ASC) VISIBLE,
  CONSTRAINT `fk_plataforma_fabricante`
    FOREIGN KEY (`idfabricante`)
      REFERENCES `mydb`.`fabricante` (`idfabricante`))
ENGINE = InnoDB
AUTO_INCREMENT = 9
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-----
-- Table `mydb`.`compativel_jogo_plataforma`
-----

CREATE TABLE IF NOT EXISTS `mydb`.`compativel_jogo_plataforma` (
  `idplataforma` INT NOT NULL,
  `idjogo` INT NOT NULL,

```

```

    `quantidade` INT NOT NULL,
    `ano_lancamento_jogo` INT NOT NULL,
    PRIMARY KEY (`idplataforma`, `idjogo`),
    INDEX `fk_compativel_jogo_plataforma_jogo1_idx` (`idjogo` ASC) VISIBLE,
    CONSTRAINT `fk_compativel_jogo_plataforma_jogo1`
        FOREIGN KEY (`idjogo`)
        REFERENCES `mydb`.`jogo` (`idjogo`),
    CONSTRAINT `fk_compativel_jogo_plataforma_plataforma1`
        FOREIGN KEY (`idplataforma`)
        REFERENCES `mydb`.`plataforma` (`idplataforma`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

```

```

-- Table `mydb`.`midia`

```

```

CREATE TABLE IF NOT EXISTS `mydb`.`midia` (
    `idmidia` INT NOT NULL AUTO_INCREMENT,
    `nomemid` VARCHAR(30) NOT NULL,
    `velocidade_leitura` INT NOT NULL,
    PRIMARY KEY (`idmidia`),
    UNIQUE INDEX `nomemid_UNIQUE` (`nomemid` ASC) VISIBLE)
ENGINE = InnoDB
AUTO_INCREMENT = 8
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

```

```

-- Table `mydb`.`compativel_midia`

```

```

CREATE TABLE IF NOT EXISTS `mydb`.`compativel_midia` (
    `idplataforma` INT NOT NULL,
    `idmidia` INT NOT NULL,
    PRIMARY KEY (`idplataforma`, `idmidia`),
    INDEX `fk_compativel_midia_plataforma1_idx` (`idplataforma` ASC) VISIBLE,
    INDEX `fk_compativel_midia_midia1_idx` (`idmidia` ASC) VISIBLE,
    CONSTRAINT `fk_compativel_midia_midia1`
        FOREIGN KEY (`idmidia`)
        REFERENCES `mydb`.`midia` (`idmidia`),
    CONSTRAINT `fk_compativel_midia_plataforma1`
        FOREIGN KEY (`idplataforma`)
        REFERENCES `mydb`.`plataforma` (`idplataforma`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

```

```

-----
-- Table `mydb`.`estatisticas_jogo`
-----

CREATE TABLE IF NOT EXISTS `mydb`.`estatisticas_jogo` (
  `idestatistica` INT NOT NULL AUTO_INCREMENT,
  `idjogo` INT NULL DEFAULT NULL,
  `perdas` INT NULL DEFAULT '0',
  `abandonos` INT NULL DEFAULT '0',
  PRIMARY KEY (`idestatistica`),
  INDEX `idjogo` (`idjogo` ASC) VISIBLE,
  CONSTRAINT `estatisticas_jogo_ibfk_1`
    FOREIGN KEY (`idjogo`)
      REFERENCES `mydb`.`jogo` (`idjogo`))
ENGINE = InnoDB
AUTO_INCREMENT = 5
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-----

-- Table `mydb`.`mascote_plataforma`
-----

CREATE TABLE IF NOT EXISTS `mydb`.`mascote_plataforma` (
  `idmascote` INT NOT NULL AUTO_INCREMENT,
  `nome_mascote` VARCHAR(30) NULL DEFAULT NULL,
  `idplataforma` INT NULL DEFAULT NULL,
  PRIMARY KEY (`idmascote`),
  INDEX `idplataforma` (`idplataforma` ASC) VISIBLE,
  CONSTRAINT `mascote_plataforma_ibfk_1`
    FOREIGN KEY (`idplataforma`)
      REFERENCES `mydb`.`plataforma` (`idplataforma`))
ENGINE = InnoDB
AUTO_INCREMENT = 5
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```