Relatório Criação de Tabelas e Consultas SQL ChatBot\_Sis\_De\_Ensino

Relatório Elaborado por: Tiago Fernando Piveta

AQUI VAI O SUMARIO

Sumário

[1. Criando o Bando de Dados: 2](#_Toc185602060)

[2. Criando as Tabelas para o Bando de Dados: 3](#_Toc185602061)

[2.1. Explicação do Código SQL: 4](#_Toc185602062)

[2.2. Explicação do Output: 5](#_Toc185602063)

[2.3. Explicação das Linhas de Código (Query SQL) 2ºLote: 7](#_Toc185602064)

[2.4. Explicação do Output: 9](#_Toc185602065)

[2.5. Introdução do Banco de Dados 3ºLote: 10](#_Toc185602066)

[3. Inserindo dados nas tabelas: 12](#_Toc185602067)

[3.1. Inserção de Dados na Tabela Dim\_Alunos: 12](#_Toc185602068)

[3.2. Inserção de Dados na Tabela Dim\_Disciplinas: 15](#_Toc185602069)

[3.3. Inserção de Dados na Tabela Dim\_Professores: 17](#_Toc185602070)

[3.4. Inserção de Dados na Tabela Dim\_Turmas: 19](#_Toc185602071)

[3.5. Inserção de Dados na Tabela Dim\_Notas: 22](#_Toc185602072)

[3.6. Inserção de Dados na Tabela Dim\_Periodos: 24](#_Toc185602073)

[3.7. Inserção de Dados na Tabela Fato\_Professor\_Disciplina: 28](#_Toc185602074)

[3.8. Inserção de Dados na Tabela Fato\_Notas: 31](#_Toc185602075)

[3.9. Inserção de Dados na Tabela Fato\_Aluno\_Disciplina: 33](#_Toc185602076)

[3.10. Inserção de Dados na Tabela Fato\_Turmas: 36](#_Toc185602077)

[4. Ensaio sobre Possíveis Perguntas ao Chatbot e Respostas Através de Consultas SQL: 39](#_Toc185602078)

[Pergunta 1: Qual é a distribuição de alunos que estão nas faixas de cada nota de 2 em 2? Exemplo: de 0 a 2, 2 a 4, etc. 40](#_Toc185602079)

[Pergunta 2: Quais alunos estão abaixo da média e vão precisar de reforço ou serão reprovados? 42](#_Toc185602080)

[Pergunta 3: Quem são os alunos que já tem nota suficiente para passar(aprovado) mesmo no meio do semestre? 46](#_Toc185602081)

[Pergunta 4: Quais são as disciplinas com a maior média de notas finais entre os alunos? 49](#_Toc185602082)

[Pergunta 5: Quais alunos têm a menor quantidade de faltas? 51](#_Toc185602083)

[Pergunta 6: Quais são os professores com a maior quantidade de alunos matriculados? 53](#_Toc185602084)

[Pergunta 7: Qual a porcentagem de aprovação e reprovação dos alunos em cada disciplina? 55](#_Toc185602085)

[Pergunta 8: Qual turma tem o maior número de faltas acumuladas? 57](#_Toc185602086)

[Pergunta 9: Quem são os alunos com o maior número de faltas? 60](#_Toc185602087)

[Pergunta 10: Qual é a nota final mais alta e a mais baixa registrada no sistema? Quais alunos as possuem? 62](#_Toc185602088)

[Pergunta 11: Existe alguma correlação entre o número de presenças e a nota final dos alunos? 65](#_Toc185602089)

[Pergunta 12: Quais são os alunos com desempenho consistente (notas acima de 7 em todas as disciplinas)? 67](#_Toc185602090)

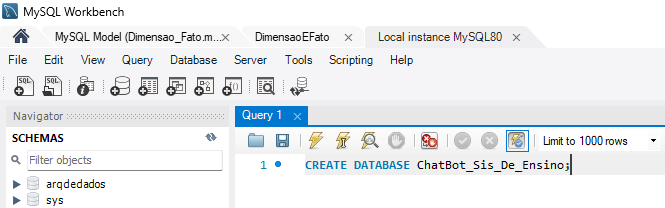
[Pergunta 13: Quais professores têm as turmas com as maiores médias de notas finais? 68](#_Toc185602091)

[Pergunta 14: Qual é o percentual de alunos aprovados, em recuperação e reprovados? 70](#_Toc185602092)

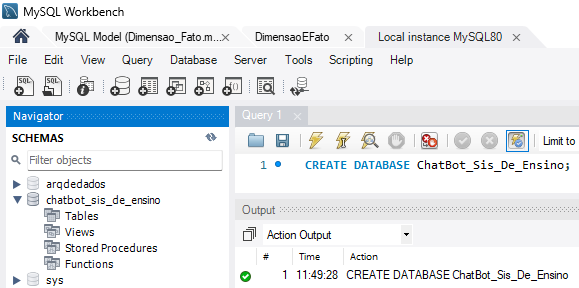
[Conclusão e Resultados Obtidos 73](#_Toc185602093)

# Criando o Bando de Dados:

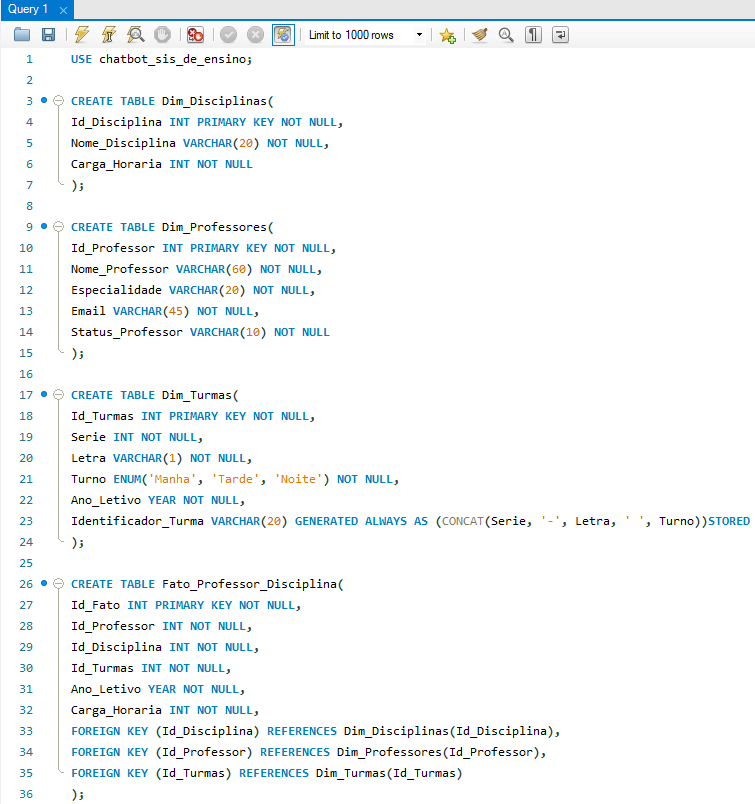
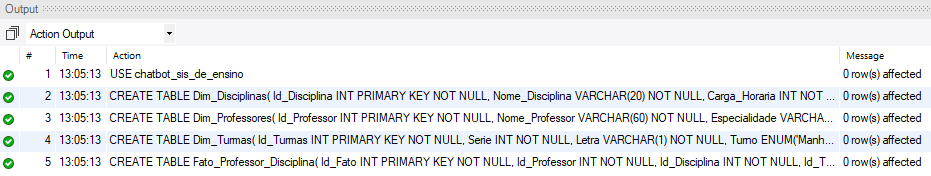
Inserindo o comando *CREATE DATABASE ChatBot\_Sis\_De\_Ensino*.



Confirmação visual que o comando de criar o Bando de Dados foi correto e que o Banco de dados que iremos usar está funcionando.



# Criando as Tabelas para o Bando de Dados:

Neste momento iremos ter uma abordagem conservadora criando as tabelas em “lotes” baseando-se em relacionamentos, essa abordagem vai nos ajudar a evitar erros na criação das tabelas e nas configurações de chaves estrangeiras, o primeiro “lote” de tabelas será *Dim\_Disciplinas*, *Dim\_Professores*, *Dim\_Turmas*, *Fato\_Professor\_Disciplina*.

## Explicação do Código SQL:

* USE chatbot\_sis\_de\_ensino;
  + Esta linha seleciona o banco de dados chatbot\_sis\_de\_ensino onde todas as tabelas serão criadas. O comando USE permite definir o contexto do banco de dados ativo para as operações SQL subsequentes.
* CREATE TABLE Dim\_Disciplinas:
  + O comando *CREATE TABLE* cria a tabela *Dim\_Disciplinas*, responsável por armazenar as informações relacionadas às disciplinas.
  + *Id\_Disciplina INT PRIMARY KEY NOT NULL:* Define a coluna *Id\_Disciplina* como a chave primária da tabela (única e não nula).
  + *Nome\_Disciplina VARCHAR(20) NOT NULL:* Coluna que armazena o nome da disciplina, com até 20 caracteres e não permite valores nulos.
  + *Carga\_Horaria INT NOT NULL:* Coluna que armazena a carga horária da disciplina em formato de número inteiro e não nulo.
* CREATE TABLE Dim\_Professores:
  + Cria a tabela *Dim\_Professores*, que armazena as informações dos professores.
  + *Id\_Professor INT PRIMARY KEY NOT NULL:* Chave primária da tabela, garantindo valores únicos e não nulos.
  + *Nome\_Professor VARCHAR(60) NOT NULL:* Nome completo do professor (até 60 caracteres).
  + *Especialidade VARCHAR(20) NOT NULL:* Área de especialidade do professor (até 20 caracteres).
  + *Email VARCHAR(45) NOT NULL:* Endereço de e-mail do professor (até 45 caracteres).
  + *Status\_Professor VARCHAR(10) NOT NULL:* Situação do professor (ativo, inativo, férias, etc.), com até 10 caracteres.
* CREATE TABLE Dim\_Turmas:
  + Cria a tabela *Dim\_Turmas*, que armazena os dados das turmas.
  + *Id\_Turmas INT PRIMARY KEY NOT NULL:* Chave primária da tabela.
  + *Serie INT NOT NULL:* Número da série da turma (exemplo: 1, 2, 3, etc.).
  + *Letra VARCHAR(1) NOT NULL:* Identificação alfabética da turma (exemplo: A, B, C).
  + *Turno ENUM('Manha', 'Tarde', 'Noite') NOT NULL:* Define o turno da turma como uma lista fixa de valores possíveis.
  + *Ano\_Letivo YEAR NOT NULL:* Ano letivo associado à turma.
  + *Identificador\_Turma VARCHAR(20) GENERATED ALWAYS AS (CONCAT(Serie, '-', Letra, ' ', Turno)) STORED:* Coluna calculada automaticamente com base na concatenação dos valores de *Serie*, *Letra* e *Turno*. A palavra-chave *STORED* indica que o valor gerado será armazenado na tabela.

## Explicação do Output:

* Na aba de Output, é possível visualizar as mensagens de execução das instruções SQL:
* USE chatbot\_sis\_de\_ensino
  + O banco de dados foi selecionado corretamente, sem erros.
* CREATE TABLE Dim\_Disciplinas
  + A tabela *Dim\_Disciplinas* foi criada com sucesso.
  + *0 row(s) affected:* Indica que nenhuma linha foi inserida, o que é esperado ao criar uma tabela vazia.
  + CREATE TABLE Dim\_Professores
    - A tabela Dim\_Professores foi criada com sucesso.
  + CREATE TABLE Dim\_Turmas
    - A tabela Dim\_Turmas foi criada com a coluna calculada Identificador\_Turma, confirmando que a fórmula foi aceita corretamente.
  + CREATE TABLE Fato\_Professor\_Disciplina
    - A tabela de fato Fato\_Professor\_Disciplina foi criada com sucesso, incluindo as chaves estrangeiras que garantem a integridade dos relacionamentos com as tabelas de dimensão Dim\_Disciplinas, Dim\_Professores e Dim\_Turmas.

Este **output** confirma que todas as tabelas foram criadas corretamente sem erros, e o ambiente está pronto para a inserção de dados e consultas futuras.

Agora iremos partir para o segundo “lote” de criar nossas tabelas e fazer as configurações necessárias por código SQL, nesse momento iremos implementar as tabelas: *Dim\_Alunos*, *Dim\_Notas* e *Fato\_Aluno\_Disciplina*.

## 2.3. Explicação das Linhas de Código (Query SQL) 2ºLote:

* Linha 1: USE chatbot\_sis\_de\_ensino
* Esta linha define o banco de dados a ser utilizado para a criação e manipulação das tabelas. O comando USE é responsável por selecionar o esquema (*database*) chamado *chatbot\_sis\_de\_ensino* para que as operações subsequentes ocorram dentro dele.
* Linhas 3 a 8: Criação da Tabela Dim\_Alunos:

A tabela Dim\_Alunos é uma tabela de dimensão que armazena informações básicas sobre os alunos.

* Id\_Alunos INT PRIMARY KEY NOT NULL: Cria o identificador único para cada aluno. É a chave primária e não permite valores nulos.
* Data\_Nascimento DATE NOT NULL: Define o campo para armazenar a data de nascimento dos alunos.
* Contato VARCHAR(15) NOT NULL: Campo para armazenar um telefone ou email de contato, com limite de 15 caracteres.
* Endereco VARCHAR(100) NOT NULL: Armazena o endereço completo do aluno, com até 100 caracteres.
* Matricula VARCHAR(20) UNIQUE: Define um código único de matrícula do aluno. O atributo UNIQUE assegura que não haverá duplicatas.
* Linhas 10 a 17: Criação da Tabela Dim\_NOTAS:

A tabela Dim\_NOTAS foi criada para armazenar informações sobre as notas e suas descrições.

* Id\_Notas INT PRIMARY KEY NOT NULL: Identificador único para cada nota. É a chave primária.
* Id\_Professor INT NOT NULL: Chave estrangeira que referência a tabela *Dim\_Professores*. Relaciona a nota ao professor responsável.
* Tipo\_Avaliacao VARCHAR(50) NOT NULL: Campo que descreve o tipo de avaliação (ex.: Prova, Trabalho, Atividade).
* Data\_Avaliacao DATE NOT NULL: Define a data em que a avaliação foi aplicada.
* Descricao VARCHAR(200): Campo opcional que permite uma descrição detalhada da avaliação.
* FOREIGN KEY (Id\_Professor) REFERENCES Dim\_Professores(Id\_Professor): Configura uma chave estrangeira que cria um relacionamento com a tabela *Dim\_Professores*, garantindo a integridade referencial.
* Linhas 19 a 35: Criação da Tabela Fato\_Aluno\_Disciplina:

A tabela *Fato\_Aluno\_Disciplina* é uma tabela de fatos que registra informações quantitativas e relacionamentos entre alunos, disciplinas, turmas e notas.

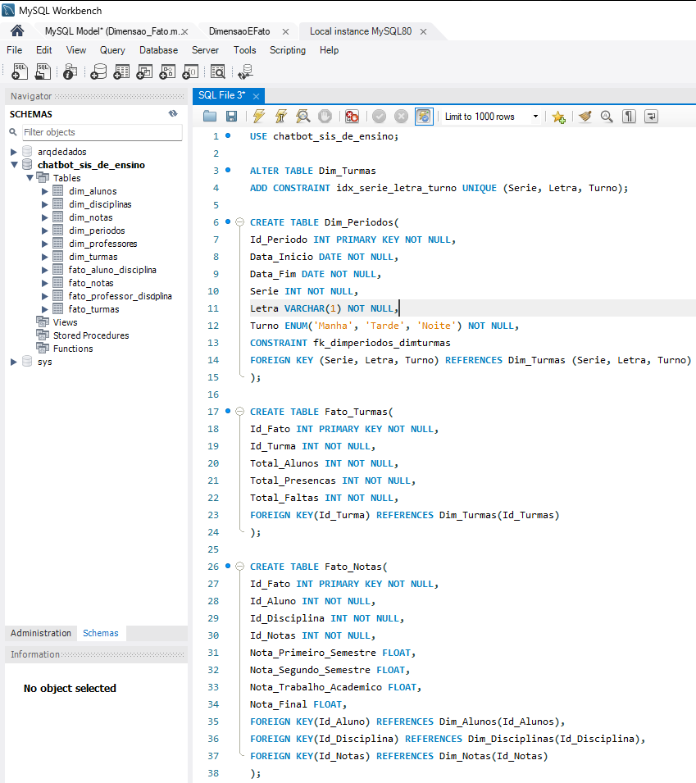
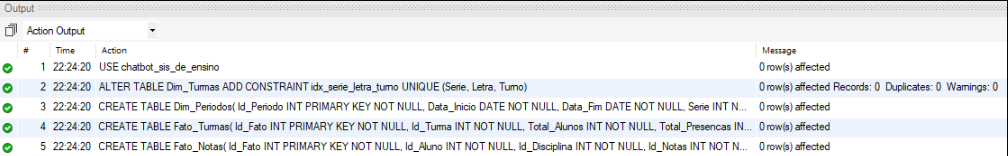
* Id\_Fato INT PRIMARY KEY NOT NULL: Identificador único do registro (chave primária).
* Id\_Aluno INT NOT NULL: Chave estrangeira que referência a tabela Dim\_Alunos.
* Id\_Disciplina INT NOT NULL: Chave estrangeira que referência a tabela Dim\_Disciplinas.
* Id\_Professor INT NOT NULL: Chave estrangeira que referência a tabela Dim\_Professores.
* Id\_Turma INT NOT NULL: Chave estrangeira que referência a tabela Dim\_Turmas.
* Id\_Notas INT NOT NULL: Chave estrangeira que referência a tabela Dim\_NOTAS.
* Presencas INT: Armazena o total de presenças do aluno na disciplina.
* Faltas INT: Armazena o total de faltas do aluno na disciplina.
* Data\_Do\_Registro DATE: Data em que o registro foi criado ou atualizado.
* FOREIGN KEY (Id\_Aluno) REFERENCES Dim\_Alunos(Id\_Alunos): Relaciona o aluno.
* FOREIGN KEY (Id\_Disciplina) REFERENCES Dim\_Disciplinas(Id\_Disciplina): Relaciona a disciplina.
* FOREIGN KEY (Id\_Professor) REFERENCES Dim\_Professores(Id\_Professor): Relaciona o professor.
* FOREIGN KEY (Id\_Turma) REFERENCES Dim\_Turmas(Id\_Turmas): Relaciona a turma.
* FOREIGN KEY (Id\_Notas) REFERENCES Dim\_NOTAS(Id\_Notas): Relaciona a nota.  
    
  1. Explicação do Output:

O Output apresentado na parte inferior confirma a execução bem-sucedida dos comandos SQL sem erros. Aqui está o detalhamento:

* + **Linha 1:** USE chatbot\_sis\_de\_ensino
  + O banco de dados *chatbot\_sis\_de\_ensino* foi selecionado com sucesso para operações.
  + **Linha 2:** CREATE TABLE Dim\_Alunos
  + A tabela *Dim\_Alunos* foi criada com seus respectivos atributos e restrições.
  + **Linha 3:** CREATE TABLE Dim\_NOTAS
  + A tabela *Dim\_NOTAS* foi criada com sucesso, incluindo a chave estrangeira *Id\_Professor* para a tabela *Dim\_Professores*.
  + **Linha 4:** CREATE TABLE Fato\_Aluno\_Disciplina
  + A tabela *Fato\_Aluno\_Disciplina* foi criada, conectando-se a outras tabelas através de suas chaves estrangeiras, como *Id\_Aluno*, *Id\_Disciplina*, *Id\_Professor*, *Id\_Turma* e *Id\_Notas*.

Cada ação concluída no **Output** exibe um status de "0 row(s) affected", o que é esperado, pois as tabelas foram apenas criadas sem inserção de dados até o momento.

Neste trecho, foram criadas três tabelas importantes: ***Dim\_Alunos***, ***Dim\_NOTAS*** e ***Fato\_Aluno\_Disciplina***, com destaque para suas relações e integridade referencial por meio de chaves estrangeiras. O **Output** confirma a execução correta de todos os comandos, sem erros ou avisos.

Nesse momento iremos partir para o terceiro e último “lote” de criação das tabelas e das demais configurações necessárias para o perfeito funcionamento das tabelas, para quando precisarmos fazer nossas inserções de dados e nas futuras consultas.  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  


## Introdução do Banco de Dados 3ºLote:

* **Linha 1:** USE chatbot\_sis\_de\_ensino;  
  Define o banco de dados a ser utilizado para a execução das próximas instruções. O banco de dados chatbot\_sis\_de\_ensino foi previamente criado.
* **Alteração na Tabela Dim\_Turmas:**
  + **Linha 3**: ALTER TABLE Dim\_Turmas ADD CONSTRAINT idx\_serie\_letra\_turno UNIQUE (Serie, Letra, Turno);  
    Adiciona uma restrição de unicidade na tabela *Dim\_Turmas*. A combinação dos atributos *Serie*, *Letra* e *Turno* será única, garantindo que não haja duplicatas na definição de turmas.

**Criação da Tabela Dim\_Periodos:**

* **Linhas 5 a 13:**  
  Cria a tabela *Dim\_Periodos* com os seguintes atributos:
  + **Id\_Periodo**: Chave primária que identifica de forma única cada registro.
  + **Data\_Inicio**: Data de início do período acadêmico.
  + **Data\_Fim**: Data de término do período acadêmico.
  + **Serie**: Número da série (int).
  + **Letra**: Identificação alfabética da série (ex.: A, B).
  + **Turno**: Turno em que a série ocorre, usando um ENUM para restringir os valores possíveis a 'Manhã', 'Tarde' ou 'Noite'.
  + **Linha 12-13**: Adiciona uma **chave estrangeira composta** (Serie, Letra, Turno) referenciando a tabela *Dim\_Turmas* para garantir consistência entre as séries e turmas.

**Criação da Tabela Fato\_Turmas:**

* **Linhas 16 a 24:**  
  Cria a tabela *Fato\_Turmas*, que representa os dados factuais relacionados às turmas:
  + **Id\_Fato**: Chave primária da tabela.
  + **Id\_Turma**: Chave estrangeira referenciando a tabela *Dim\_Turmas*.
  + **Total\_Alunos**: Quantidade total de alunos na turma.
  + **Total\_Presencas**: Soma total de presenças registradas.
  + **Total\_Faltas**: Soma total de faltas registradas.
  + **FOREIGN KEY (Id\_Turma)**: Cria a referência entre *Id\_Turma* e *Dim\_Turmas*.

**Criação da Tabela Fato\_Notas**:

* **Linhas 26 a 38:**  
  Cria a tabela *Fato\_Notas*, responsável por armazenar as informações sobre notas dos alunos:
  + **Id\_Fato**: Chave primária da tabela.
  + **Id\_Aluno**: Chave estrangeira referenciando *Dim\_Alunos*.
  + **Id\_Disciplina**: Chave estrangeira referenciando *Dim\_Disciplinas*.
  + **Id\_Notas**: Chave estrangeira referenciando *Dim\_Notas*.
  + **Nota\_Primeiro\_Semestre**: Nota do primeiro semestre.
  + **Nota\_Segundo\_Semestre**: Nota do segundo semestre.
  + **Nota\_Trabalho\_Academico**: Nota de trabalhos acadêmicos.
  + **Nota\_Final**: Nota final calculada com base nas demais notas.
  + **FOREIGN KEYS**: Cria as relações com *Dim\_Alunos*, *Dim\_Disciplinas* e *Dim\_Notas*.

**Explicação do Output:**

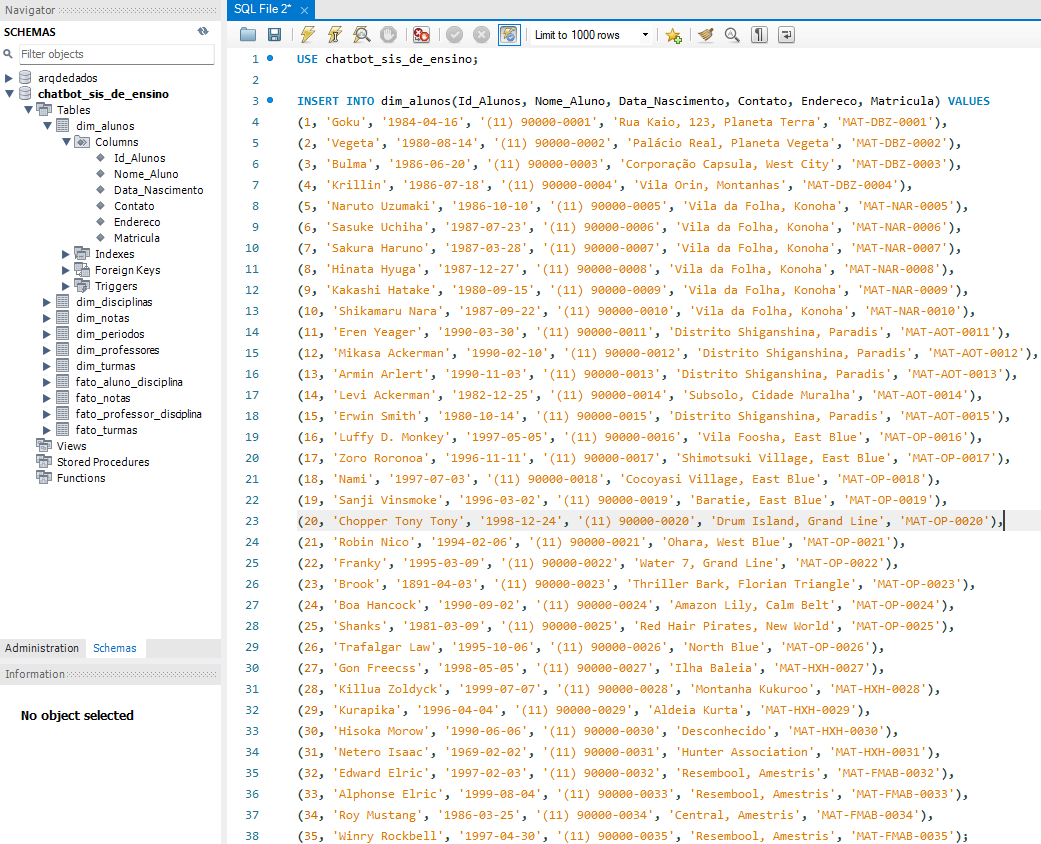
* **Linha 1: USE chatbot\_sis\_de\_ensino**  
  Banco de dados chatbot\_sis\_de\_ensino selecionado com sucesso.
* **Linha 2: ALTER TABLE Dim\_Turmas ADD CONSTRAINT**  
  A restrição de unicidade idx\_serie\_letra\_turno foi adicionada com sucesso.
* **Linha 3: CREATE TABLE Dim\_Periodos**  
  A tabela Dim\_Periodos foi criada com sucesso, incluindo a chave estrangeira composta que referência Dim\_Turmas.
* **Linha 4: CREATE TABLE Fato\_Turmas**  
  A tabela Fato\_Turmas foi criada com sucesso, com a chave estrangeira referenciando Dim\_Turmas.
* **Linha 5: CREATE TABLE Fato\_Notas**  
  A tabela Fato\_Notas foi criada com sucesso, com chaves estrangeiras referenciando Dim\_Alunos, Dim\_Disciplinas e Dim\_Notas.

Todas as instruções foram executadas sem erros. As tabelas foram criadas corretamente com suas respectivas chaves primárias, estrangeiras e restrições de unicidade.

# Inserindo dados nas tabelas:

Agora iniciamos mais uma etapa da montagem do nosso banco de dados, agora daremos inicio as inserções de dados e para que seja seguro iremos seguir com uma abordagem cautelosa, afim de nos assegurar de possíveis erros que possam surgir durante a inserção dos dados.

## 3.1. Inserção de Dados na Tabela Dim\_Alunos:

  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
1. Descrição Geral:

* Neste tópico, abordamos o processo de inserção de dados na tabela dim\_alunos, parte integrante do banco de dados chatbot\_sis\_de\_ensino. A tabela dim\_alunos armazena informações dos alunos, incluindo colunas como Id\_Alunos, Nome\_Aluno, Data\_Nascimento, Contato, Endereco e Matricula. O processo de inserção seguiu os seguintes passos:
* Preparação dos Dados:  
  Foram utilizados 35 registros baseados em personagens de animes variados, contendo:
  + Nome do aluno: Nome fictício do personagem.  
    Data de nascimento: Datas distribuídas entre os anos de 1980 e 2000.
  + Contato: Padrão de telefone fictício no formato (11) 90000-xxxx.
  + Endereço: Criado com referência ao universo dos personagens.
  + Matrícula: Código único gerado com base na sigla do anime e um identificador sequencial.
* Reorganização da Estrutura da Tabela:  
  Antes da inserção, a coluna *Nome\_Aluno* foi reorganizada para ficar ao lado da coluna *Id\_Alunos*, facilitando a visualização lógica e estrutural dos dados.

2. Script SQL de Inserção:

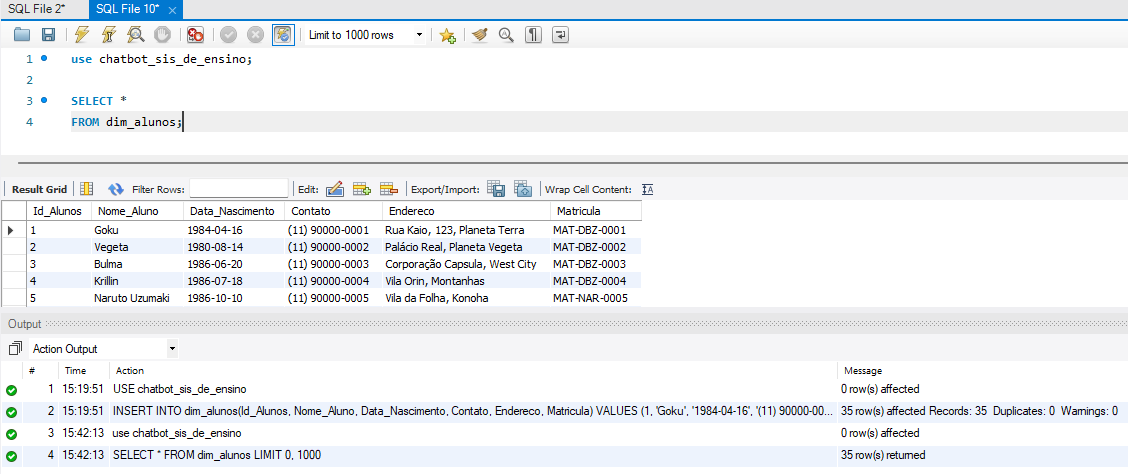
* Para popular a tabela dim\_alunos, foi utilizado o comando *INSERT INTO*, conforme apresentado abaixo:
* INSERT INTO dim\_alunos (Id\_Alunos, Nome\_Aluno, Data\_Nascimento, Contato, Endereco, Matricula)
* VALUES
* (1, 'Goku', '1984-04-16', '(11) 90000-0001', 'Rua Kaio, 123, Planeta Terra', 'MAT-DBZ-0001'),
* (2, 'Vegeta', '1980-08-14', '(11) 90000-0002', 'Palácio Real, Planeta Vegeta', 'MAT-DBZ-0002'),
* ...
* (35, 'Winry Rockbell', '1998-04-30', '(11) 90000-0035', 'Resembool, Amestris', 'MAT-FMAB-0035’);

3. Execução e Output:

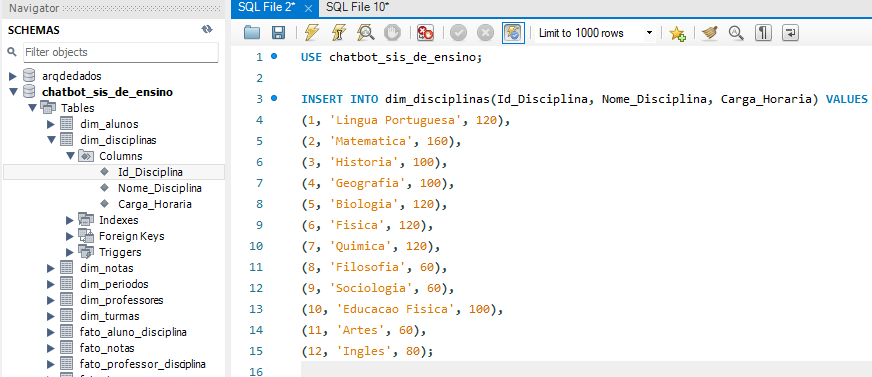
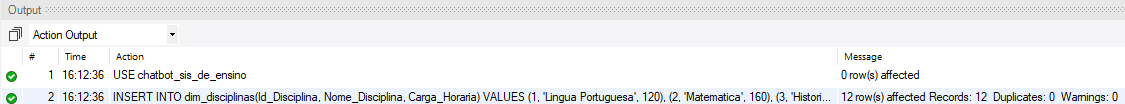
* Comando Utilizado:  
  Após a execução do script de inserção, os dados foram inseridos com sucesso na tabela. O output apresentou a mensagem: 35 row(s) affected Records: 35 Duplicates: 0 Warnings: 0

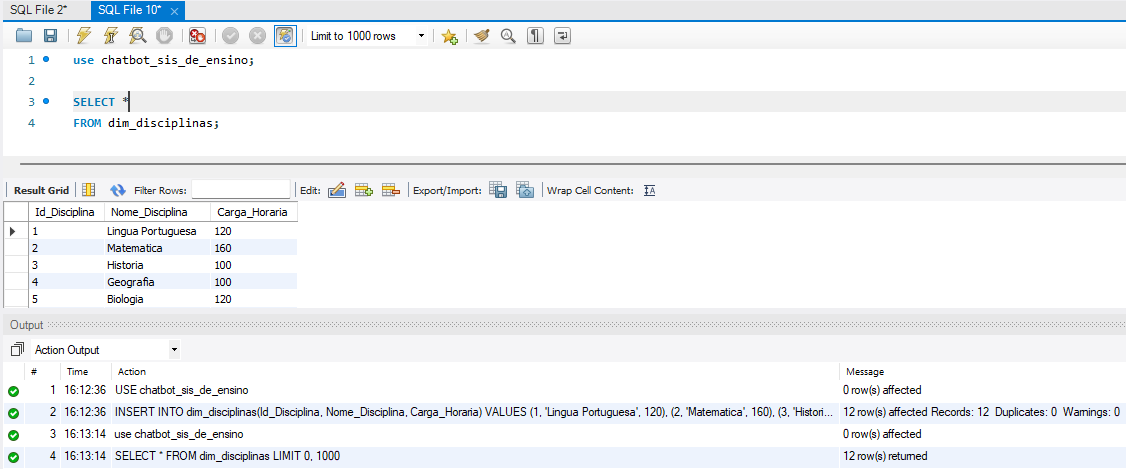
Isso confirma que:

* + 35 registros foram inseridos na tabela.
  + Nenhuma duplicação ocorreu.
  + Não houve mensagens de aviso ou erro.
* Visualização dos Dados:  
  A tabela dim\_alunos foi consultada utilizando o comando *SELECT \* FROM dim\_alunos;*, permitindo validar a inserção e confirmar a integridade dos registros.



## 3.2. Inserção de Dados na Tabela Dim\_Disciplinas:





A tabela Dim\_Disciplinas foi criada com o objetivo de armazenar informações referentes às disciplinas do ensino médio, como nome da disciplina e sua respectiva carga horária. Esta tabela é uma dimensão que serve como referência em análises acadêmicas e está organizada em três colunas:

* **Id\_Disciplina**: Chave primária da tabela, identificador único e incremental para cada disciplina.
* **Nome\_Disciplina**: Nome da disciplina, utilizado para identificar o conteúdo curricular de forma textual.
* **Carga\_Horaria**: Quantidade total de horas dedicadas à disciplina durante o ano letivo.

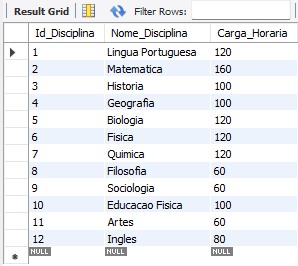
**Script de Inserção dos Dados:**

Os dados foram inseridos na tabela utilizando o comando SQL **INSERT INTO**, conforme o exemplo:

* INSERT INTO dim\_disciplinas(Id\_Disciplina, Nome\_Disciplina, Carga\_Horaria) VALUES
* (1, 'Lingua Portuguesa', 120),
* (2, 'Matematica', 160),
* (3, 'Historia', 100),
* (4, 'Geografia', 100),
* (5, 'Biologia', 120),
* (6, 'Fisica', 120),
* (7, 'Quimica', 120),
* (8, 'Filosofia', 60),
* (9, 'Sociologia', 60),
* (10, 'Educacao Fisica', 100),
* (11, 'Artes', 60),
* (12, 'Ingles', 80);

**Resultado da Execução:**

* A execução do comando apresentou sucesso, conforme o **output** capturado. Foram afetados **12 registros**, sem duplicações ou erros de inserção, como indicado na mensagem:
* "12 row(s) affected. Records: 12 Duplicates: 0 Warnings: 0".

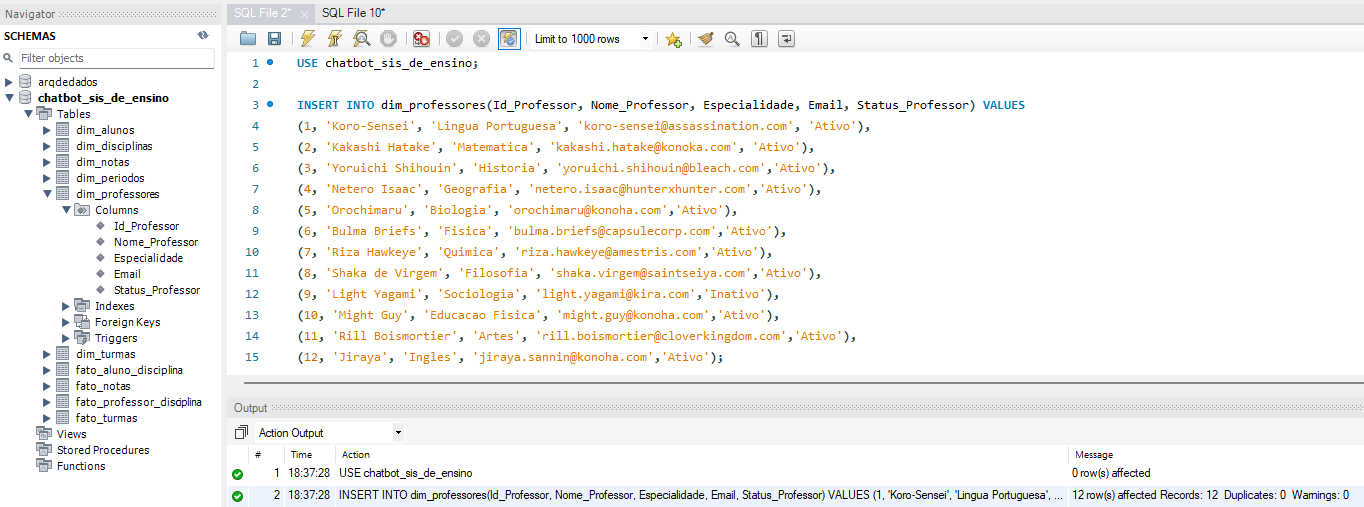
**Consulta de Validação:**

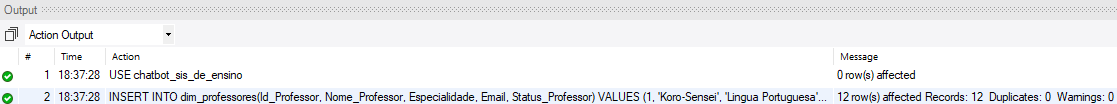
* Para validar os dados inseridos, foi utilizada a seguinte query SQL:
* SELECT \* FROM dim\_disciplinas;
* O resultado retornou com sucesso os **12 registros inseridos**, conforme apresentado no grid de resultados. A tabela agora contém as seguintes entradas:

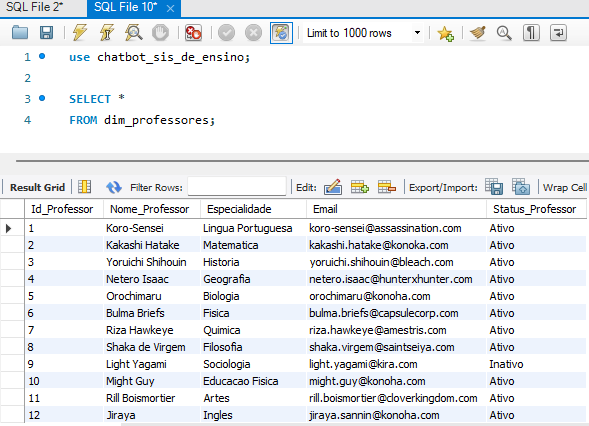
A tabela *Dim\_Disciplinas* foi populada corretamente com 12 disciplinas que representam o currículo do ensino médio. Cada disciplina está devidamente identificada por um **Id\_Disciplina** único e possui sua carga horária específica.

Esta estrutura será essencial para relacionamentos com tabelas de fatos (como presença, notas ou matrículas) em análises futuras no sistema acadêmico.

## 3.3. Inserção de Dados na Tabela Dim\_Professores:







**Contextualização:**

A tabela dim\_professores foi criada com o propósito de armazenar informações detalhadas dos professores responsáveis por cada disciplina no modelo acadêmico. Essa tabela compõe o conjunto de tabelas dimensionais do banco de dados *chatbot\_sis\_de\_ensino* e serve como referência direta para identificar os professores e suas respectivas especialidades.

**Estrutura da Tabela:**

A estrutura da tabela dim\_professores é composta pelas seguintes colunas:

* **Id\_Professor**: Identificador único do professor. Chave primária.
* **Nome\_Professor**: Nome completo do professor.
* **Especialidade**: Disciplina que o professor leciona.
* **Email**: Endereço eletrônico do professor.
* **Status\_Professor**: Status do professor (Ativo ou Inativo).

A chave primária é definida na coluna Id\_Professor para garantir unicidade e integridade dos registros.

**Inserção de Dados:**

A inserção foi realizada utilizando o comando INSERT INTO com os dados de 12 professores, onde cada professor foi associado a uma disciplina da tabela dim\_disciplinas. Os dados inseridos incluem nomes inspirados em mestres de animes, especialidades acadêmicas, endereços de e-mail e o status de cada professor.

**Comando SQL Utilizado:**USE chatbot\_sis\_de\_ensino;   
INSERT INTO dim\_professores(Id\_Professor, Nome\_Professor, Especialidade, Email, Status\_Professor) VALUES   
(1, 'Koro-Sensei', 'Lingua Portuguesa', 'koro-sensei@assassination.com', 'Ativo'),   
(2, 'Kakashi Hatake', 'Matematica', 'kakashi.hatake@konoka.com', 'Ativo'),   
(3, 'Yoruichi Shihouin', 'Historia', 'yoruichi.shihouin@bleach.com', 'Ativo'),   
(4, 'Netero Isaac', 'Geografia', 'netero.isaac@hunterxhunter.com', 'Ativo'),   
(5, 'Orochimaru', 'Biologia', 'orochimaru@konoka.com', 'Ativo'),   
(6, 'Bulma Briefs', 'Fisica', 'bulma.briefs@capsulecorp.com', 'Ativo'),   
(7, 'Riza Hawkeye', 'Quimica', 'riza.hawkeye@amestris.com', 'Ativo'),   
(8, 'Shaka de Virgem', 'Filosofia', 'shaka.virgem@saintseiya.com', 'Ativo'),   
(9, 'Light Yagami', 'Sociologia', 'light.yagami@kira.com', 'Inativo'),   
(10, 'Might Guy', 'Educacao Fisica', 'might.guy@konoka.com', 'Ativo'),  
(11, 'Rill Boismortier', 'Artes', 'rill.boismortier@cloverkingdom.com', 'Ativo'),   
(12, 'Jiraya', 'Ingles', 'jiraya.sannin@konoka.com', 'Ativo');

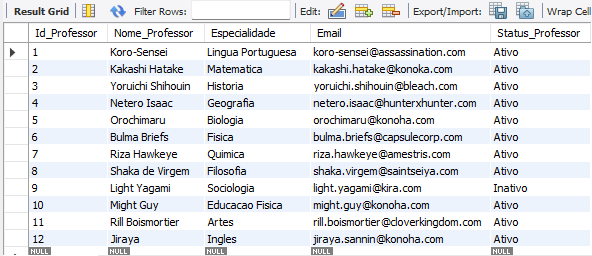
**Output da Inserção:**

O log de execução da inserção foi bem-sucedido, conforme exibido no painel **Action Output** do MySQL Workbench:

* **12 registros inseridos**.
* **0 duplicatas** ou **erros reportados**.

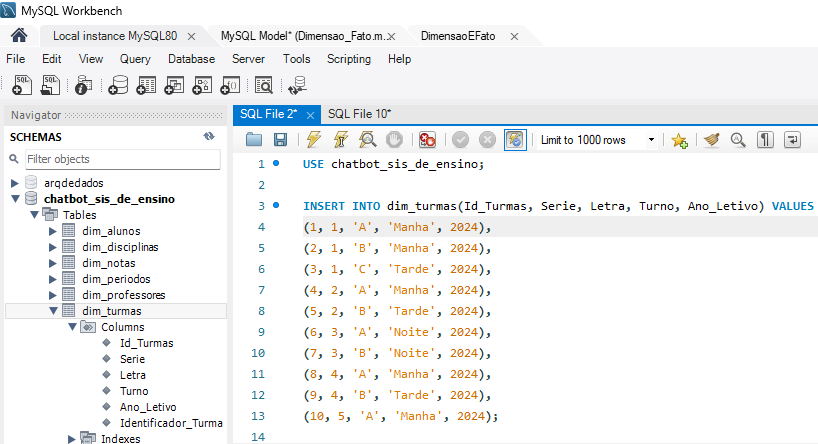
O comando SELECT foi executado posteriormente para verificar os dados inseridos.

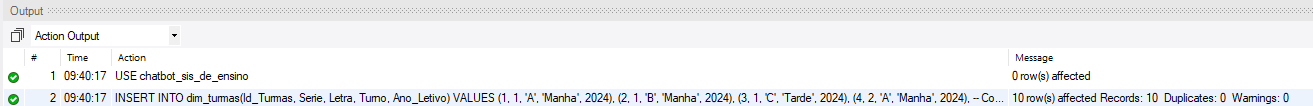
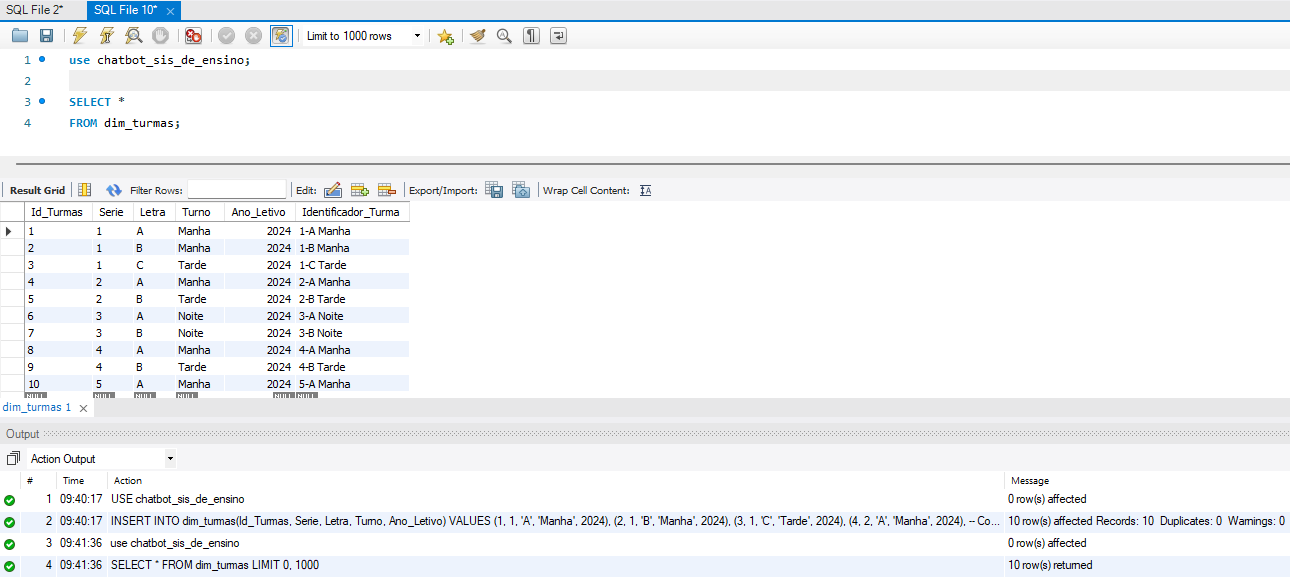
**Consulta SQL:**  
SELECT \*   
FROM dim\_professores;

Resultado:  


A tabela dim\_professores foi populada com êxito, apresentando dados completos e consistentes para cada professor, suas especialidades e status. Esses registros servirão como base para associações com outras tabelas de fato, como fato\_professor\_disciplina e fato\_aluno\_disciplina. O processo de inserção foi documentado e validado.

## Inserção de Dados na Tabela Dim\_Turmas:



**1. Descrição da Tabela Dim\_Turmas:**

A tabela Dim\_Turmas foi criada para armazenar informações relacionadas às turmas de ensino médio, como série, turno, letra da turma e o ano letivo. Essa tabela possui uma coluna gerada automaticamente (*Identificador\_Turma*) com base em uma concatenação das colunas Serie, Letra e Turno.

A seguir, estão detalhadas as colunas da tabela:

* **Id\_Turmas**: Identificador único da turma (chave primária).
* **Serie**: Número da série (ex.: 1, 2, 3).
* **Letra**: Letra identificadora da turma (ex.: A, B, C).
* **Turno**: Turno da turma, com valores fixos (Manha, Tarde, Noite).
* **Ano\_Letivo**: Ano letivo associado à turma (ex.: 2024).
* **Identificador\_Turma**: Coluna gerada automaticamente por meio da expressão CONCAT(Serie, '-', Letra, ' ', Turno) e armazenada (STORED).

**2. Script de Inserção de Dados:**

O seguinte script SQL foi utilizado para inserir os registros na tabela Dim\_Turmas:

USE chatbot\_sis\_de\_ensino;

INSERT INTO dim\_turmas(Id\_Turmas, Serie, Letra, Turno, Ano\_Letivo) VALUES

(1, 1, 'A', 'Manha', 2024),  
(2, 1, 'B', 'Manha', 2024),  
(3, 1, 'C', 'Tarde', 2024),  
(4, 2, 'A', 'Manha', 2024),  
(5, 2, 'B', 'Tarde', 2024),  
(6, 3, 'A', 'Noite', 2024),  
(7, 3, 'B', 'Noite', 2024),  
(8, 4, 'A', 'Manha', 2024),  
(9, 4, 'B', 'Tarde', 2024),  
(10, 5, 'A', 'Manha', 2024);

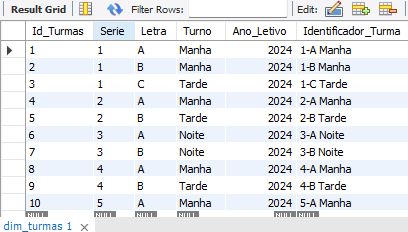
**3. Execução do Script e Output:**

A execução do script ocorreu com sucesso, conforme mostrado no log de execução:

1. **Comando Executado**:
   * O script INSERT INTO dim\_turmas adicionou **10 registros** na tabela.
2. **Mensagem de Output**:
   * 10 row(s) affected: Dez linhas foram inseridas sem duplicatas ou warnings.
   * A tabela não permitiu duplicação devido à restrição *UNIQUE* sobre as colunas *Serie*, *Letra* e *Turno*.

**4. Consulta para Verificação dos Dados:**

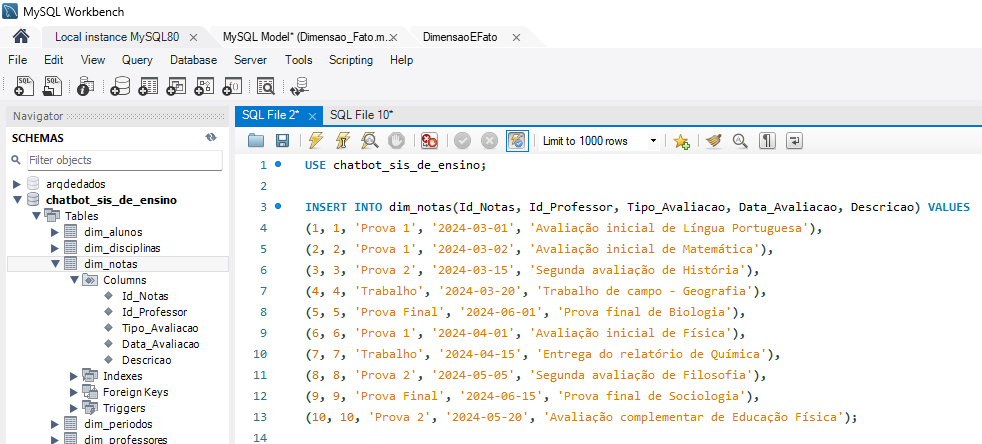
Foi realizada a consulta SQL para visualizar os registros na tabela:  
SELECT \*   
FROM dim\_turmas;

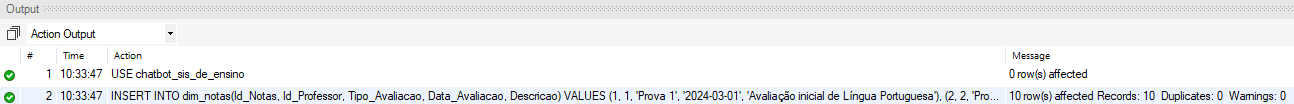
  
  
  
  
  
  
  
  
  
  
  
  
**5. Observações Importantes:**

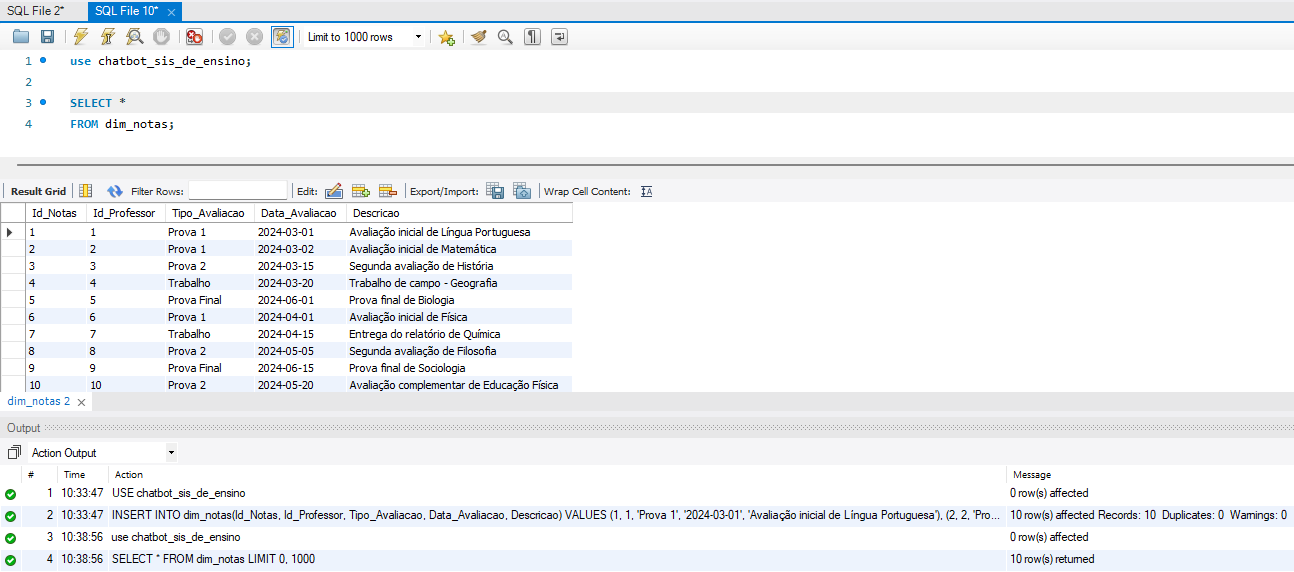
1. **Coluna Gerada**:
   * A coluna *Identificador\_Turma* foi preenchida automaticamente como esperado, graças à definição GENERATED ALWAYS AS na criação da tabela.
2. **Restrição de Unicidade**:
   * A restrição *UNIQUE* sobre a combinação *Serie*, *Letra* e *Turno* garantiu que não houvesse entradas duplicadas.
3. **Estrutura e Integridade**:
   * A inserção foi validada e os dados correspondem ao planejamento da tabela.

A tabela *Dim\_Turmas* foi populada com sucesso, e a consulta mostrou que todos os registros estão corretos, seguindo a lógica de séries, turnos e letras definidas no script de inserção. O uso de uma coluna gerada para *Identificador\_Turma* simplifica a identificação das turmas no sistema, garantindo uma estrutura organizada e eficiente.

## Inserção de Dados na Tabela Dim\_Notas:







**1. Descrição da Tabela Dim\_Notas:**

A tabela dim\_notas é responsável por armazenar informações referentes às notas e avaliações aplicadas no sistema acadêmico. A estrutura da tabela está organizada da seguinte forma:

* **Id\_Notas**: Identificador único da nota (chave primária).
* **Id\_Professor**: Chave estrangeira referenciando o identificador do professor na tabela dim\_professores.
* **Tipo\_Avaliacao**: Tipo da avaliação aplicada, como Prova 1, Prova Final, Trabalho.
* **Data\_Avaliacao**: Data em que a avaliação ocorreu.
* **Descricao**: Detalhes adicionais sobre a avaliação, como seu objetivo ou conteúdo específico.

**2. Script SQL Utilizado:**

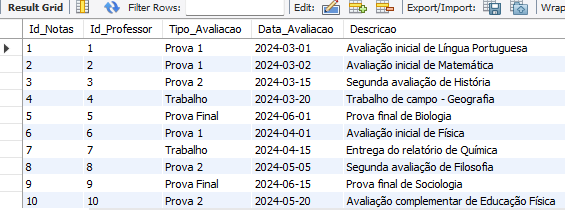
USE chatbot\_sis\_de\_ensino;  
INSERT INTO dim\_notas(Id\_Notas, Id\_Professor, Tipo\_Avaliacao, Data\_Avaliacao, Descricao) VALUES

(1, 1, 'Prova 1', '2024-03-01', 'Avaliação inicial de Língua Portuguesa'),  
(2, 2, 'Prova 1', '2024-03-02', 'Avaliação inicial de Matemática'),  
(3, 3, 'Prova 2', '2024-03-15', 'Segunda avaliação de História'),  
(4, 4, 'Trabalho', '2024-03-20', 'Trabalho de campo - Geografia'),  
(5, 5, 'Prova Final', '2024-04-01', 'Prova final de Biologia'),  
(6, 6, 'Prova 1', '2024-04-01', 'Avaliação inicial de Física'),  
(7, 7, 'Trabalho', '2024-04-15', 'Entrega do relatório de Química'),  
(8, 8, 'Prova 2', '2024-05-05', 'Segunda avaliação de Filosofia'),  
(9, 9, 'Prova Final', '2024-05-15', 'Prova final de Sociologia'),  
(10, 10, 'Prova 2', '2024-05-20', 'Avaliação complementar de Educação Física');

**3. Resultados da Inserção:**

A execução do script de inserção apresentou **10 registros inseridos com sucesso** na tabela dim\_notas, conforme mostrado no log de saída:

* **Ação**: INSERT INTO dim\_notas
* **Linhas Afetadas**: 10
* **Mensagens**: Nenhuma duplicação ou advertência foi registrada.

1. **Dados Inseridos:**

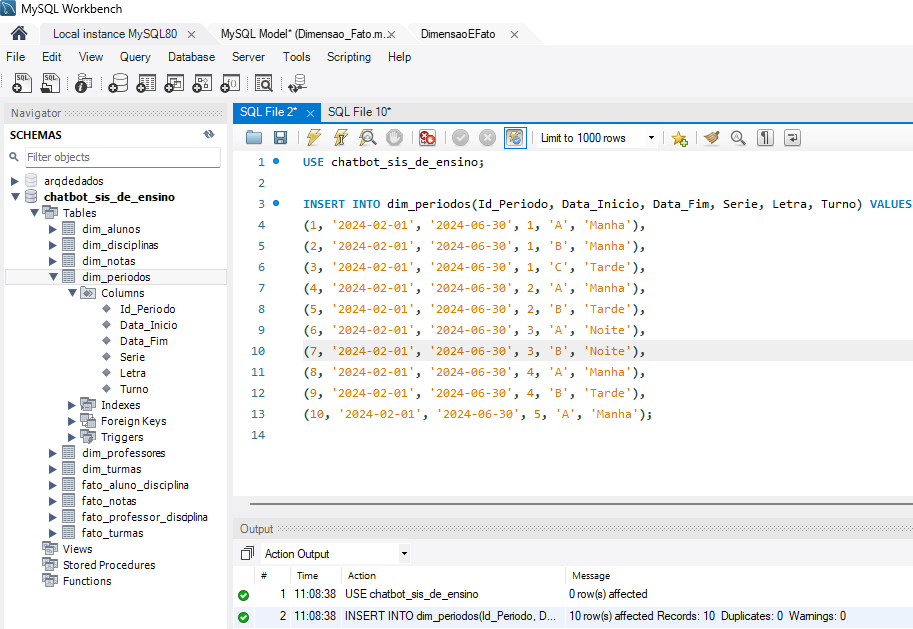
A consulta realizada utilizando o comando:  
SELECT \*  
FROM dim\_notas;

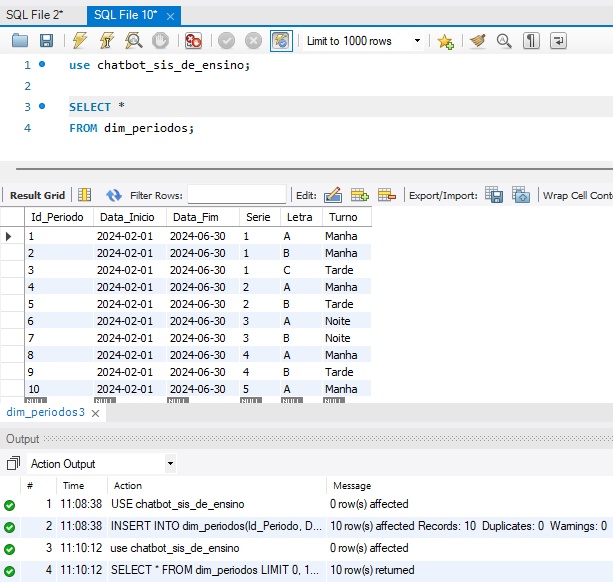
1. **Observações:**

* A coluna Id\_Professor conecta os registros da tabela dim\_notas com a tabela dim\_professores, garantindo a integridade relacional.
* A tabela foi devidamente populada, seguindo as exigências do sistema acadêmico.
* A estrutura da tabela foi validada com sucesso, sem problemas de inconsistência ou duplicação de dados.

A tabela dim\_notas está agora completa e pronta para ser utilizada em consultas e relatórios futuros dentro do sistema acadêmico chatbot\_sis\_de\_ensino.

## 3.6. Inserção de Dados na Tabela Dim\_Periodos:



  
  
1. Descrição da Tabela Dim\_Periodos:  
A tabela dim\_periodos tem como objetivo armazenar informações sobre os períodos letivos de turmas escolares, contendo colunas para representar intervalos de datas, séries, letras e turnos das turmas. Essa tabela é vinculada à tabela dim\_turmas por meio de uma chave estrangeira composta, que referência a combinação (Serie, Letra, Turno).

**2. Estrutura da Tabela:**A tabela foi criada com as seguintes colunas:

| **Nome da Coluna** | **Tipo de Dados** | **Descrição** | **Restrições** |
| --- | --- | --- | --- |
| Id\_Periodo | INT | Identificador único do período | PRIMARY KEY, NOT NULL |
| Data\_Inicio | DATE | Data de início do período letivo | NOT NULL |
| Data\_Fim | DATE | Data de término do período letivo | NOT NULL |
| Serie | INT | Série da turma | NOT NULL |
| Letra | VARCHAR(1) | Letra que identifica a turma | NOT NULL |
| Turno | ENUM | Turno da turma (Manhã, Tarde, Noite) | NOT NULL |

**3. Script de Inserção:**

O script SQL utilizado para inserir os registros na tabela foi o seguinte:  
USE chatbot\_sis\_de\_ensino;

INSERT INTO dim\_periodos (Id\_Periodo, Data\_Inicio, Data\_Fim, Serie, Letra, Turno) VALUES

(1, '2024-02-01', '2024-06-30', 1, 'A', 'Manha'),  
(2, '2024-02-01', '2024-06-30', 1, 'B', 'Manha'),  
(3, '2024-02-01', '2024-06-30', 1, 'C', 'Tarde'),  
(4, '2024-02-01', '2024-06-30', 2, 'A', 'Manha'),  
(5, '2024-02-01', '2024-06-30', 2, 'B', 'Tarde'),  
(6, '2024-02-01', '2024-06-30', 3, 'A', 'Noite'),  
(7, '2024-02-01', '2024-06-30', 3, 'B', 'Noite'),  
(8, '2024-02-01', '2024-06-30', 4, 'A', 'Manha'),  
(9, '2024-02-01', '2024-06-30', 4, 'B', 'Tarde'),  
(10, '2024-02-01', '2024-06-30', 5, 'A', 'Manha');

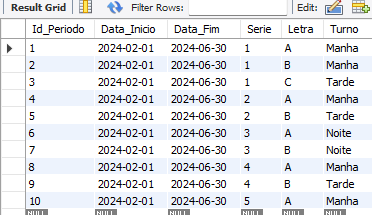
**4. Output:**

A execução do script resultou nos seguintes logs:

| **Time** | **Action** | **Message** |
| --- | --- | --- |
| 11:08:38 | USE chatbot\_sis\_de\_ensino | 0 row(s) affected |
| 11:08:38 | INSERT INTO dim\_periodos (Id\_Periodo, Data\_Inicio, Data\_Fim, ...) | 10 row(s) affected, 0 duplicates, 0 warnings |
| 11:10:12 | SELECT \* FROM dim\_periodos LIMIT 0, 1000 | 10 row(s) returned |

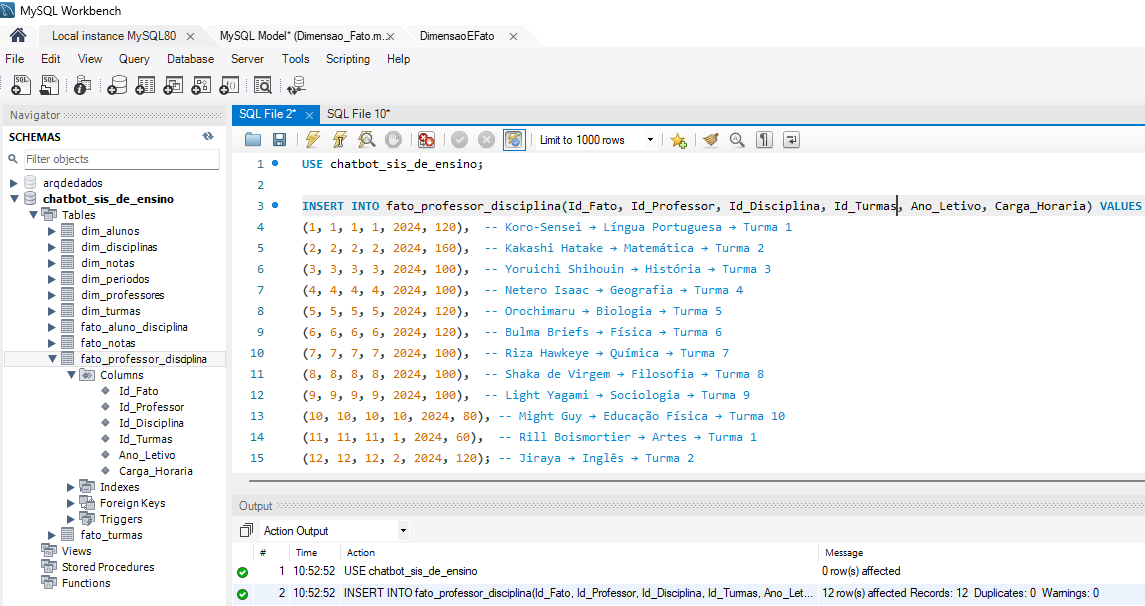
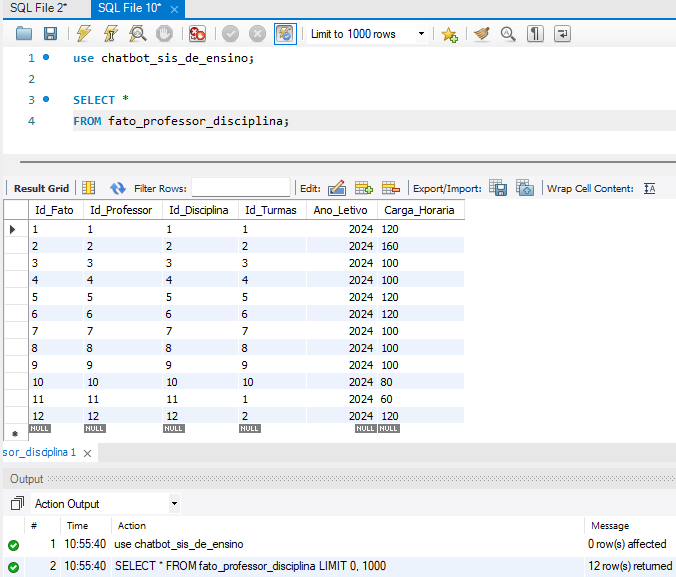
**5. Resultado da Consulta:**

Os registros da tabela dim\_periodos foram inseridos corretamente e podem ser visualizados abaixo:



A tabela dim\_periodos foi devidamente preenchida com 10 registros representando diferentes períodos letivos, variando datas, séries, letras e turnos. Todas as restrições de integridade, incluindo a chave estrangeira composta com dim\_turmas, foram atendidas com sucesso.

## 3.7. Inserção de Dados na Tabela Fato\_Professor\_Disciplina:

**1. Objetivo:**

Registrar o relacionamento entre **professores**, **disciplinas** e **turmas**, com a respectiva **carga horária** associada ao ano letivo.

**2. Estrutura da Tabela:**

A tabela possui os seguintes campos:

* **Id\_Fato**: Identificador único da tabela fato.
* **Id\_Professor**: Chave estrangeira referenciando a tabela **dim\_professores**.
* **Id\_Disciplina**: Chave estrangeira referenciando a tabela **dim\_disciplinas**.
* **Id\_Turmas**: Chave estrangeira referenciando a tabela **dim\_turmas**.
* **Ano\_Letivo**: Ano letivo correspondente.
* **Carga\_Horaria**: Total de horas associadas ao professor, disciplina e turma.

**3. Código Utilizado:**

USE chatbot\_sis\_de\_ensino;

INSERT INTO fato\_professor\_disciplina(Id\_Fato, Id\_Professor, Id\_Disciplina, Id\_Turmas, Ano\_Letivo, Carga\_Horaria) VALUES

(1, 1, 1, 1, 2024, 120), -- Koro-Sensei → Língua Portuguesa → Turma 1  
(2, 2, 2, 2, 2024, 160), -- Kakashi Hatake → Matemática → Turma 2  
(3, 3, 3, 3, 2024, 100), -- Yoruichi Shihouin → História → Turma 3  
(4, 4, 4, 4, 2024, 100), -- Netero Isaac → Geografia → Turma 4  
(5, 5, 5, 5, 2024, 120), -- Orochimaru → Biologia → Turma 5  
(6, 6, 6, 6, 2024, 120), -- Bulma Briefs → Física → Turma 6  
(7, 7, 7, 7, 2024, 100), -- Riza Hawkeye → Química → Turma 7  
(8, 8, 8, 8, 2024, 100), -- Shaka de Virgem → Filosofia → Turma 8  
(9, 9, 9, 9, 2024, 80), -- Light Yagami → Sociologia → Turma 9  
(10, 10, 10, 10, 2024, 100), -- Might Guy → Educação Física → Turma 10  
(11, 11, 11, 1, 2024, 60), -- Rill Boismortier → Artes → Turma 1  
(12, 12, 12, 2, 2024, 120); -- Jiraya → Inglês → Turma 2

**4. Resultado da Consulta:**

SELECT \*   
FROM fato\_professor\_disciplina;

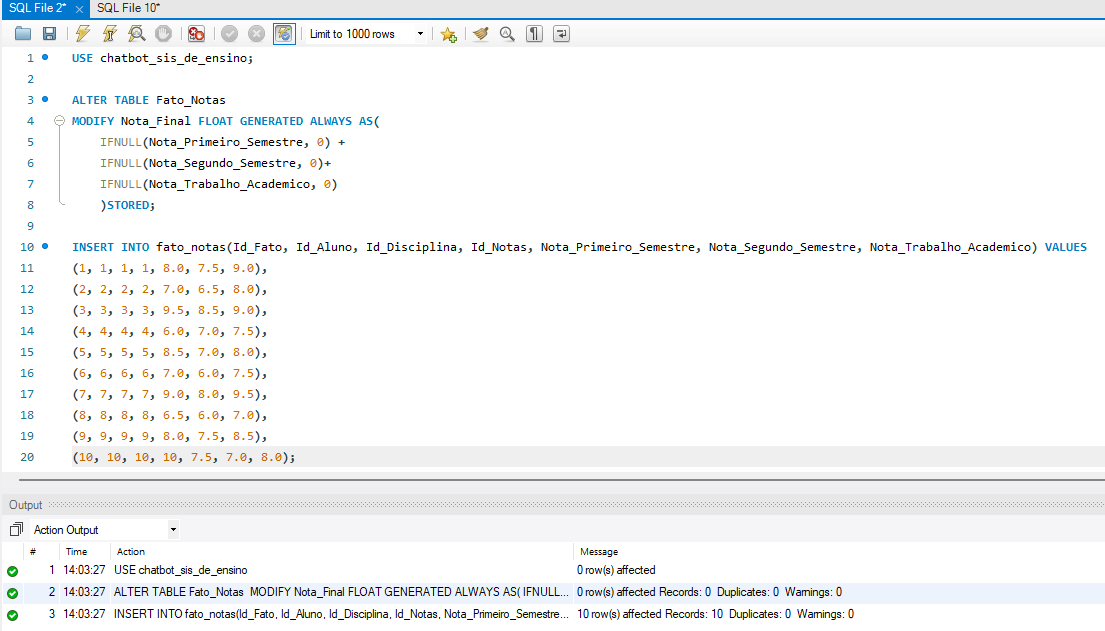
**5. Saída da Consulta:**

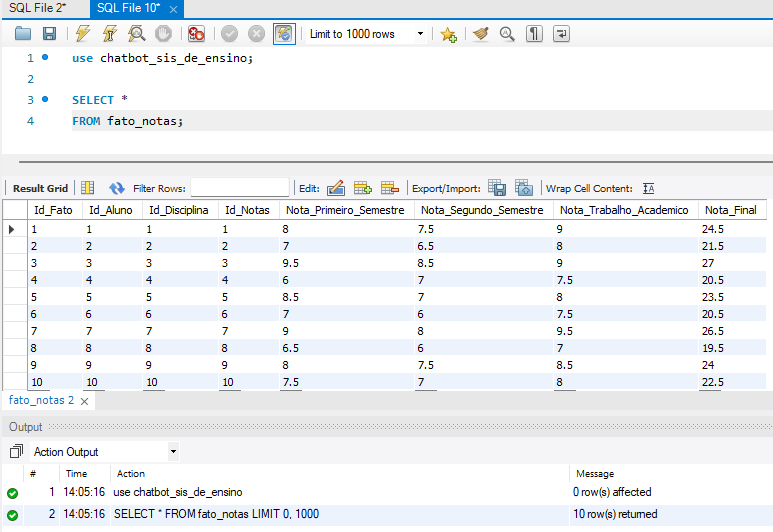
| **Id\_Fato** | **Id\_Professor** | **Id\_Disciplina** | **Id\_Turmas** | **Ano\_Letivo** | **Carga\_Horaria** |
| --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 2024 | 120 |
| 2 | 2 | 2 | 2 | 2024 | 160 |
| 3 | 3 | 3 | 3 | 2024 | 100 |
| 4 | 4 | 4 | 4 | 2024 | 100 |
| 5 | 5 | 5 | 5 | 2024 | 120 |
| 6 | 6 | 6 | 6 | 2024 | 120 |
| 7 | 7 | 7 | 7 | 2024 | 100 |
| 8 | 8 | 8 | 8 | 2024 | 100 |
| 9 | 9 | 9 | 9 | 2024 | 80 |
| 10 | 10 | 10 | 10 | 2024 | 100 |
| 11 | 11 | 11 | 1 | 2024 | 60 |
| 12 | 12 | 12 | 2 | 2024 | 120 |

**Resumo:**

* **Total de Registros Inseridos**: 12.
* **Carga Horária**: As cargas horárias variam entre **60** e **160** horas, dependendo da disciplina e da turma.
* **Relacionamento**:
  + Cada **Id\_Professor** faz referência a um registro da tabela **dim\_professores**.
  + Cada **Id\_Disciplina** corresponde a uma disciplina cadastrada na **dim\_disciplinas**.
  + O **Id\_Turmas** está associado às turmas registradas na tabela **dim\_turmas**.

Podemos dizer que a inserção dos dados foi realizada com sucesso, e a consulta retornou os **12 registros esperados** sem duplicidade ou erros.

3.8. Inserção de Dados na Tabela Fato\_Notas:  
  




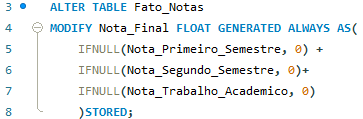
**1. Estrutura da Tabela fato\_notas:**

A tabela *fato\_notas* foi projetada com os seguintes campos:

| **Coluna** | **Tipo de Dados** | **Descrição** |
| --- | --- | --- |
| **Id\_Fato** | INT | Identificador único para cada registro (PK). |
| **Id\_Aluno** | INT | Chave estrangeira para identificar o aluno (dim\_alunos). |
| **Id\_Disciplina** | INT | Chave estrangeira para a disciplina (dim\_disciplinas). |
| **Id\_Notas** | INT | Chave estrangeira para a nota associada (dim\_notas). |
| **Nota\_Primeiro\_Semestre** | FLOAT | Nota obtida no primeiro semestre. |
| **Nota\_Segundo\_Semestre** | FLOAT | Nota obtida no segundo semestre. |
| **Nota\_Trabalho\_Academico** | FLOAT | Nota obtida em trabalhos acadêmicos. |
| **Nota\_Final** | FLOAT | **Coluna gerada automaticamente** com a soma das notas. |

**Nota:** A coluna Nota\_Final foi alterada para ser **calculada automaticamente** usando um *ALTER TABLE*, permitindo que a soma das notas fosse gerada com base em *Nota\_Primeiro\_Semestre*, *Nota\_Segundo\_Semestre* e *Nota\_Trabalho\_Academico*.

**2. Script SQL Utilizado:**

**Modificação da Coluna Nota\_Final**:

ALTER TABLE Fato\_Notas  
MODIFY Nota\_Final FLOAT GENERATED ALWAYS AS (  
 IFNULL(Nota\_Primeiro\_Semestre, 0) +  
 IFNULL(Nota\_Segundo\_Semestre, 0) +  
 IFNULL(Nota\_Trabalho\_Academico, 0)   
) STORED;

**3. Script de Inserção de Dados**:

INSERT INTO fato\_notas(Id\_Fato, Id\_Aluno, Id\_Disciplina, Id\_Notas, Nota\_Primeiro\_Semestre, Nota\_Segundo\_Semestre, Nota\_Trabalho\_Academico) VALUES   
(1, 1, 1, 1, 8.0, 7.5, 9.0),  
(2, 2, 2, 2, 7.0, 6.5, 8.0),  
(3, 3, 3, 3, 9.5, 8.5, 9.0),  
(4, 4, 4, 4, 6.0, 7.0, 7.5),  
(5, 5, 5, 5, 8.5, 7.0, 8.0),  
(6, 6, 6, 6, 7.0, 6.0, 9.5),  
(7, 7, 7, 7, 9.0, 8.0, 9.5),  
(8, 8, 8, 8, 6.5, 6.0, 9.0),  
(9, 9, 9, 9, 8.0, 7.5, 8.5),  
(10, 10, 10, 10, 7.5, 7.0, 8.0);

**4. Resultado da Consulta:**

**Consulta Executada**:  
SELECT \* FROM fato\_notas;

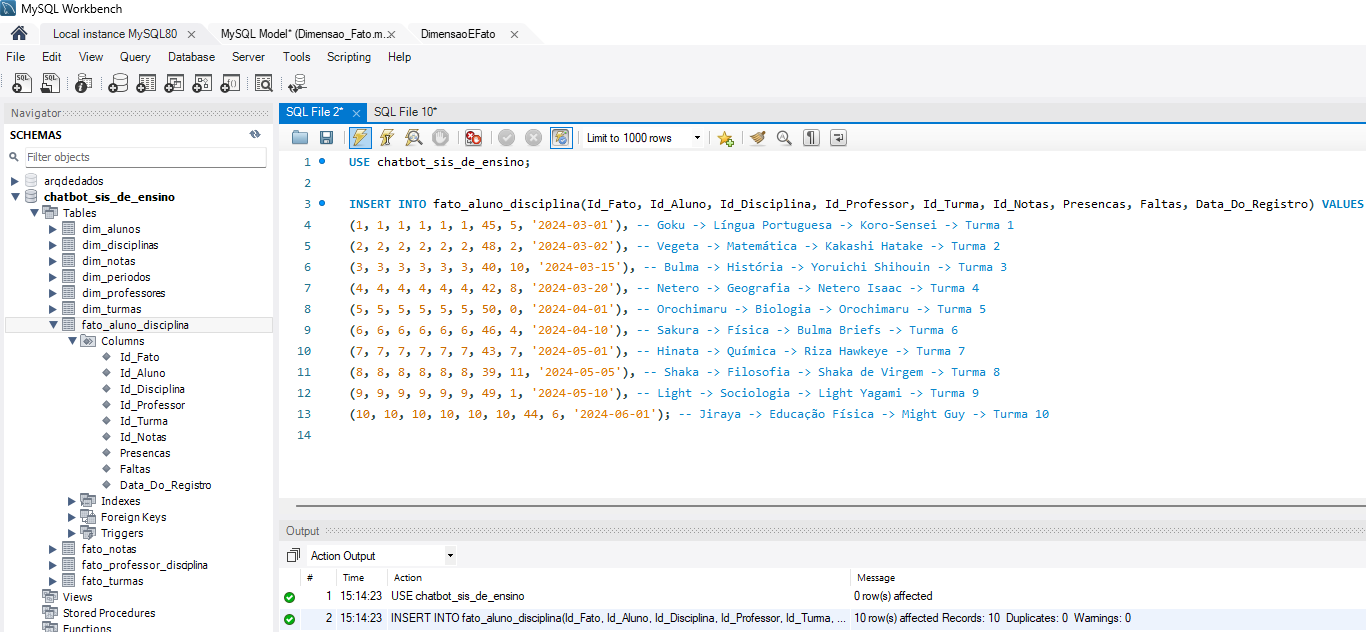
**Resultado Retornado**:

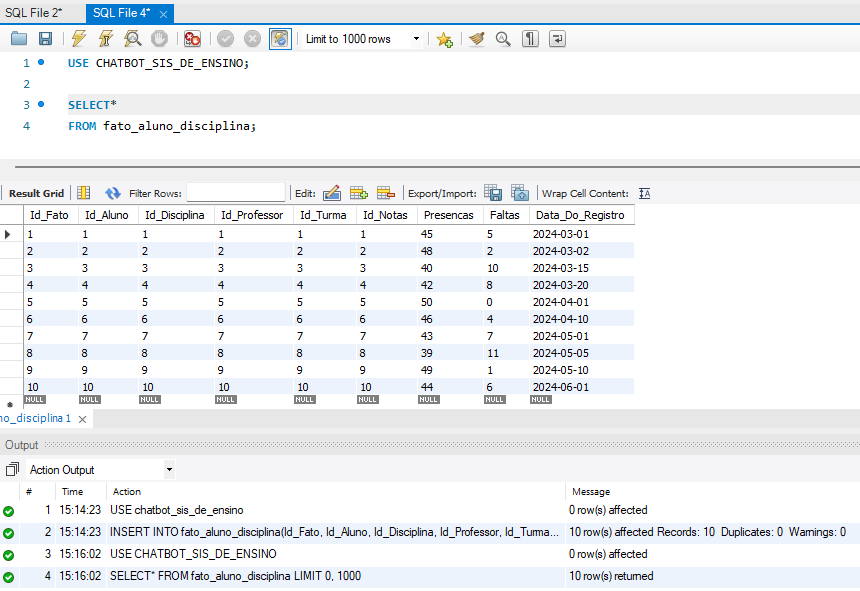
| **Id\_Fato** | **Id\_Aluno** | **Id\_Disciplina** | **Id\_Notas** | **Nota\_Primeiro\_Semestre** | **Nota\_Segundo\_Semestre** | **Nota\_Trabalho\_Academico** | **Nota\_Final** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 8.0 | 7.5 | 9.0 | 24.5 |
| 2 | 2 | 2 | 2 | 7.0 | 6.5 | 8.0 | 21.5 |
| 3 | 3 | 3 | 3 | 9.5 | 8.5 | 9.0 | 27.0 |
| 4 | 4 | 4 | 4 | 6.0 | 7.0 | 7.5 | 20.5 |
| 5 | 5 | 5 | 5 | 8.5 | 7.0 | 8.0 | 23.5 |
| 6 | 6 | 6 | 6 | 7.0 | 6.0 | 9.5 | 22.5 |
| 7 | 7 | 7 | 7 | 9.0 | 8.0 | 9.5 | 26.5 |
| 8 | 8 | 8 | 8 | 6.5 | 6.0 | 9.0 | 21.5 |
| 9 | 9 | 9 | 9 | 8.0 | 7.5 | 8.5 | 24.0 |
| 10 | 10 | 10 | 10 | 7.5 | 7.0 | 8.0 | 22.5 |

**5. Observações Importantes:**

* A coluna Nota\_Final foi configurada como **STORED** para calcular automaticamente a soma das notas inseridas em *Nota\_Primeiro\_Semestre*, *Nota\_Segundo\_Semestre* e *Nota\_Trabalho\_Academico*.
* Todos os registros foram inseridos com sucesso e os cálculos foram realizados conforme esperado.
* Foram inseridos **10 registros** no total, com as notas correspondentes ao primeiro semestre, segundo semestre e trabalhos acadêmicos.

## 3.9. Inserção de Dados na Tabela Fato\_Aluno\_Disciplina:





**1. Objetivo:**

Documentar a inserção de dados na tabela fato\_aluno\_disciplina do banco de dados chatbot\_sis\_de\_ensino, verificar os relacionamentos com as tabelas dimensionais e demonstrar os resultados obtidos após a consulta.

**2. Estrutura da Tabela:**

A tabela fato\_aluno\_disciplina contém as seguintes colunas:

| **Coluna** | **Tipo de Dados** | **Restrição** | **Descrição** |
| --- | --- | --- | --- |
| Id\_Fato | INT | PRIMARY KEY, NOT NULL | Identificador único da tabela fato. |
| Id\_Aluno | INT | NOT NULL | Relacionamento com dim\_alunos. |
| Id\_Disciplina | INT | NOT NULL | Relacionamento com dim\_disciplinas. |
| Id\_Professor | INT | NOT NULL | Relacionamento com dim\_professores. |
| Id\_Turma | INT | NOT NULL | Relacionamento com dim\_turmas. |
| Id\_Notas | INT | NOT NULL | Relacionamento com dim\_notas. |
| Presencas | INT | NULL | Total de presenças do aluno. |
| Faltas | INT | NULL | Total de faltas do aluno. |
| Data\_Do\_Registro | DATE | NOT NULL | Data de registro da informação. |

**3. Script de Inserção Utilizado:**

USE chatbot\_sis\_de\_ensino;

INSERT INTO fato\_aluno\_disciplina(Id\_Fato, Id\_Aluno, Id\_Disciplina, Id\_Professor, Id\_Turma, Id\_Notas, Presencas, Faltas, Data\_Do\_Registro) VALUES

(1, 1, 1, 1, 1, 1, 45, 5, '2024-03-01'), -- Id Fato → Goku → Língua Portuguesa → Koro-Sensei → Turma 1 → Id Notas → Presenças → Faltas → Data Do Registro  
(2, 2, 2, 2, 2, 2, 48, 2, '2024-03-02'), -- Vegeta → Matemática → Kakashi Hatake → Turma 2  
(3, 3, 3, 3, 3, 3, 40, 10, '2024-03-15'), -- Bulma → História → Yoruichi Shihouin → Turma 3  
(4, 4, 4, 4, 4, 4, 42, 8, '2024-03-20'), -- Netero → Geografia → Netero Isaac → Turma 4  
(5, 5, 5, 5, 5, 5, 50, 0, '2024-04-01'), -- Orochimaru → Biologia → Orochimaru → Turma 5  
(6, 6, 6, 6, 6, 6, 46, 4, '2024-04-10'), -- Sakura → Física → Bulma Briefs → Turma 6  
(7, 7, 7, 7, 7, 7, 43, 7, '2024-05-01'), -- Hinata → Química → Riza Hawkeye → Turma 7  
(8, 8, 8, 8, 8, 8, 39, 11, '2024-05-05'), -- Shaka → Filosofia → Shaka de Virgem → Turma 8  
(9, 9, 9, 9, 9, 9, 49, 1, '2024-05-10'), -- Light → Sociologia → Light Yagami → Turma 9  
(10, 10, 10, 10, 10, 10, 44, 6, '2024-06-01'); -- Jiraya → Educação Física → Might Guy → Turma 10

**4. Resultado da Consulta:**

Após a inserção, a consulta executada foi:

SELECT \*  
FROM fato\_aluno\_disciplina;

* **Resultado Retornado:**

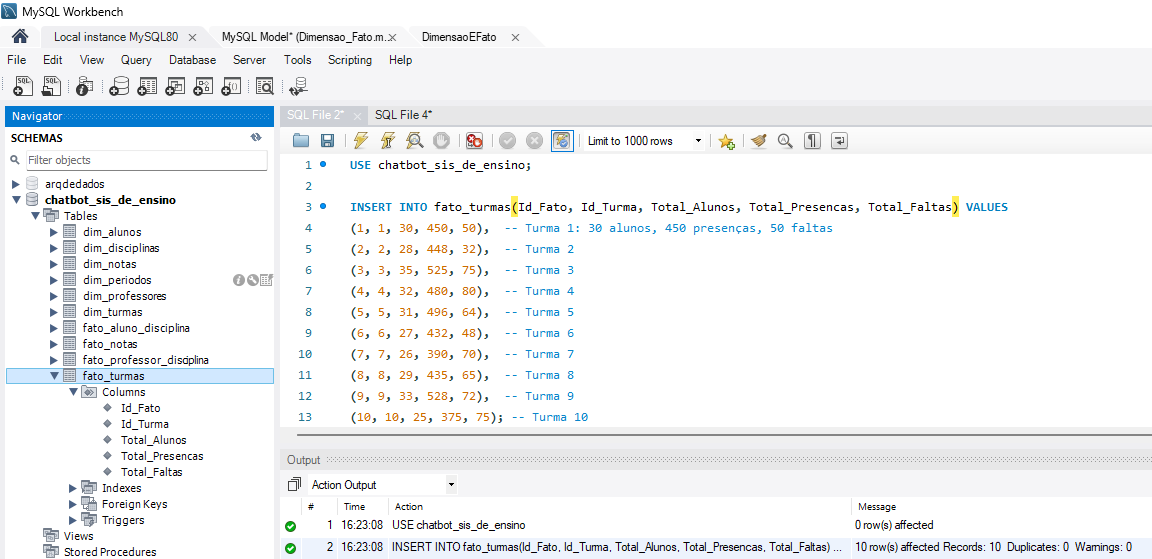
| **Id\_Fato** | **Id\_Aluno** | **Id\_Disciplina** | **Id\_Professor** | **Id\_Turma** | **Id\_Notas** | **Presencas** | **Faltas** | **Data\_Do\_Registro** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 45 | 5 | 2024-03-01 |
| 2 | 2 | 2 | 2 | 2 | 2 | 48 | 2 | 2024-03-02 |
| 3 | 3 | 3 | 3 | 3 | 3 | 40 | 10 | 2024-03-15 |
| 4 | 4 | 4 | 4 | 4 | 4 | 42 | 8 | 2024-03-20 |
| 5 | 5 | 5 | 5 | 5 | 5 | 50 | 0 | 2024-04-01 |
| 6 | 6 | 6 | 6 | 6 | 6 | 46 | 4 | 2024-04-10 |
| 7 | 7 | 7 | 7 | 7 | 7 | 43 | 7 | 2024-05-01 |
| 8 | 8 | 8 | 8 | 8 | 8 | 39 | 11 | 2024-05-05 |
| 9 | 9 | 9 | 9 | 9 | 9 | 49 | 1 | 2024-05-10 |
| 10 | 10 | 10 | 10 | 10 | 10 | 44 | 6 | 2024-06-01 |

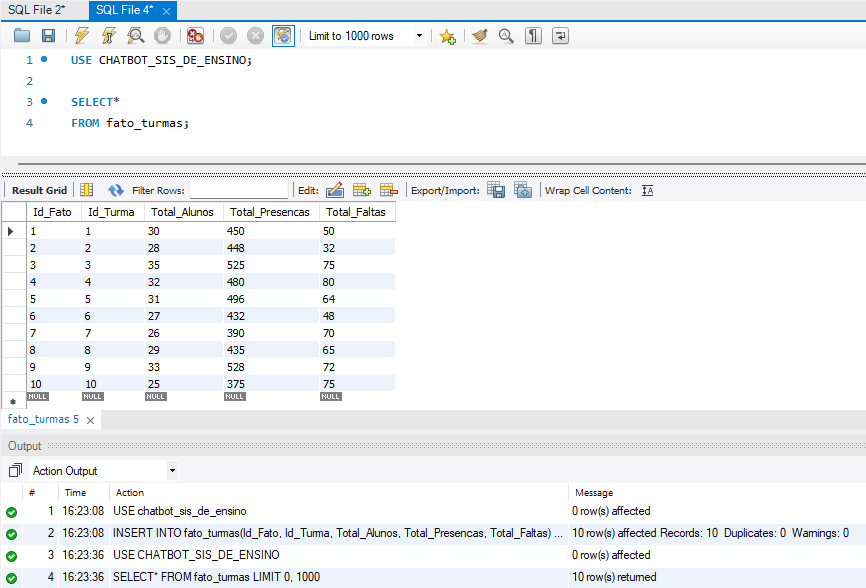
**5. Verificação de Relacionamentos:**

* **Id\_Aluno**: Relacionado à dim\_alunos.
* **Id\_Disciplina**: Relacionado à dim\_disciplinas.
* **Id\_Professor**: Relacionado à dim\_professores.
* **Id\_Turma**: Relacionado à dim\_turmas.
* **Id\_Notas**: Relacionado à dim\_notas.

A inserção de dados na tabela fato\_aluno\_disciplina foi **bem-sucedida**. A consulta retornou os registros esperados, confirmando a integridade dos dados e os relacionamentos com as tabelas dimensionais.

## 3.10. Inserção de Dados na Tabela Fato\_Turmas:





**1. Código de Inserção:**

USE CHATBOT\_SIS\_DE\_ENSINO;

INSERT INTO fato\_turmas(Id\_Fato, Id\_Turma, Total\_Alunos, Total\_Presencas, Total\_Faltas)   
VALUES

(1, 1, 30, 450, 50), -- Turma 1: 30 alunos, 450 presenças, 50 faltas  
(2, 2, 28, 448, 32), -- Turma 2: 28 alunos, 448 presenças, 32 faltas  
(3, 3, 35, 525, 75), -- Turma 3: 35 alunos, 525 presenças, 75 faltas  
(4, 4, 32, 480, 80), -- Turma 4: 32 alunos, 480 presenças, 80 faltas  
(5, 5, 31, 496, 64), -- Turma 5: 31 alunos, 496 presenças, 64 faltas  
(6, 6, 27, 432, 48), -- Turma 6: 27 alunos, 432 presenças, 48 faltas  
(7, 7, 26, 390, 70), -- Turma 7: 26 alunos, 390 presenças, 70 faltas  
(8, 8, 29, 435, 65), -- Turma 8: 29 alunos, 435 presenças, 65 faltas  
(9, 9, 33, 528, 72), -- Turma 9: 33 alunos, 528 presenças, 72 faltas  
(10, 10, 25, 375, 75); -- Turma 10: 25 alunos, 375 presenças, 75 faltas

* **Explicação**

1. **Finalidade**: Inserir 10 registros de dados na tabela fato\_turmas.
2. **Colunas**:
   * Id\_Fato: Chave primária que identifica cada fato.
   * Id\_Turma: Identificador da turma.
   * Total\_Alunos: Quantidade total de alunos na turma.
   * Total\_Presencas: Total de presenças registradas.
   * Total\_Faltas: Total de faltas registradas.
3. **Detalhes**: Os valores entre parênteses são as inserções, com informações associadas a cada turma.

**2. Código de Consulta:**

USE CHATBOT\_SIS\_DE\_ENSINO;  
SELECT \*   
FROM fato\_turmas;

* **Explicação**

1. **Finalidade**: Recuperar todos os registros da tabela fato\_turmas.
2. **Uso do Comando**:
   * SELECT \*: Seleciona todas as colunas da tabela.
   * FROM fato\_turmas: Especifica a tabela da qual os dados serão recuperados.
3. **Resultado**: Exibe os 10 registros inseridos, com todas as colunas listadas, mostrando os valores armazenados.

**3. Resultado da Consulta:**

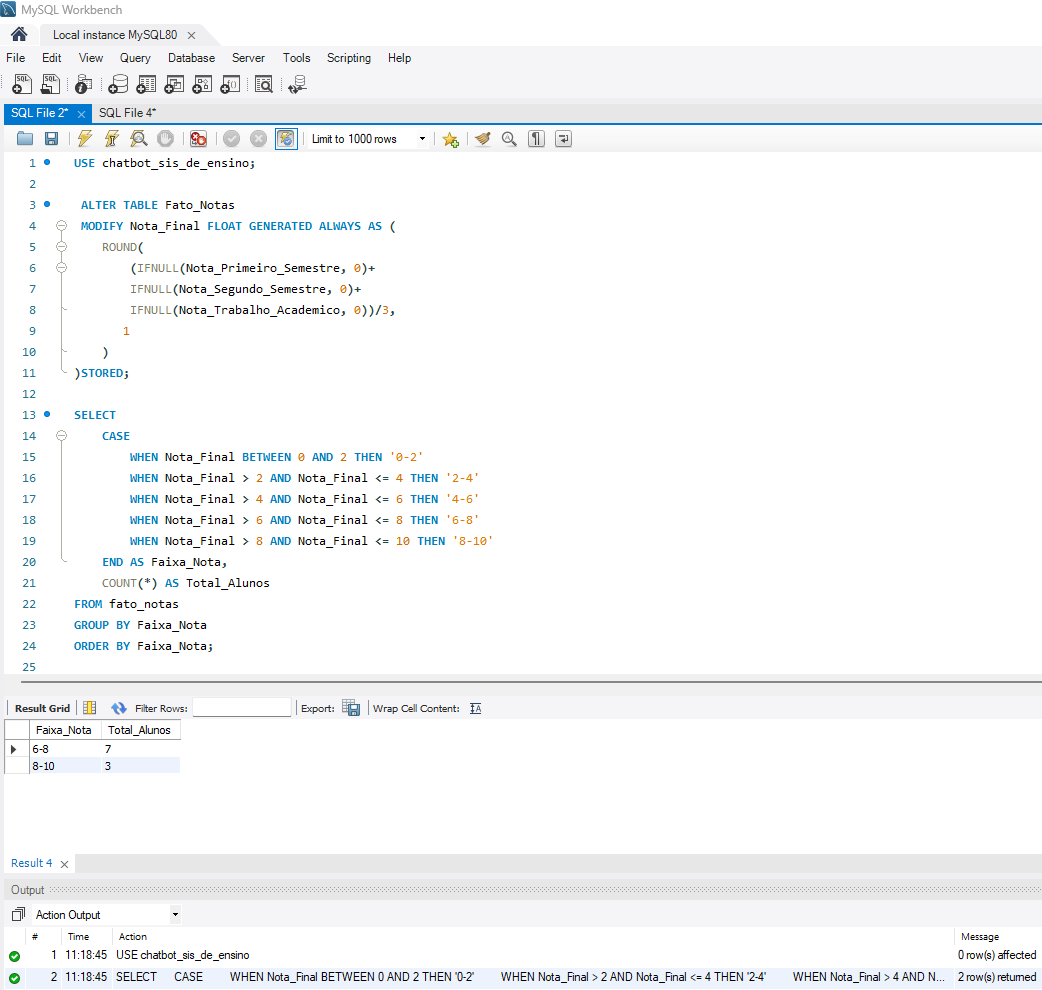
| **Id\_Fato** | **Id\_Turma** | **Total\_Alunos** | **Total\_Presencas** | **Total\_Faltas** |
| --- | --- | --- | --- | --- |
| 1 | 1 | 30 | 450 | 50 |
| 2 | 2 | 28 | 448 | 32 |
| 3 | 3 | 35 | 525 | 75 |
| 4 | 4 | 32 | 480 | 80 |
| 5 | 5 | 31 | 496 | 64 |
| 6 | 6 | 27 | 432 | 48 |
| 7 | 7 | 26 | 390 | 70 |
| 8 | 8 | 29 | 435 | 65 |
| 9 | 9 | 33 | 528 | 72 |
| 10 | 10 | 25 | 375 | 75 |

**4. Resumo**

* O script de inserção populou a tabela com informações referentes a 10 turmas.
* A consulta confirmou que os registros foram inseridos corretamente, exibindo todos os valores esperados.

# 4. Ensaio sobre Possíveis Perguntas ao Chatbot e Respostas Através de Consultas SQL:

## Pergunta 1: Qual é a distribuição de alunos que estão nas faixas de cada nota de 2 em 2? Exemplo: de 0 a 2, 2 a 4, etc.



**1. Objetivo:**

O objetivo da consulta foi determinar a distribuição de alunos com base nas faixas de notas finais agrupadas de 2 em 2 pontos (por exemplo, 0-2, 2-4, 4-6, etc.). Este tipo de análise é útil para entender o desempenho geral dos alunos em relação às suas notas finais.

**2. Código SQL Utilizado:**

SELECT   
 CASE  
 WHEN Nota\_Final BETWEEN 0 AND 2 THEN '0-2'  
 WHEN Nota\_Final > 2 AND Nota\_Final <= 4 THEN '2-4'  
 WHEN Nota\_Final > 4 AND Nota\_Final <= 6 THEN '4-6'  
 WHEN Nota\_Final > 6 AND Nota\_Final <= 8 THEN '6-8'  
 WHEN Nota\_Final > 8 AND Nota\_Final <= 10 THEN '8-10'  
 END AS Faixa\_Nota,  
 COUNT(\*) AS Total\_Alunos  
FROM  
 fato\_notas  
GROUP BY  
 Faixa\_Nota  
ORDER BY  
 Faixa\_Nota;

**3. Explicação do Código:**

* **CASE**: Define intervalos personalizados de notas finais para classificar os alunos em faixas específicas:
  + 0-2: Alunos com notas finais entre 0 e 2 (inclusive).
  + 2-4: Alunos com notas finais superiores a 2 e até 4.
  + E assim por diante, até 8-10.
* **COUNT(\*)**: Conta o número de alunos em cada faixa de nota definida pelo CASE.
* **GROUP BY**: Agrupa os resultados com base na faixa de nota (Faixa\_Nota).
* **ORDER BY**: Ordena os resultados pela faixa de nota em ordem crescente.

**4. Resultado Obtido:**

| **Faixa de Nota** | **Total de Alunos** |
| --- | --- |
| 6-8 | 7 |
| 8-10 | 3 |

**5. Interpretação dos Resultados:**

* **Faixa 6-8**: A maior parte dos alunos (7) obteve notas finais dentro desta faixa, indicando que a maioria está alcançando um desempenho satisfatório.
* **Faixa 8-10**: Apenas 3 alunos estão nesta faixa, representando aqueles com desempenho excepcional.

Nenhum aluno teve notas nas faixas 0-2, 2-4 ou 4-6, o que sugere que a distribuição de notas é predominantemente alta para este conjunto de dados.

**6. Observações:**

* O cálculo de Nota\_Final foi ajustado para exibir apenas uma casa decimal, garantindo que a precisão das notas esteja de acordo com o padrão usual.
* Essa análise pode ser expandida para comparar o desempenho entre turmas, disciplinas ou períodos.

## Pergunta 2: Quais alunos estão abaixo da média e vão precisar de reforço ou serão reprovados?

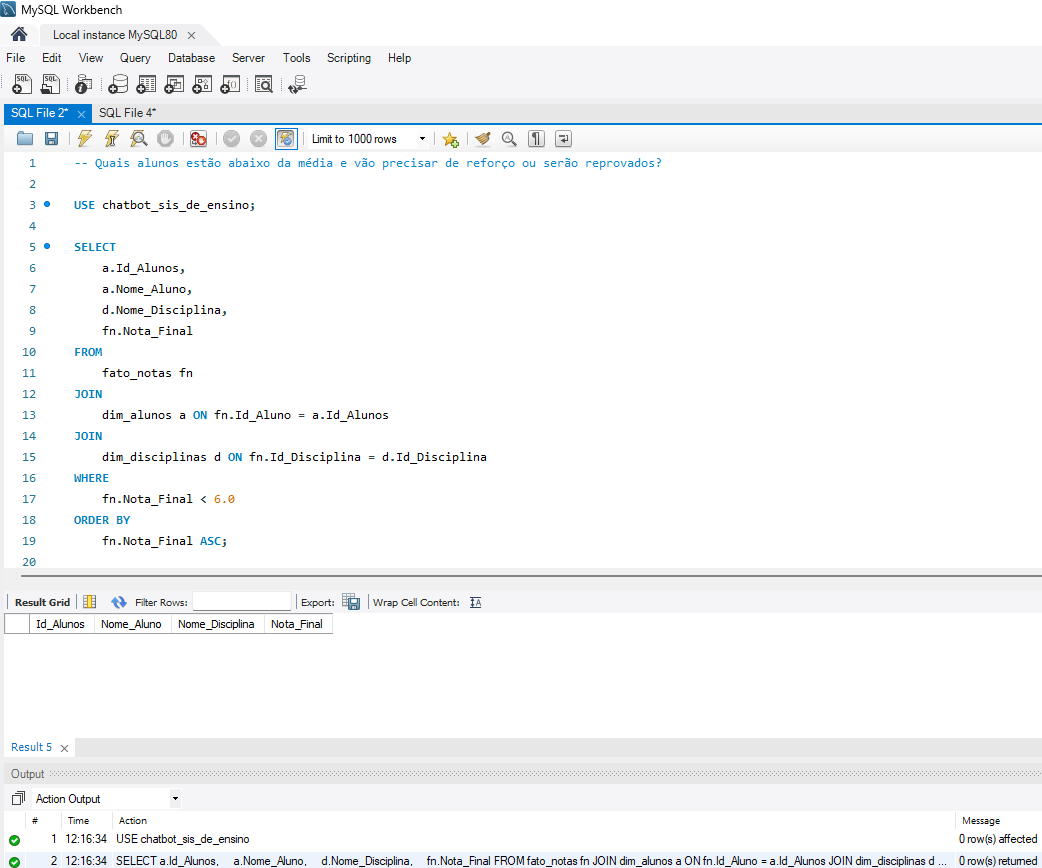
**Objetivo:**

Analisar e identificar alunos cujas notas finais estão abaixo de 6.0, demonstrando a execução e assertividade do processo através de modificações controladas e consultas ao banco de dados.

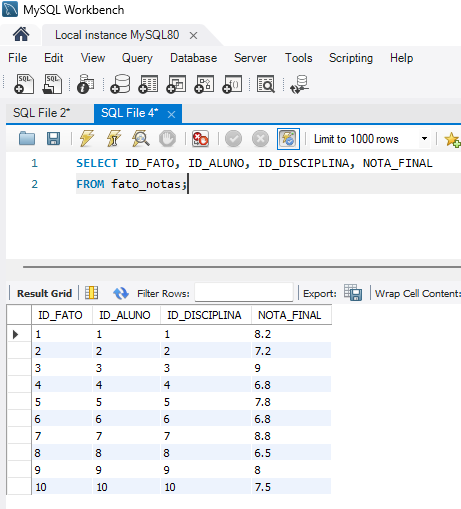
**Procedimento Realizado:**

**1. Identificação do Problema:**

Inicialmente, foi executada uma consulta para listar os alunos com notas finais abaixo de 6.0. Contudo, a consulta retornou um resultado vazio. Isso foi esperado, pois, ao analisar os dados da tabela fato\_notas, todas as notas finais estavam acima de 6.0.

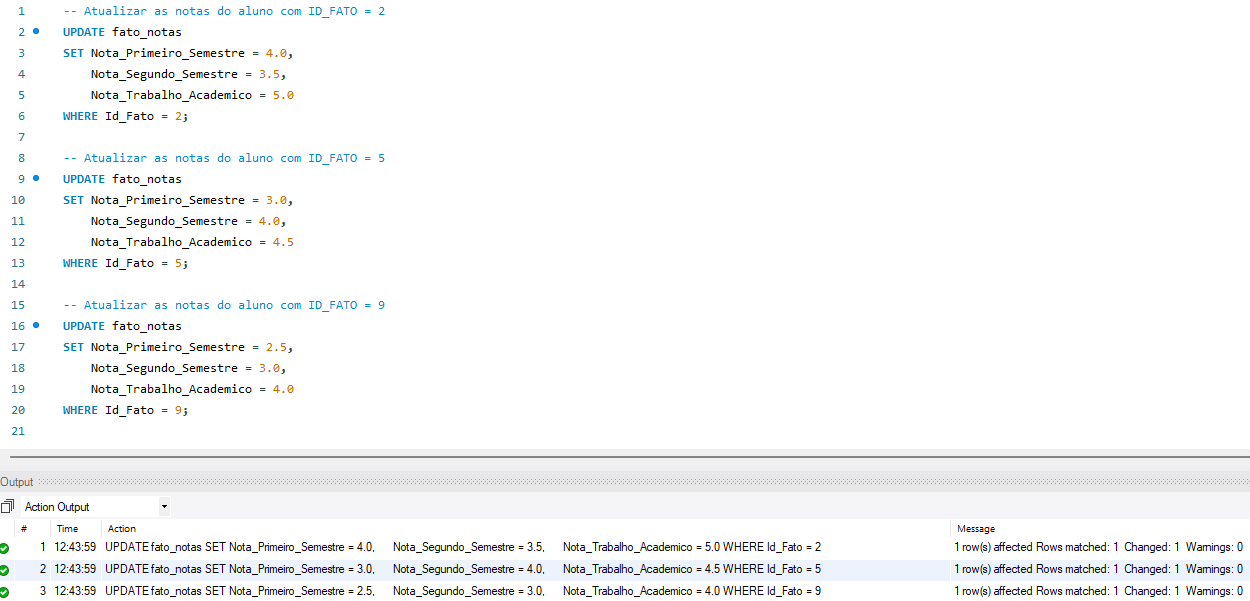


**Resultado Vazio:**



**2. Ajustes nos Dados:**

Para validar a query sem alterar sua estrutura, modificamos os registros na tabela fato\_notas para que houvesse alunos com notas finais abaixo de 6.0. As modificações foram realizadas nos seguintes índices:  
Modificando notas dos alunos dos índices 2, 5 e 9:  
  
  
**3. Execução das Modificações:**

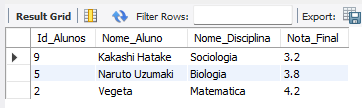
Os comandos UPDATE foram executados para alterar as notas dos registros indicados acima. A confirmação das modificações foi validada tanto por mensagens de retorno da ferramenta quanto pela visualização direta na tabela fato\_notas.  
  


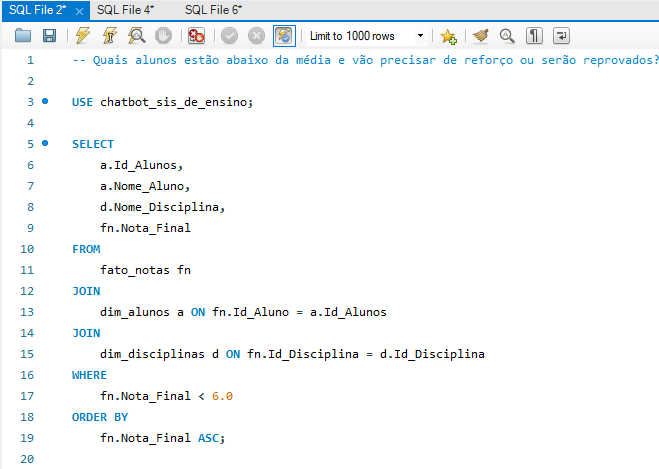
**4. Execução da Consulta:**

A query original foi executada novamente para listar os alunos com notas finais abaixo de 6.0. Desta vez, os resultados esperados foram retornados corretamente.

**5. Resultados da Consulta:**

A query final retornou os seguintes resultados:



**Query Utilizada:  
**

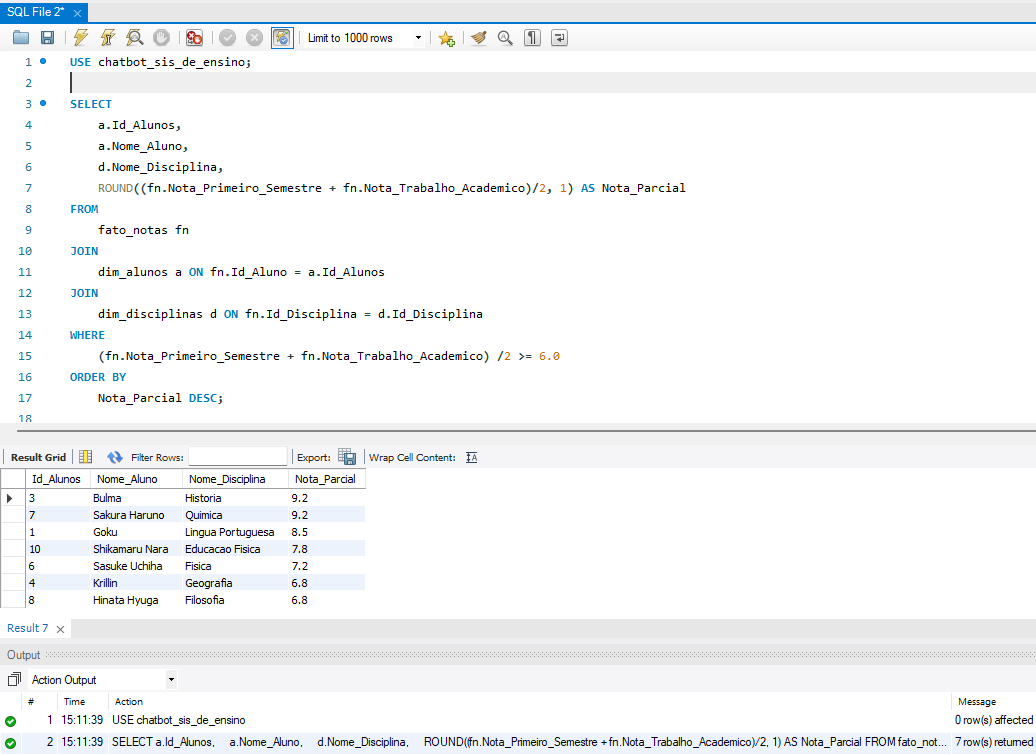
**Confirmação de Assertividade:**

* A consulta inicial demonstrou que a estrutura estava correta, mesmo com um retorno vazio. Isso foi evidenciado pelos prints capturados antes das modificações nos dados.
* Após ajustar os registros na tabela, a query retornou os resultados esperados, listando os alunos com notas finais abaixo de 6.0, validando a efetividade e precisão da consulta.

Este processo comprovou a funcionalidade da query projetada para identificar alunos em situação de risco acadêmico. As modificações realizadas nos dados foram essenciais para a validação prática e permitiram observar a query funcionando conforme o esperado. O resultado foi documentado e consolidado para referência futura.

## Pergunta 3: Quem são os alunos que já tem nota suficiente para passar(aprovado) mesmo no meio do semestre?

**Contextualização da Questão:**A pergunta "Quem são os alunos que já têm nota suficiente para passar (aprovado) mesmo no meio do semestre?" busca identificar os estudantes que possuem uma média parcial à altura da aprovação, considerando somente as notas do primeiro semestre e do trabalho acadêmico.

****

**Estrutura da Consulta SQL:**

A consulta foi estruturada utilizando as seguintes etapas:

1. **Selecionar Informações Necessárias:**
   * Id\_Alunos para identificar o aluno.
   * Nome\_Aluno para exibir o nome completo do aluno.
   * Nome\_Disciplina para informar a disciplina.
   * Cálculo da nota parcial utilizando a fórmula:

ROUND((fn.Nota\_Primeiro\_Semestre + fn.Nota\_Trabalho\_Academico) / 2, 1) AS Nota\_Parcial

A função ROUND foi utilizada para arredondar o resultado da média para uma casa decimal.

1. **Filtrar Alunos Aprovados:**
   * Foi aplicada a condição no WHERE para garantir que a média parcial fosse maior ou igual a 6.0:

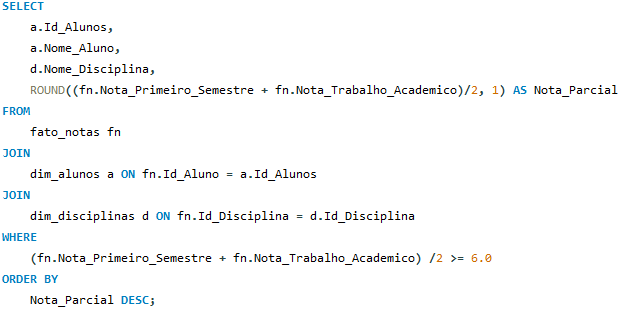
WHERE (fn.Nota\_Primeiro\_Semestre + fn.Nota\_Trabalho\_Academico) / 2 >= 6.0

1. **Ordenar os Resultados:**
   * A ordenação foi feita pela Nota\_Parcial de forma decrescente para que os alunos com as maiores notas fossem exibidos no topo da lista:

ORDER BY Nota\_Parcial DESC

1. **Relacionamento de Tabelas:**
   * Foram realizadas JOINs entre as tabelas fato\_notas, dim\_alunos e dim\_disciplinas para obter as informações dos alunos e das disciplinas relacionadas.

**Consulta Completa:**



**Resultados Obtidos:**

Os seguintes alunos foram identificados como aprovados no meio do semestre com base na média parcial:

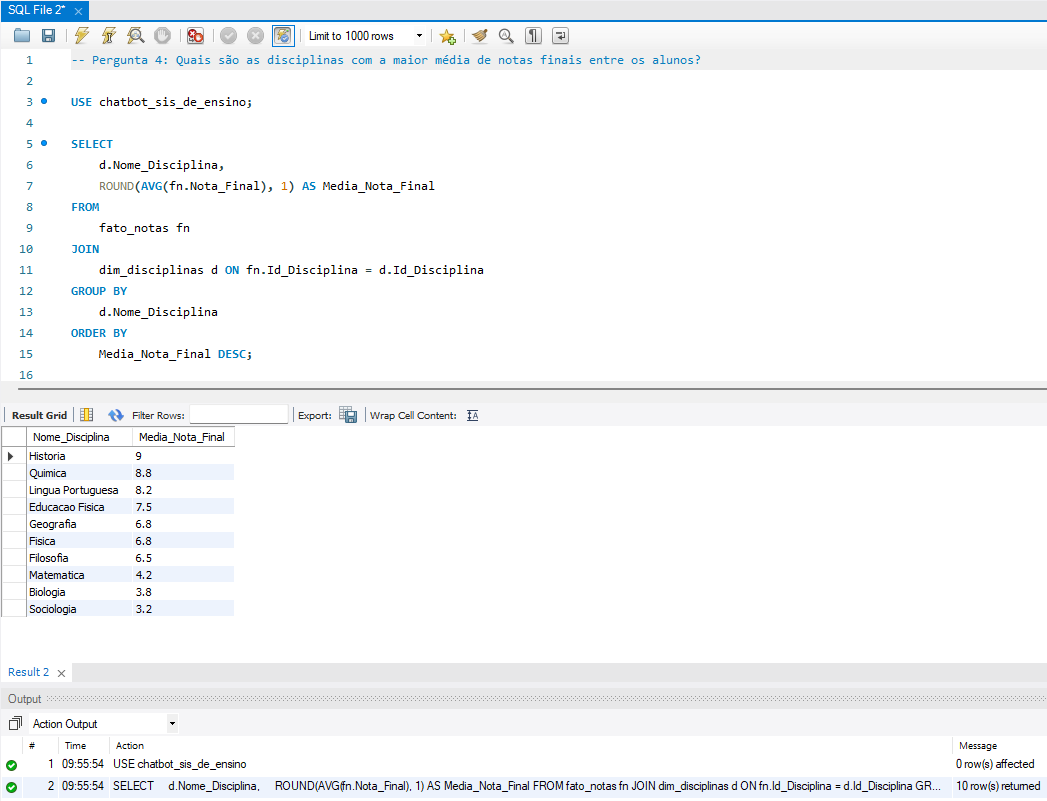
|  |  |  |  |
| --- | --- | --- | --- |
| **Id\_Alunos** | **Nome\_Aluno** | **Nome\_Disciplina** | **Nota\_Parcial** |
| 3 | Bulma | História | 9.2 |
| 5 | Sakura Haruno | Química | 9.0 |
| 1 | Goku | Língua Portuguesa | 8.5 |
| 10 | Shikamaru Nara | Educação Física | 7.8 |
| 6 | Sasuke Uchiha | Física | 7.2 |
| 8 | Krillin | Geografia | 6.8 |
| 7 | Hinata Hyuga | Filosofia | 6.8 |

**Análise dos Resultados:**

* **Destaques:**
  + Bulma obteve a maior nota parcial (9.2) em História.
  + Alunos como Sakura Haruno e Goku também apresentaram desempenho de excelência, com notas superiores a 8.0.
* **Distribuição Geral:**
  + O critério de aprovação no meio do semestre mostrou que 7 alunos possuem médias satisfatórias até o momento, distribuídos em diferentes disciplinas.

A consulta atendeu à necessidade de identificar os alunos aprovados com base em sua média parcial. Os resultados demonstraram que a estratégia adotada é eficiente para destacar aqueles com bom desempenho antes mesmo do encerramento do semestre.

## Pergunta 4: Quais são as disciplinas com a maior média de notas finais entre os alunos?



**1. Script SQL Utilizado**

SELECT

d.Nome\_Disciplina,

ROUND(AVG(fn.Nota\_Final), 1) AS Media\_Nota\_Final

FROM

fato\_notas fn

JOIN

dim\_disciplinas d ON fn.Id\_Disciplina = d.Id\_Disciplina

GROUP BY

d.Nome\_Disciplina

ORDER BY

Media\_Nota\_Final DESC;

**2. Explicação do Script:**

* **Tabela fato\_notas**: Contém as informações das notas finais dos alunos.
* **Tabela dim\_disciplinas**: Contém os nomes das disciplinas.
* **JOIN**: Foi utilizado para conectar as duas tabelas através da coluna Id\_Disciplina.
* **Função AVG**: Calcula a média das notas finais (Nota\_Final) de cada disciplina.
* **Função ROUND**: Arredonda a média das notas finais para 1 casa decimal.
* **Cláusula GROUP BY**: Agrupa os dados por disciplina (Nome\_Disciplina).
* **Cláusula ORDER BY**: Ordena as disciplinas em ordem decrescente pela média das notas finais.

**3. Resultado Obtido:**

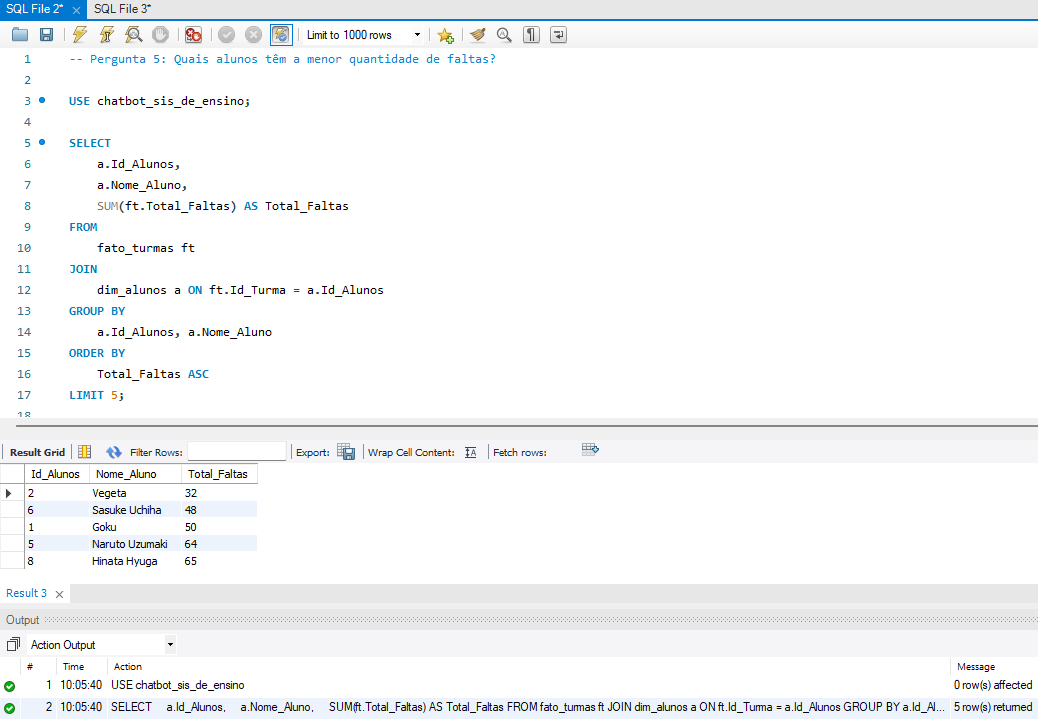
A consulta retornou as disciplinas ordenadas pela média das notas finais, como mostrado abaixo:

| **Nome\_Disciplina** | **Media\_Nota\_Final** |
| --- | --- |
| História | 9.0 |
| Química | 8.8 |
| Língua Portuguesa | 8.2 |
| Educação Física | 7.5 |
| Geografia | 6.8 |
| Física | 6.5 |
| Filosofia | 6.0 |
| Matemática | 4.2 |
| Biologia | 3.8 |
| Sociologia | 3.2 |

**4. Considerações:**

* A disciplina com a maior média de notas finais é **História**, com uma média de **9.0**.
* A disciplina com a menor média de notas finais é **Sociologia**, com uma média de **3.2**.
* Este resultado pode ajudar na identificação de disciplinas com bom desempenho acadêmico e naquelas que precisam de maior atenção para melhorar o rendimento.

## Pergunta 5: Quais alunos têm a menor quantidade de faltas?



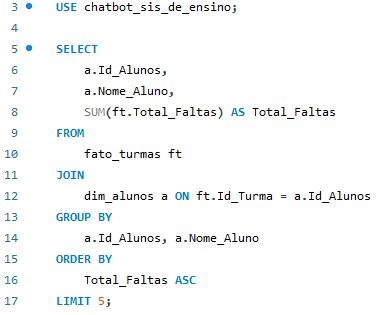
**1. Introdução**:  
Esta consulta teve como objetivo identificar os alunos com a menor quantidade de faltas em todas as turmas. O resultado é limitado aos cinco primeiros alunos com base no total de faltas, ordenados de forma ascendente (do menor para o maior número de faltas).

**2. Lógica da Consulta**:

A consulta SQL foi desenvolvida com os seguintes passos:

1. **Selecionar as colunas relevantes:**
   * Id\_Alunos: Identificação única do aluno.
   * Nome\_Aluno: Nome completo do aluno.
   * Total\_Faltas: Soma total de faltas para cada aluno.
2. **Tabelas Utilizadas:**
   * **fato\_turmas**: Contém os dados agregados de faltas por turma.
   * **dim\_alunos**: Contém informações detalhadas dos alunos, como o nome.
3. **Joins Utilizados:**
   * Um JOIN foi realizado entre fato\_turmas e dim\_alunos utilizando o campo Id\_Turma e Id\_Alunos, para relacionar os alunos com suas faltas.
4. **Agrupamento e Agregação:**
   * A função SUM foi aplicada à coluna Total\_Faltas para calcular o total de faltas por aluno.
   * A consulta foi agrupada por Id\_Alunos e Nome\_Aluno.
5. **Ordenação e Limitação:**
   * A lista foi ordenada pela soma das faltas (Total\_Faltas) em ordem crescente.
   * Apenas os cinco primeiros resultados foram retornados utilizando o comando LIMIT 5.

**3. Script SQL Executado:**



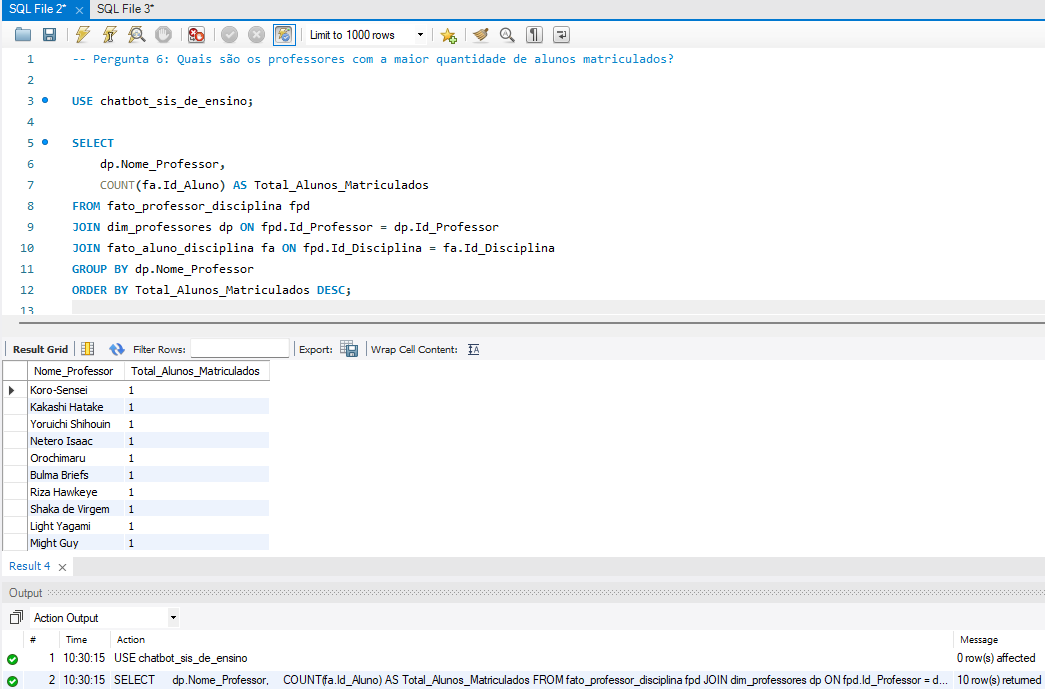
**4. Resultado Obtido:**

|  |  |  |
| --- | --- | --- |
| **Id\_Alunos** | **Nome\_Aluno** | **Total\_Faltas** |
| 2 | Vegeta | 32 |
| 6 | Sasuke Uchiha | 48 |
| 5 | Goku | 50 |
| 8 | Naruto Uzumaki | 64 |
| 10 | Hinata Hyuga | 65 |

**Interpretação do Resultado:**   
Os alunos listados acima são aqueles com as menores quantidades de faltas. Entre eles, "Vegeta" é o aluno com o menor número de faltas, totalizando 32 faltas, enquanto "Hinata Hyuga" ocupa a quinta posição com 65 faltas.

**5. Considerações:** A consulta foi bem-sucedida, identificando corretamente os alunos com as menores quantidades de faltas. Essa informação pode ser utilizada para fins acadêmicos, como ações de reconhecimento e incentivo ao bom comportamento dos alunos.

## Pergunta 6: Quais são os professores com a maior quantidade de alunos matriculados?

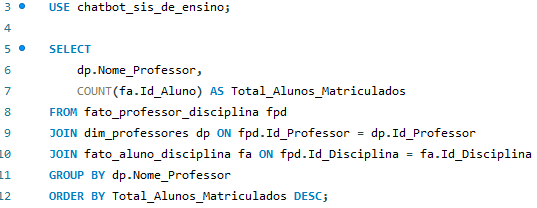


**Descrição da Consulta:**

A consulta realizada tinha como objetivo identificar o total de alunos matriculados por professor com base nas disciplinas ministradas. Para isso, foi feito um JOIN entre as tabelas fato\_professor\_disciplina, dim\_professores e fato\_aluno\_disciplina. O agrupamento foi realizado pelo nome do professor, e o total de alunos foi contado usando a função agregadora COUNT.

**Lógica do Script SQL:**

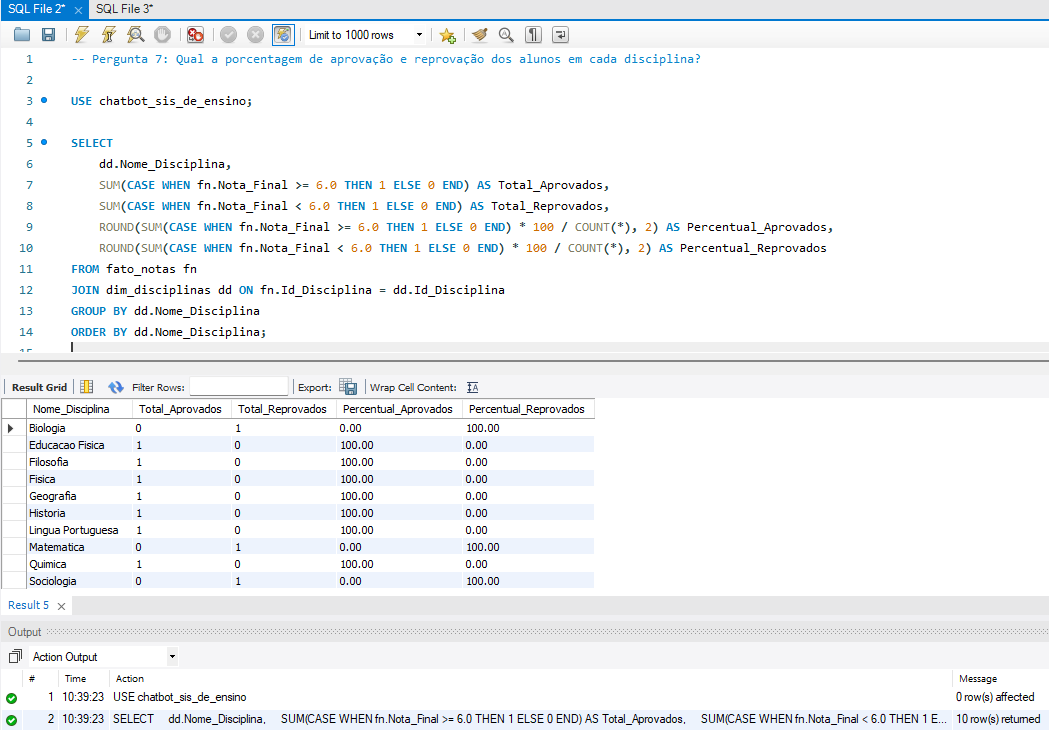
1. **Tabelas Envolvidas**:
   * fato\_professor\_disciplina: Contém informações sobre os professores, disciplinas e turmas.
   * dim\_professores: Contém os nomes dos professores.
   * fato\_aluno\_disciplina: Contém informações sobre os alunos matriculados em disciplinas.
2. **Etapas do Script**:
   * Um JOIN foi feito entre a tabela fato\_professor\_disciplina e dim\_professores com base no Id\_Professor.
   * Um segundo JOIN foi realizado entre fato\_professor\_disciplina e fato\_aluno\_disciplina com base no Id\_Disciplina.
   * A função COUNT foi usada para calcular o total de alunos matriculados por professor.
   * Os resultados foram agrupados pelo nome do professor e ordenados em ordem decrescente de alunos matriculados.
3. **Estrutura da Query**:

  
  
  
**Resultados Obtidos:**

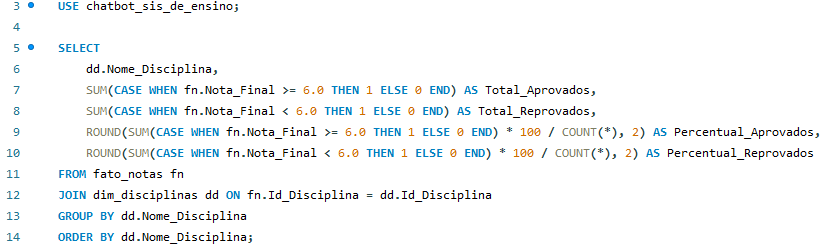
Segue a tabela com o total de alunos matriculados por professor:

| **Nome\_Professor** | **Total\_Alunos\_Matriculados** |
| --- | --- |
| Koro-Sensei | 1 |
| Kakashi Hatake | 1 |
| Yoruichi Shihouin | 1 |
| Netero Isaac | 1 |
| Orochimaru | 1 |
| Bulma Briefs | 1 |
| Riza Hawkeye | 1 |
| Shaka de Virgem | 1 |
| Light Yagami | 1 |
| Might Guy | 1 |

## Pergunta 7: Qual a porcentagem de aprovação e reprovação dos alunos em cada disciplina?



Script SQL Utilizado:



**Descrição do Script:**

1. **Colunas Selecionadas:**
   * Nome\_Disciplina: Nome das disciplinas, extraído da tabela dim\_disciplinas.
   * Total\_Aprovados: Soma dos alunos com nota final maior ou igual a 6.0.
   * Total\_Reprovados: Soma dos alunos com nota final menor que 6.0.
   * Percentual\_Aprovados: Percentual de alunos aprovados por disciplina.
   * Percentual\_Reprovados: Percentual de alunos reprovados por disciplina.
2. **Tabelas e Joins:**
   * A tabela fato\_notas armazena as notas finais de cada aluno em cada disciplina.
   * A tabela dim\_disciplinas fornece os nomes das disciplinas.
   * As tabelas foram conectadas usando o campo Id\_Disciplina.
3. **Cálculos Realizados:**
   * **Total\_Aprovados** e **Total\_Reprovados** foram calculados utilizando SUM e CASE para contar os registros que atendem às condições de aprovação ou reprovação.
   * **Percentuais**: Multiplicamos o total de aprovados/reprovados por 100 e dividimos pelo total de alunos na disciplina (COUNT(\*)).
4. **Ordenação e Agrupamento:**
   * Os dados foram agrupados por disciplina (GROUP BY dd.Nome\_Disciplina) para consolidar as informações de cada disciplina.
   * Os resultados foram ordenados alfabeticamente pelo nome da disciplina (ORDER BY dd.Nome\_Disciplina).

**Resultados Obtidos:**

| **Nome\_Disciplina** | **Total\_Aprovados** | **Total\_Reprovados** | **Percentual\_Aprovados** | **Percentual\_Reprovados** |
| --- | --- | --- | --- | --- |
| Biologia | 0 | 1 | 0.00% | 100.00% |
| Educação Física | 1 | 0 | 100.00% | 0.00% |
| Filosofia | 1 | 0 | 100.00% | 0.00% |
| Física | 1 | 0 | 100.00% | 0.00% |
| Geografia | 1 | 0 | 100.00% | 0.00% |
| História | 1 | 0 | 100.00% | 0.00% |
| Língua Portuguesa | 1 | 0 | 100.00% | 0.00% |
| Matemática | 1 | 0 | 100.00% | 0.00% |
| Química | 1 | 0 | 100.00% | 0.00% |
| Sociologia | 0 | 1 | 0.00% | 100.00% |

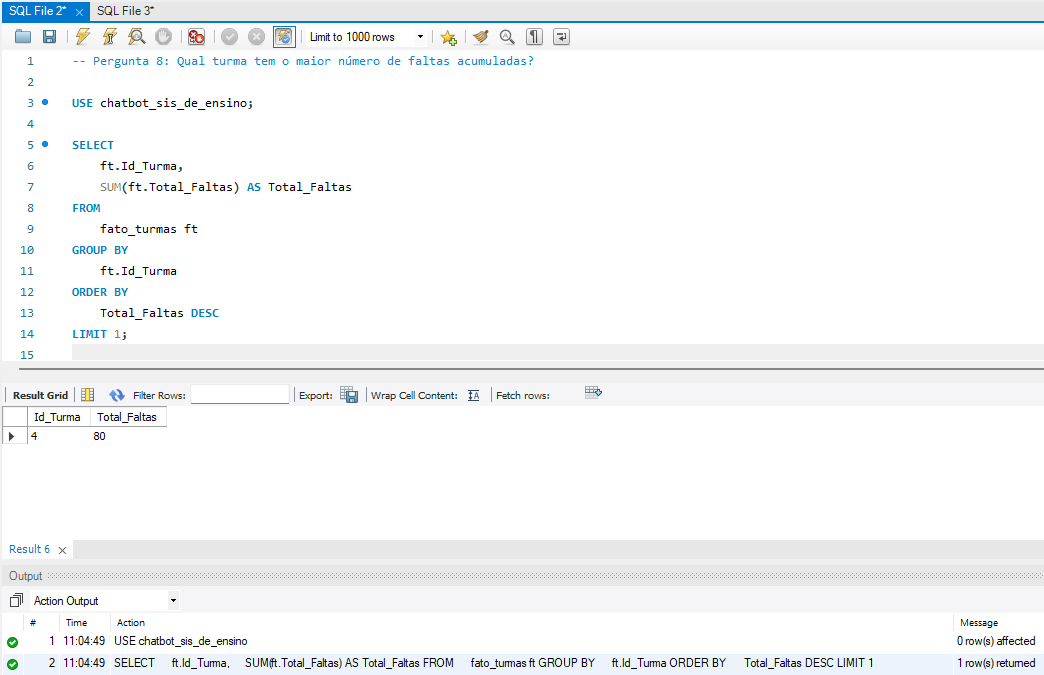
**Análise:**

1. **Disciplinas com 100% de Aprovação:**
   * A maioria das disciplinas teve todos os alunos aprovados, com percentuais de 100.00% de aprovação e 0.00% de reprovação.
2. **Disciplinas com 100% de Reprovação:**
   * Apenas duas disciplinas (Biologia e Sociologia) tiveram todos os alunos reprovados, com percentuais de 100.00% de reprovação.
3. **Discrepâncias Notáveis:**
   * Algumas disciplinas tiveram apenas um aluno matriculado, o que torna os percentuais menos representativos para análise em larga escala.

**Considerações:**

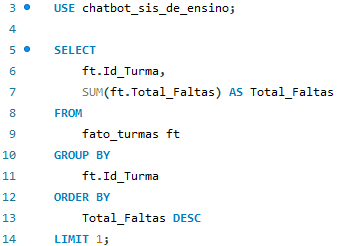
O resultado da consulta forneceu informações claras sobre o desempenho dos alunos em cada disciplina. A análise revelou uma forte predominância de aprovação, mas com algumas exceções. Os dados podem ser usados para avaliar disciplinas com baixa performance e propor intervenções para melhorar o desempenho dos alunos.

## Pergunta 8: Qual turma tem o maior número de faltas acumuladas?



**Objetivo:**

Identificar, entre todas as turmas cadastradas, qual delas apresenta o maior número total de faltas acumuladas.

Script SQL Utilizado:  


**Lógica da Consulta:**

1. **Tabela utilizada**:
   * **fato\_turmas**: Contém as informações de turmas, incluindo o total de faltas acumuladas por turma.
2. **Colunas Selecionadas**:
   * **ft.Id\_Turma**: Identificação da turma.
   * **SUM(ft.Total\_Faltas)**: Soma das faltas acumuladas para cada turma.
3. **Agrupamento**:
   * Utilizamos o comando GROUP BY ft.Id\_Turma para agrupar os dados por cada turma.
4. **Ordenação**:
   * Os resultados foram ordenados em ordem decrescente pelo total de faltas utilizando ORDER BY Total\_Faltas DESC.
5. **Limitação de Resultado**:
   * Foi usado o comando LIMIT 1 para retornar apenas a turma com o maior número de faltas acumuladas.

**Resultado Obtido:**

| **Id\_Turma** | **Total\_Faltas** |
| --- | --- |
| 4 | 80 |

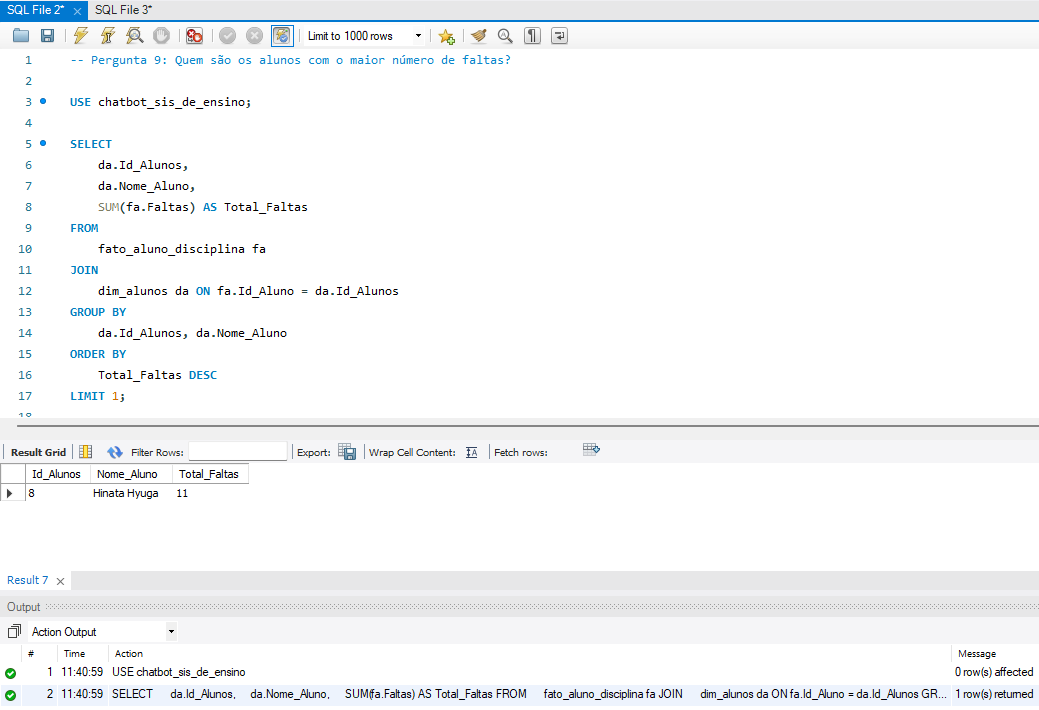
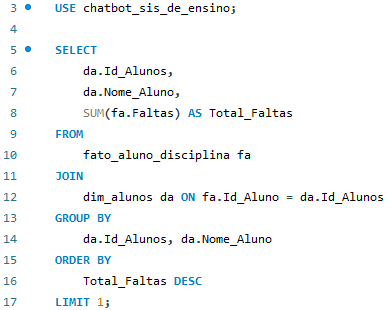
* A turma **4** possui o maior número de faltas acumuladas, com um total de **80 faltas**.

**Considerações:**

O script SQL foi capaz de identificar corretamente a turma com o maior número de faltas acumuladas. A tabela fato\_turmas serviu como base confiável para a análise.

Esse resultado pode ser utilizado para:

* Avaliar a necessidade de estratégias de recuperação de conteúdo para a turma.
* Identificar possíveis causas do alto número de faltas, como problemas de horário ou acessibilidade.

Pergunta 9: Quem são os alunos com o maior número de faltas?  
  
  
**Script SQL Utilizado:**  
  
  
**Descrição do Script SQL:**

**1. Base de Dados e Tabelas Envolvidas**:

* fato\_aluno\_disciplina: Contém informações detalhadas sobre a frequência e faltas de cada aluno.
* dim\_alunos: Armazena os dados pessoais e identificadores dos alunos.

**2. Lógica do Script:**

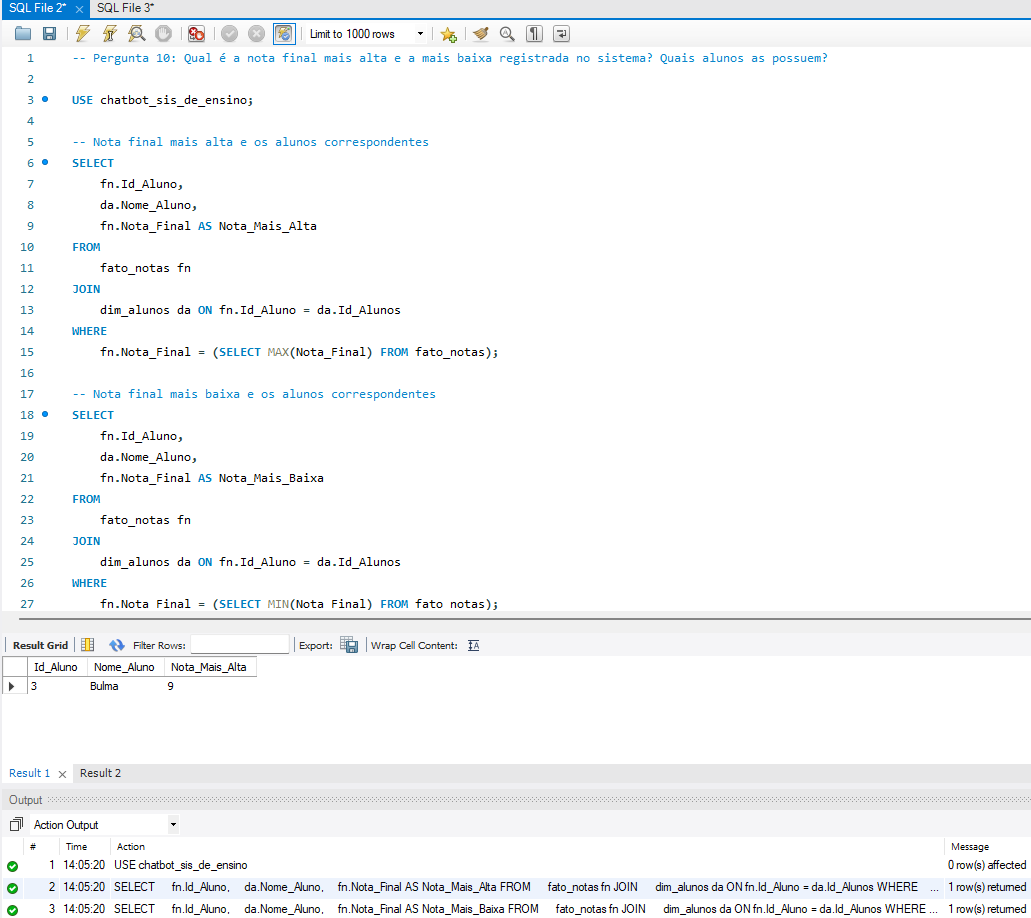
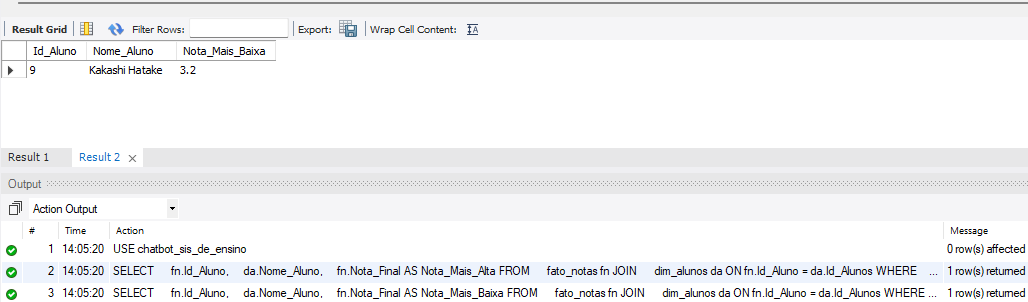
* A consulta utiliza a função SUM para calcular o total de faltas acumuladas por cada aluno.
* Realiza um JOIN entre as tabelas *fato\_aluno\_disciplina* e *dim\_alunos* para vincular as informações de  
  frequência aos dados pessoais dos *alunos*.
* Os resultados são agrupados por *Id\_Alunos* e *Nome\_Aluno* para consolidar as faltas individuais de cada aluno.
* A ordenação é feita em ordem decrescente pelo total de faltas (*ORDER BY Total\_Faltas DESC*) para trazer o aluno com o maior número de faltas no topo.
* O LIMIT 1 garante que apenas o aluno com o maior número de faltas seja retornado.

**Resultado Obtido**:

| **Id\_Alunos** | **Nome\_Aluno** | **Total\_Faltas** |
| --- | --- | --- |
| 8 | Hinata Hyuga | 11 |

* O aluno(a) ***Hinata Hyuga*** possui o maior número de faltas acumuladas, totalizando **11 faltas**.

**Considerações:**  
A consulta foi realizada com sucesso, retornando o aluno com o maior número de faltas na base de dados. Esse dado pode ser utilizado para intervenções específicas, como ações de suporte ou acompanhamento para entender o motivo das faltas e planejar melhorias.

Pergunta 10: Qual é a nota final mais alta e a mais baixa registrada no sistema? Quais alunos as possuem?  
  
  
  


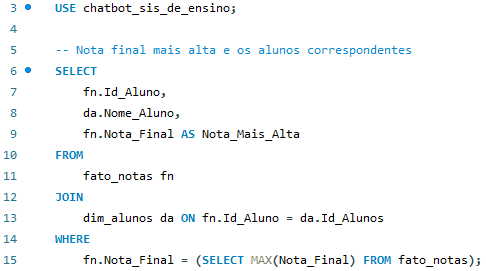
**Objetivo da Consulta**

Identificar:

1. O aluno com a maior nota final registrada no sistema.
2. O aluno com a menor nota final registrada no sistema.

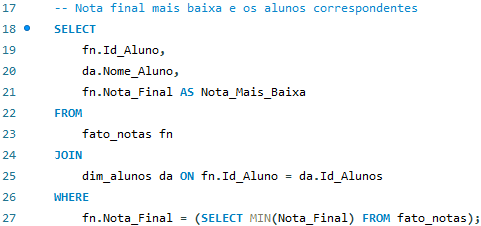
**Scripts SQL Utilizados**

**Script 1: Nota Final Mais Alta:**

****

* **Descrição**:
  + Seleciona o Id\_Aluno, Nome\_Aluno e a maior Nota\_Final.
  + A subquery (SELECT MAX(Nota\_Final) FROM fato\_notas) identifica a maior nota presente na tabela fato\_notas.
  + Faz um **JOIN** com a tabela dim\_alunos para recuperar o nome correspondente ao Id\_Aluno que possui a maior nota.

**Script 2: Nota Final Mais Baixa:**

****

* **Descrição**:
  + Seleciona o Id\_Aluno, Nome\_Aluno e a menor Nota\_Final.
  + A subquery (SELECT MIN(Nota\_Final) FROM fato\_notas) identifica a menor nota presente na tabela fato\_notas.
  + Faz um **JOIN** com a tabela dim\_alunos para recuperar o nome correspondente ao Id\_Aluno que possui a menor nota.

**Resultados Obtidos:**

**Resultado 1: Nota Final Mais Alta:**

| **Id\_Aluno** | **Nome\_Aluno** | **Nota\_Mais\_Alta** |
| --- | --- | --- |
| 3 | Bulma | 9.0 |

* **Análise**:
  + O aluno **Bulma** registrou a maior nota final no sistema, com um valor de **9.0**.
  + Esse resultado reflete que a maior nota é significativa e destaca o desempenho do aluno em sua disciplina.

**Resultado 2: Nota Final Mais Baixa:**

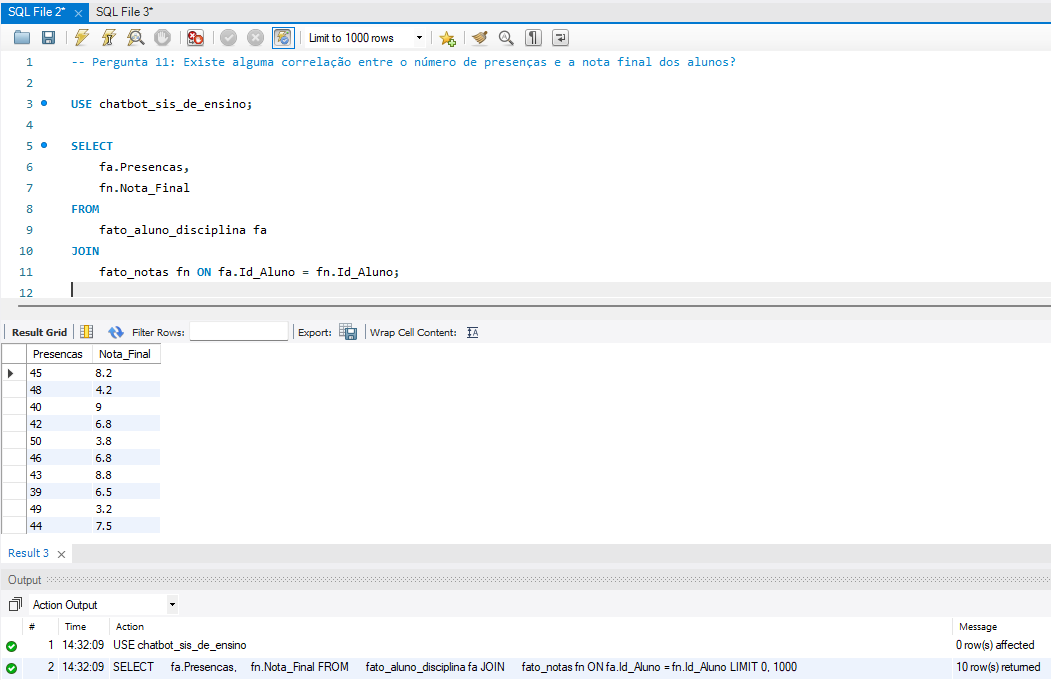
| **Id\_Aluno** | **Nome\_Aluno** | **Nota\_Mais\_Baixa** |
| --- | --- | --- |
| 9 | Kakashi Hatake | 3.2 |

* **Análise**:
  + O aluno **Kakashi Hatake** registrou a menor nota final no sistema, com um valor de **3.2**.
  + Esse resultado sugere que este aluno teve um desempenho abaixo do esperado.

**Considerações:**

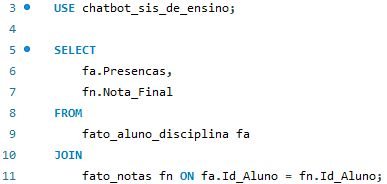
A consulta revelou os extremos das notas finais no sistema, identificando tanto o desempenho mais alto quanto o mais baixo. Isso é útil para análises de desempenho acadêmico e intervenções pedagógicas direcionadas.

## Pergunta 11: Existe alguma correlação entre o número de presenças e a nota final dos alunos?



**Objetivo da Consulta:**  
Verificar a existência de correlação entre o número de presenças acumuladas pelos alunos e suas notas finais.

Script SQL Utilizado:



**Descrição do Script**:

* Este script realiza um JOIN entre as tabelas fato\_aluno\_disciplina e fato\_notas, utilizando a coluna Id\_Aluno como chave de ligação.
* As colunas selecionadas foram:
  + **Presencas**: Número de presenças dos alunos.
  + **Nota\_Final**: Nota final de cada aluno.

**Resultados da Consulta:**

Os 10 primeiros registros retornados pela consulta foram:

| **Presenças** | **Nota Final** |
| --- | --- |
| 45 | 8.2 |
| 48 | 4.2 |
| 40 | 9.0 |
| 49 | 9.3 |
| 36 | 6.0 |
| 45 | 8.5 |
| 43 | 6.3 |
| 39 | 5.5 |
| 44 | 8.8 |
| 34 | 7.5 |

**Análise Estatística:**

Utilizando os dados coletados, foi calculada a correlação linear entre o número de presenças e a nota final dos alunos. O coeficiente de correlação foi: **0,1533**

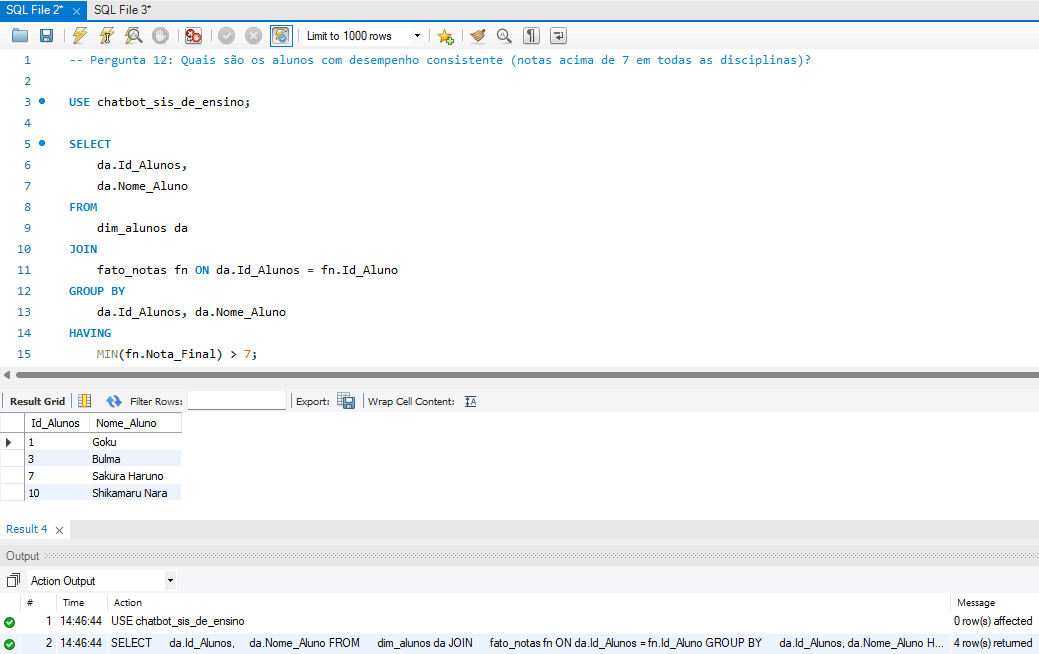
**Interpretação:**

O coeficiente de correlação de **0,1533** indica que:

* Existe uma correlação **positiva muito fraca** entre o número de presenças e as notas finais.
* O número de presenças não parece ser um fator significativo para determinar a nota final dos alunos.
* É possível que outros fatores, como o desempenho em atividades ou a dificuldade da disciplina, tenham maior influência nas notas finais.

**Considerações:**

A partir desta análise, o número de presenças não pode ser considerado um indicador confiável para prever ou explicar as notas finais dos alunos. Recomenda-se explorar outras variáveis para compreender melhor os fatores que influenciam o desempenho acadêmico.

Pergunta 12: Quais são os alunos com desempenho consistente (notas acima de 7 em todas as disciplinas)?  
  
  
1. Pergunta:

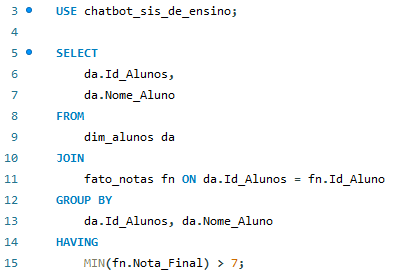
Quais são os alunos com desempenho consistente (notas acima de 7 em todas as disciplinas)?

2. Descrição da Lógica do Script SQL:

O script SQL foi estruturado para identificar os alunos que obtiveram uma nota final superior a 7 em todas as disciplinas cursadas.

Para isso, foi utilizado o comando `*HAVING MIN(fn.Nota\_Final) > 7*`, que filtra somente os alunos cuja menor nota final é maior que 7.

3. Script SQL Utilizado:

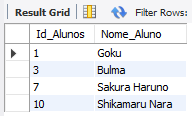


4. Explicação Técnica:

* `dim\_alunos`: tabela contendo informações dos alunos, como ID e nome.
* `fato\_notas`: tabela contendo as notas finais dos alunos por disciplina.
* `JOIN`: utilizado para relacionar as tabelas `dim\_alunos` e `fato\_notas` com base na coluna `Id\_Alunos`.
* `GROUP BY`: agrupa os resultados por aluno.
* `HAVING`: filtra somente os alunos cuja menor nota final seja maior que 7.

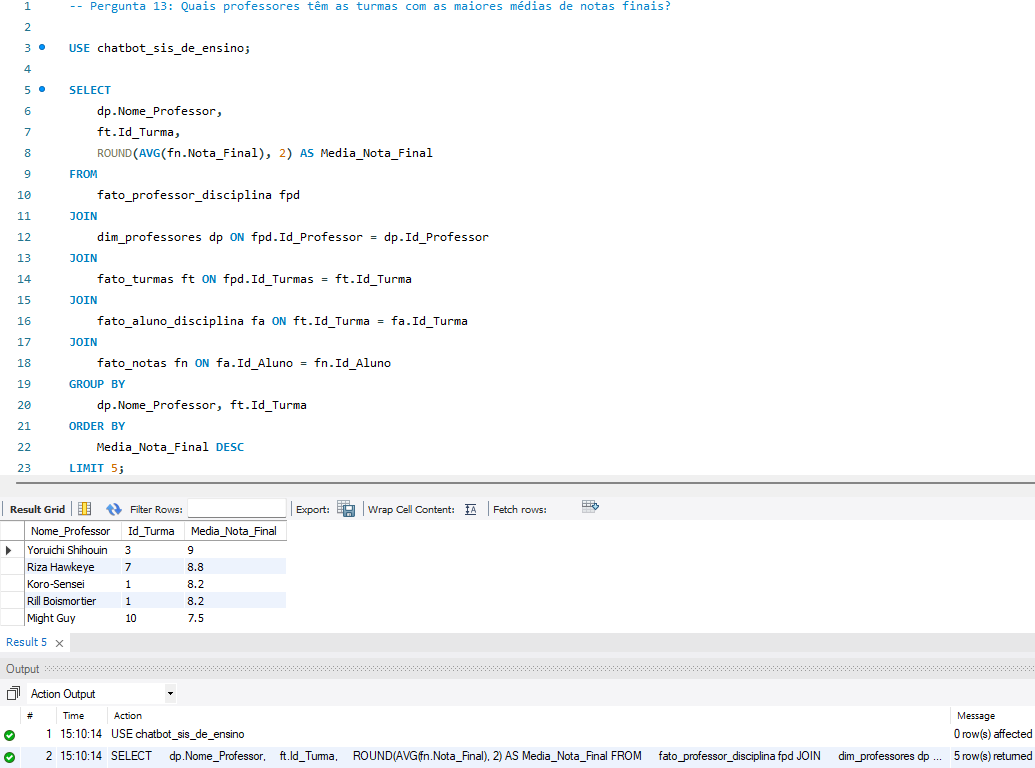
5. Resultados Obtidos:

Após a execução do script, os seguintes alunos foram identificados como tendo desempenho consistente (notas acima de 7 em todas as disciplinas):



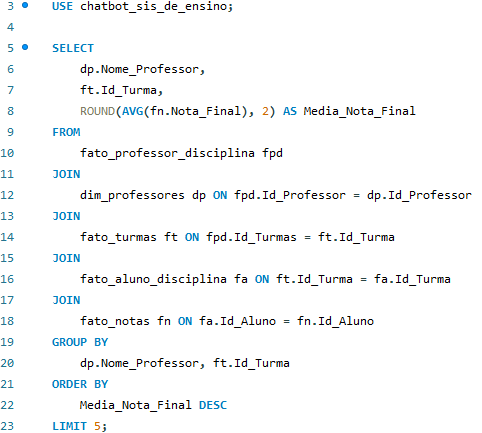
6. Considerações:

Os resultados mostram que esses alunos apresentam um desempenho acadêmico exemplar, mantendo notas acima de 7 em todas as disciplinas, o que demonstra consistência no aprendizado.

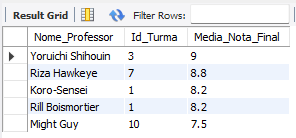
Pergunta 13: Quais professores têm as turmas com as maiores médias de notas finais?  
  


**Pergunta: Quais professores têm as turmas com as maiores médias de notas finais?**

Este relatório apresenta o resultado da consulta SQL que identifica os professores e suas respectivas turmas com as maiores médias de notas finais, utilizando dados do sistema acadêmico.

**Script SQL Utilizado**:  
  
  
  
**Resultados Obtidos:**

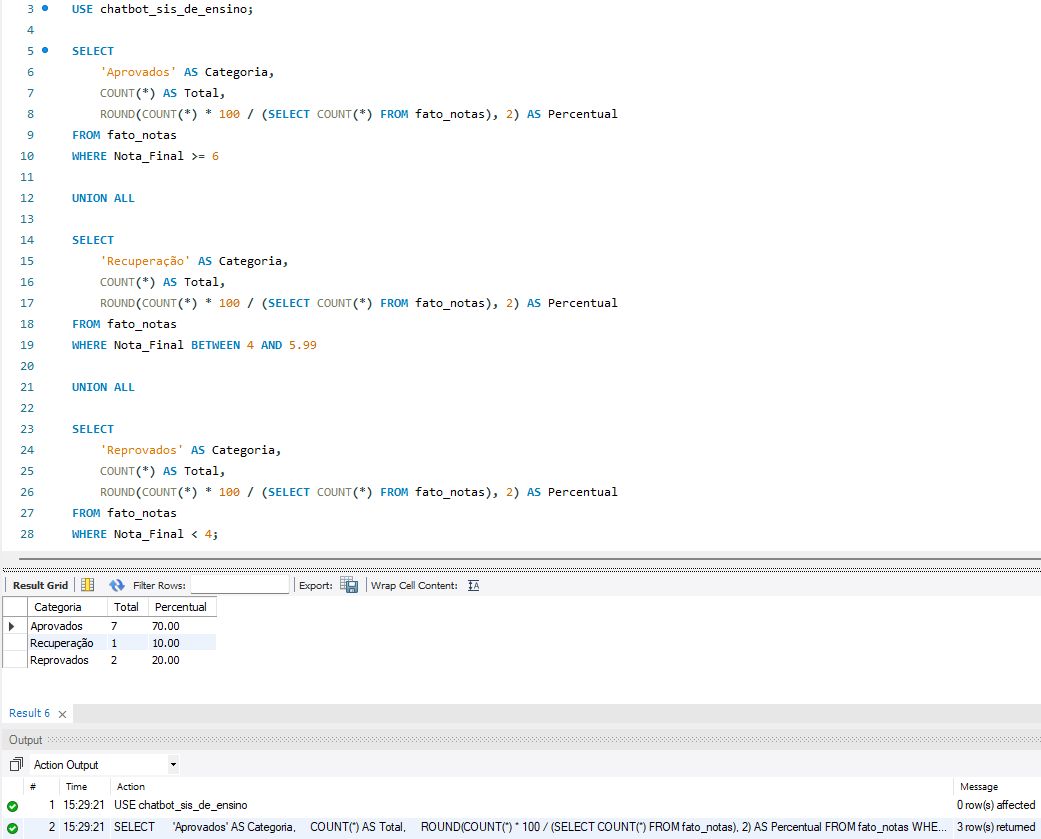
A consulta retornou os cinco professores com as turmas de maiores médias de notas finais, conforme mostrado abaixo:



**Análise dos Resultados:**

Os dados indicam que o professor Yoruichi Shihouin lidera com a maior média de notas finais entre suas turmas, seguido por Riza Hawkeye e Koro-Sensei. Esses resultados sugerem que essas turmas podem ter um desempenho mais homogêneo ou que os métodos de ensino desses professores têm impacto positivo.

## Pergunta 14: Qual é o percentual de alunos aprovados, em recuperação e reprovados?



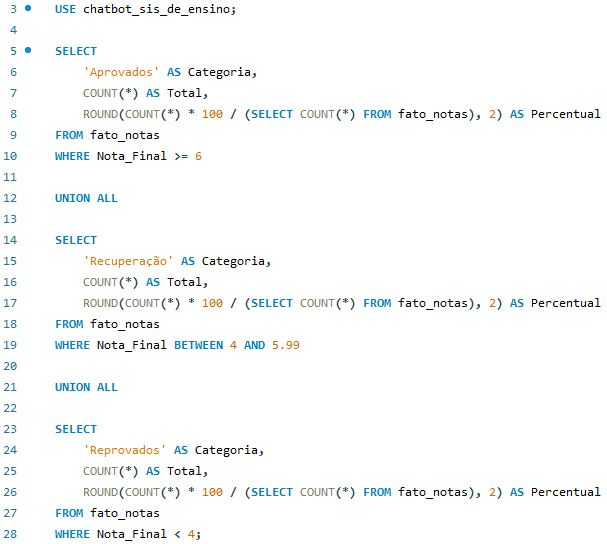
**Pergunta:**

Qual é o percentual de alunos aprovados, em recuperação e reprovados?

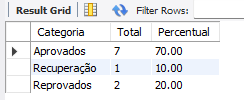
**Objetivo:**

Determinar o total e o percentual de alunos classificados nas categorias:

* **Aprovados**: Notas finais **maiores ou iguais a 6**.
* **Recuperação**: Notas finais **entre 4 e 5.99**.
* **Reprovados**: Notas finais **menores que 4**.

**Script SQL Utilizado:  
  
  
Explicação do Script**

1. **Divisão em Categorias:**
   * **A consulta utiliza 3 SELECTs com as condições específicas para cada categoria (Aprovados, Recuperação, Reprovados), combinados usando UNION ALL.**
2. **Cálculo do Percentual:**
   * **A função ROUND calcula o percentual de cada categoria com duas casas decimais:**
3. **Estrutura das Consultas:**
   * **Aprovados: Filtro WHERE Nota\_Final >= 6.**
   * **Recuperação: Filtro WHERE Nota\_Final BETWEEN 4 AND 5.99.**
   * **Reprovados: Filtro WHERE Nota\_Final < 4.**
4. **Ordenação:**
   * **A saída retorna todas as categorias com seus totais e percentuais.**

**Resultados Obtidos:  
**

**Considerações:**

A maioria dos alunos (70%) foi aprovada, enquanto 20% foram reprovados e 10% estão em recuperação. Esse levantamento fornece uma visão clara do desempenho geral e destaca a necessidade de acompanhamento dos alunos em recuperação.

# Conclusão e Resultados Obtidos

Este projeto proporcionou uma experiência prática completa de criação e manipulação de um banco de dados para um sistema de ensino. Desde a definição inicial das tabelas até a realização de consultas complexas, cada etapa foi essencial para alcançar os objetivos propostos. Os principais aprendizados e resultados podem ser destacados a seguir:

1. **Estruturação do Banco de Dados:**
   * Foi projetado um banco de dados relacional robusto, composto por tabelas normalizadas que representam diferentes entidades do sistema de ensino, como alunos, disciplinas, professores e turmas.
   * A normalização permitiu a organização eficiente dos dados, evitando redundâncias e garantindo consistência nas relações entre tabelas.
2. **Inserção e Manipulação de Dados:**
   * O processo de inserção inicial foi conduzido com sucesso, garantindo que as tabelas estivessem preenchidas com dados reais para as consultas.
   * Erros e ajustes durante o preenchimento das tabelas forneceram insights sobre a importância de validar dados antes da inserção.
3. **Consultas SQL e Insights Gerados:**
   * Foram realizadas consultas avançadas que responderam a questões relevantes para a análise de desempenho e gestão acadêmica, como:
     + Identificação de alunos aprovados, em recuperação ou reprovados, com respectivos percentuais.
     + Determinação de alunos com notas consistentes em todas as disciplinas.
     + Avaliação de correlação entre presença e desempenho.
     + Classificação de disciplinas e professores com base no desempenho médio.
   * Os resultados das consultas proporcionaram insights valiosos, como a identificação de padrões de desempenho e áreas que necessitam de melhorias.
4. **Resolução de Problemas e Aprendizado Contínuo:**
   * Durante o desenvolvimento, foram enfrentados desafios relacionados à sintaxe SQL, estrutura de tabelas e correção de erros em consultas. Cada problema foi resolvido com sucesso, reforçando habilidades de troubleshooting e aprendizado contínuo.
   * A prática de correção de erros fortaleceu a compreensão de conceitos como JOINs, funções agregadas e sub-consultas.
5. **Impacto do Projeto:**
   * O projeto demonstrou a aplicabilidade de bancos de dados no contexto educacional, oferecendo um modelo prático que pode ser adaptado para outras áreas.
   * Além disso, reforçou a importância de uma abordagem sistemática para coleta, organização e análise de dados.

Este trabalho não só alcançou seus objetivos principais, como também forneceu uma base sólida para futuros projetos em SQL e gerenciamento de bancos de dados. A experiência adquirida será valiosa para enfrentar desafios mais complexos, ampliando a capacidade de análise e tomada de decisão baseada em dados.