

Trabalho individual da disciplina FBD 2021.1

Aluno: Eduardo de Oliveira Castro

Matrícula: 210008164

Contextualização

Para este trabalho decidi explorar o CENSO ESCOLAR do INEP com o objetivo de obter alguma visão sobre a realidade das escolas de ensino básico do Distrito Federal.

Requisitos, Premissas e Obtenção dos Dados

Primeiramente, o CENSO ESCOLAR do INEP apresenta dados de todo o Brasil, motivo pelo qual os arquivos também possuem um tamanho extenso que dificulta o seu processamento em computadores pessoais convencionais. Por esse motivo, se mostrou necessário fazer um recorte dos dados para que fossem geradas tabelas contendo apenas dados do Distrito Federal e de toda a sua rede educacional, tanto pública quanto privada. Também foram explorados os relacionados entre Escolas, Turmas e Alunos, sendo dispensadas algumas informações e tabelas que não se mostraram relevantes para esta análise como é o caso de dados, por exemplo, das figuras dos gestores educacionais. Os dados foram obtidos por meio do link https://download.inep.gov.br/dados_abertos/microdados_censo_escolar_2020.zip, encontrado na página <https://www.gov.br/inep/pt-br/acesso-a-informacao/dados-abertos/microdados/censo-escolar>.

Análise preliminar dos dados

Os dados obtidos e estão dispostos em 5 arquivos .csv que, juntos, totalizam aproximadamente 2GB de informação, conforme imagem a seguir:

Nome	Data de modificação	Tipo	Tamanho
docentes_co	18/01/2021 17:45	Arquivo de Valores Separados por Vírgulas do Microsoft Excel	278.670 KB
escolas	18/01/2021 11:08	Arquivo de Valores Separados por Vírgulas do Microsoft Excel	108.666 KB
gestor	21/01/2021 08:58	Arquivo de Valores Separados por Vírgulas do Microsoft Excel	42.241 KB
matricula_co	18/01/2021 21:34	Arquivo de Valores Separados por Vírgulas do Microsoft Excel	941.934 KB
MD5_microdados_ed_basica_2020	21/01/2021 09:34	Documento de Texto	1 KB
turmas	18/01/2021 14:37	Arquivo de Valores Separados por Vírgulas do Microsoft Excel	439.969 KB

O INEP também oferece, juntamente com os dados, um excelente dicionário de dados por meio do qual utilizei para compreender com quais dados estou lidando, o que representam os valores de referência nas tabelas, etc. O dicionário está disposto na imagem a seguir:

N	Nome da Variável	Descrição da Variável	Tipo	Tam. ⁽¹⁾	Categoria ⁽²⁾	Coleta por ano ("sssim","não")															Notas sobre diferenças entre os anos
						07	08	09	10	11	12	13	14	15	16	17	18	19	20		
1	NU_ANO_CENSO	Ano do Censo	Num	4		S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S		
2	CO_ENTIDADE	Código da Escola	Num	8		S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S		
3	NO_ENTIDADE	Nome da Escola	Char	100																	
4	CO_ORGAO REGIONAL	Código do Órgão Regional de Ensino	Char	5		S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S		
5	TP_SITUACAO_FUNCIONAMENTO	Situação de funcionamento	Num	1	1 - Em Atividade 2 - Paralisada 3 - Extinta (ano do Censo) 4 - Extinta em Anos Anteriores	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S		
6	DT_ANO_INICIO	Ínicio do ano letivo	Data			- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S		
7	DT_ANO_TERMINO	Término (Previsão) do ano letivo	Data			- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	- S S S S	
8	CO_REGIAO	Código da região geográfica	Num	1		S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	
9	CO_MESORREGIAO	Código da mesorregião	Num	4		S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	
10	CO_MICROREGIAO	Código da microrregião	Num	5		S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	
11	CO_UF	Código da UF	Num	2		S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	
12	CO_MUNICIPIO	Código do Município	Num	7		S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	
13	CO_DISTRITO	Código completo do Distrito da escola	Num	9		S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	
14	TP_DEPENDENCIA	Dependência Administrativa	Num	1	1 - Federal 2 - Estadual 3 - Municipal 4 - Privada	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	
15	TP_LOCALIZACAO	Localização	Num	1	1 - Urbana 2 - Rural	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	
16	TP_LOCALIZACAO_DIFERENCIADA	Localização diferenciada da escola	Num	1	0 - A escola não está em área de localização diferenciada 1 - Área de assentamento 2 - Terra indígena 3 - Área onde se localiza comunidade remanescente de quilombos	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	S S S S	<small>De 2007 a 2015: 0 - Não se aplica 1 - Área de assentamento 2 - Terra indígena 3 - Área remanescente de quilombos De 2016 a 2018: 0 - A escola não está em área de localização diferenciada 1 - Área de assentamento 2 - Terra indígena 3 - Área remanescente de quilombos 4 - Unidade de uso sustentável 5 - Unidade de uso sustentável em terra indígena 6 - Unidade de uso sustentável em área remanescente de quilombos</small>	
17	IN_VINCULO_SECRETARIA EDUCACAO	Órgão que a escola pública está vinculada - Secretaria de Educação/Ministério da Educação	Num	1	0 - Não 1 - Sim - Não aplicável para escolas privadas	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	S S		
18	IN_VINCULO_SEGURANCA PUBLICA	Órgão que a escola pública está vinculada - Secretaria de Segurança Pública/Forças Armadas/Militar	Num	1	0 - Não 1 - Sim - Não aplicável para escolas privadas	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	S S		
19	IN_VINCULO_SECRETARIA SAUDE	Órgão que a escola pública está vinculada - Secretaria de Saúde/Ministério da Saúde	Num	1	0 - Não 1 - Sim - Não aplicável para escolas privadas	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	- - - - -	S S		
		Órgão que a escola pública está vinculada - Outro órgão da			0 - Não 1 - Sim																
<small>BAS_ESCOLA BAS_MATRICULA BAS_TURMA BAS_DOCENTE BAS_GESTOR (1)</small>																					

Limpeza e Transformação dos Dados

Naturalmente, o primeiro objetivo do trabalho envolveu o processo de ETL (Extract, transform, load ou, em português, extração, transformação, carregamento). Na etapa atual, avaliamos as colunas que são pertinentes para o trabalho e retiramos aquelas que não foram relevantes. Também procedemos com ajustes em dados inconsistentes e/ou vazios por meio da sua devida correção e/ou preenchimento para que tenhamos uma tabela completamente preenchida e com valores válidos. Nesta etapa, todos aqueles campos vazios foram preenchidos com o valor 0. O código desenvolvido, em Python, pode ser verificado a seguir:

```
In [1]: # Importação da biblioteca utilizada para manipular os dados
import pandas as pd

In [2]: # Carregamento do maior csv, o de alunos/matrículas. Isso é feito agora para que não haja
alunos = pd.read_csv(r'C:\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.csv')

In [3]: # Carregando do arquivo referente as escolas
escolas = pd.read_csv(r'C:\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.csv')

# Limpeza exclusão das tabelas que não foram consideradas relevantes para este trabalho
escolas = escolas.drop(columns=['NU_ANO_CENSO', 'CO_REGIAO', 'CO_MESORREGIAO', 'CO_MICR...', 'CO_DISTRITO'])

# Fazendo apenas parte as escolas cuja UF seja equivalente ao número 53, que representa
escolas_df = escolas[escolas['CO_UF'] == 53]

# Preenche todas as células vazias com o valor 0
escolas_df = escolas_df.fillna(0)

# Cria uma lista com os códigos das escolas que estão no Distrito Federal, esse código
escolas_df_codigos = escolas_df['CO_ENTIDADE'].values.tolist()

# Salva uma nova versão do CSV, contendo apenas as colunas consideradas relevantes e os
escolas_df.to_csv(r'C:\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.csv')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3165: DtypeWarning: Columns (3) have mixed types.Specify dtype option on import or set low_memory=False.  
    has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

In [4]:

```
# Carregando do arquivo referente aos docentes  
docentes = pd.read_csv(r'C:\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\  
  
# Limpeza exclusão das tabelas que não foram consideradas relevantes para este trabalho  
docentes = docentes.drop(columns=['NU_ANO_CENSO', 'NU_IDADE_REFERENCIA', 'CO_PAIS_ORIGEM'])  
  
# Cria um novo dataframe apenas contendo os docentes cuja escola seja do DF, para tal com  
docentes_df = docentes[docentes['CO_ENTIDADE'].isin(escolas_df_codigos)]  
  
# Preenche todas as células vazias com o valor 0  
docentes_df = docentes_df.fillna(0)  
  
# Salva uma nova versão do CSV, contendo apenas as colunas consideradas relevantes e os  
docentes_df.to_csv(r'C:\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.1.csv')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3165: DtypeWarning: Columns (45) have mixed types.Specify dtype option on import or set low_memory=False.  
    has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

In [5]:

```
# Carregando do arquivo referente aos gestores  
gestor = pd.read_csv(r'C:\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.1.csv')  
  
# Limpeza exclusão das tabelas que não foram consideradas relevantes para este trabalho  
gestor = gestor.drop(columns=['NU_ANO_CENSO', 'NU_IDADE_REFERENCIA', 'CO_PAIS_ORIGEM', 'CO_GESTOR'])  
  
# Cria um novo dataframe apenas contendo os gestores cuja escola seja do DF, para tal com  
gestor_df = gestor[gestor['CO_ENTIDADE'].isin(escolas_df_codigos)]  
  
# Preenche todas as células vazias com o valor 0  
gestor_df = gestor_df.fillna(0)  
  
# Salva uma nova versão do CSV, contendo apenas as colunas consideradas relevantes e os  
gestor_df.to_csv(r'C:\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.1.csv')
```

In [6]:

```
# Carregando do arquivo referente as turmas  
turma = pd.read_csv(r'C:\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.1.csv')  
  
# Limpeza exclusão das tabelas que não foram consideradas relevantes para este trabalho  
turma = turma.drop(columns=['NU_ANO_CENSO', 'NU_DURACAO_TURMA', 'TP_TIPO_ATENDIMENTO_TURMA'])  
  
# Cria um novo dataframe apenas contendo as turmas cuja escola seja do DF, para tal com  
turma_df = turma[turma['CO_ENTIDADE'].isin(escolas_df_codigos)]  
  
# Preenche todas as células vazias com o valor 0  
turma_df = turma_df.fillna(0)  
  
# Salva uma nova versão do CSV, contendo apenas as colunas consideradas relevantes e os  
turma_df.to_csv(r'C:\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.1.csv')
```

In [7]:

```
# Limpeza exclusão das tabelas que não foram consideradas relevantes para este trabalho
```

```

alunos = alunos.drop(columns=['NU_ANO_CENSO', 'NU_IDADE_REFERENCIA', 'CO_PAIS_ORIGEM',  

# Cria um novo dataframe apenas contendo os alunos cuja escola seja do DF, para tal com  

alunos_df = alunos[alunos['CO_ENTIDADE'].isin(escolas_df_codigos)]  

# Preenche todas as células vazias com o valor 0  

alunos_df = alunos_df.fillna(0)  

# Salva uma nova versão do CSV, contendo apenas as colunas consideradas relevantes e os  

alunos_df.to_csv(r'C:\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.1'

```

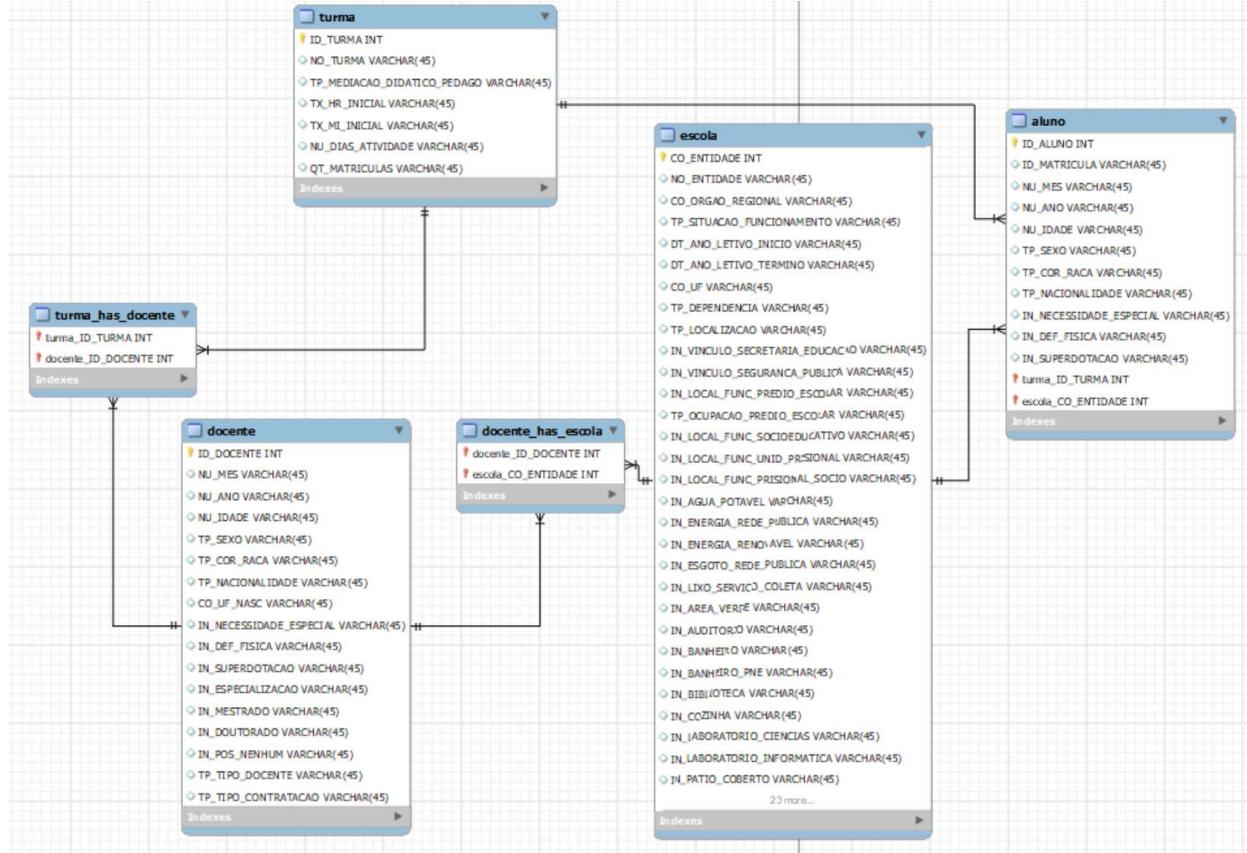
Volume de Dados

Desta forma, temos os seguintes volumes de dados em cada uma das tabelas que serão utilizadas para alimentar o banco de dados:

- Tabela Escolas, com 1372 linhas e 52 colunas;
- Tabela Docentes, 151.384 linhas e 20 colunas;
- Tabela Docentes, com 1.301 linhas e 16 colunas;
- Tabela Turma, com 34.626 linhas e 9 colunas;
- E, por fim, Tabela Aluno, com 724.465 linhas e 14 colunas;

A modelagem do Banco de Dados

Para a modelagem do banco foi utilizada a ferramenta MySQL Workbench e os dados provenientes das tabelas citadas anteriormente. Também utilizei como referência o dicionário de dados provido pelo próprio INEP. A seguir segue uma imagem que representa o modelo lógico desenvolvido:



Os requisitos de negócio observados

Ademais, ao observar as planilhas, o dicionário dos dados e as dinâmicas de gestão educacional, fomos capazes de observar algumas relações entre as entidades que representam a forma como esses elementos se organizam e como eles estão dispostos nas tabelas, tais como:

- Cada escola contém n alunos, n docentes e n turmas;
- Cada docente possui n turmas e n escolas, uma vez que podem lecionar em algumas delas e em várias turmas ao mesmo tempo; e
- Cada aluno possui uma escola e uma turma.

O script de criação do Banco de Dados

```

-- 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_AUTO_CREATE_USER,NO_ENGINE_NAME_CHECK';

```

```
-- Schema mydb
```

```
-- Schema mydb
```

```
-- -----  
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;  
USE `mydb` ;  
  
-- -----  
-- Table `mydb`.`escola`  
-- -----  
CREATE TABLE IF NOT EXISTS `mydb`.`escola` (  
`CO_ENTIDADE` INT NOT NULL,  
`NO_ENTIDADE` VARCHAR(200) NULL,  
`CO_ORGAO_REGIONAL` VARCHAR(10) NULL,  
`TP_SITUACAO_FUNCIONAMENTO` INT NULL,  
`DT_ANO_LETIVO_INICIO` VARCHAR(45) NULL,  
`DT_ANO_LETIVO_TERMINO` VARCHAR(45) NULL,  
`CO_UF` INT NULL,  
`TP_DEPENDENCIA` INT NULL,  
`TP_LOCALIZACAO` INT NULL,  
`IN_VINCULO_SECRETARIA_EDUCACAO` INT NULL,  
`IN_VINCULO_SEGURANCA_PUBLICA` INT NULL,  
`IN_LOCAL_FUNC_PREDIO_ESCOLAR` INT NULL,  
`TP_OCUPACAO_PREDIO_ESCOLAR` INT NULL,  
`IN_LOCAL_FUNC_SOCIOEDUCATIVO` INT NULL,  
`IN_LOCAL_FUNC_UNID_PRISIONAL` INT NULL,  
`IN_LOCAL_FUNC_PRISIONAL_SOCIO` INT NULL,  
`IN_AGUA_POTAVEL` INT NULL,  
`IN_ENERGIA_REDE_PUBLICA` INT NULL,  
`IN_ENERGIA_RENOVAVEL` INT NULL,  
`IN_ESGOTO_REDE_PUBLICA` INT NULL,  
`IN_LIXO_SERVICO_COLETA` INT NULL,  
`IN_AREA_VERDE` INT NULL,  
`IN_AUDITORIO` INT NULL,  
`IN_BANHEIRO` INT NULL,  
`IN_BANHEIRO_PNE` INT NULL,  
`IN_BIBLIOTECA` INT NULL,  
`IN_COZINHA` INT NULL,  
`IN_LABORATORIO_CIENCIAS` INT NULL,  
`IN_LABORATORIO_INFORMATICA` INT NULL,  
`IN_PATIO_COBERTO` INT NULL,  
`IN_PATIO_DESCOBERTO` INT NULL,  
`IN_PARQUE_INFANTIL` INT NULL,  
`IN_PISCINA` INT NULL,  
`IN_QUADRA_ESPORTES` INT NULL,  
`IN_QUADRA_ESPORTES_COBERTA` INT NULL,  
`IN_QUADRA_ESPORTES_DESCOBERTA` INT NULL,  
`IN_REFEITORIO` INT NULL,  
`IN_SALA_ATELIE_ARTES` INT NULL,  
`IN_DESKTOP_ALUNO` INT NULL,  
`QT_DESKTOP_ALUNO` INT NULL,  
`IN_COMP_PORTATIL_ALUNO` INT NULL,  
`QT_COMP_PORTATIL_ALUNO` INT NULL,  
`IN_TABLET_ALUNO` INT NULL,  
`QT_TABLET_ALUNO` INT NULL,  
`IN_INTERNET` INT NULL,  
`IN_INTERNET_ALUNOS` INT NULL,
```

```

`TP_REDE_LOCAL` INT NULL,
`IN_BANDA_LARGA` INT NULL,
`QT_PROF_FONAUDIOLOGO` INT NULL,
`QT_PROF_NUTRICIONISTA` INT NULL,
`QT_PROF_PSICOLOGO` INT NULL,
`IN_ALIMENTACAO` INT NULL,
PRIMARY KEY (`CO_ENTIDADE`))
ENGINE = InnoDB;

-----  

-- Table `mydb`.`turma`  

-----  

CREATE TABLE IF NOT EXISTS `mydb`.`turma` (
`ID_TURMA` INT NOT NULL,
`NO_TURMA` VARCHAR(200) NULL,
`TP_MEDIACAO_DIDATICO_PEDAGO` INT NULL,
`TX_HR_INICIAL` VARCHAR(45) NULL,
`TX_MI_INICIAL` VARCHAR(45) NULL,
`NU_DIAS_ATIVIDADE` INT NULL,
`QT_MATRICULAS` INT NULL,
PRIMARY KEY (`ID_TURMA`))
ENGINE = InnoDB;

-----  

-- Table `mydb`.`aluno`  

-----  

CREATE TABLE IF NOT EXISTS `mydb`.`aluno` (
`ID_ALUNO` VARCHAR(45) NOT NULL,
`ID_MATRICULA` INT NULL,
`NU_MES` INT NULL,
`NU_ANO` INT NULL,
`NU_IDADE` INT NULL,
`TP_SEXO` INT NULL,
`TP_COR_RACA` INT NULL,
`TP_NACIONALIDADE` INT NULL,
`IN_NECESSIDADE_ESPECIAL` INT NULL,
`IN_DEF_FISICA` INT NULL,
`IN_SUPERDOTACAO` INT NULL,
`turma_ID_TURMA` INT NOT NULL,
`escola_CO_ENTIDADE` INT NOT NULL,
PRIMARY KEY (`ID_ALUNO`, `turma_ID_TURMA`, `escola_CO_ENTIDADE`),
INDEX `fk_aluno_turma1_idx` (`turma_ID_TURMA` ASC) VISIBLE,
INDEX `fk_aluno_escola1_idx` (`escola_CO_ENTIDADE` ASC) VISIBLE,
CONSTRAINT `fk_aluno_turma1`
    FOREIGN KEY (`turma_ID_TURMA`)
    REFERENCES `mydb`.`turma` (`ID_TURMA`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
CONSTRAINT `fk_aluno_escola1`
    FOREIGN KEY (`escola_CO_ENTIDADE`)
    REFERENCES `mydb`.`escola` (`CO_ENTIDADE`)
    ON DELETE NO ACTION

```

```

        ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----
-- Table `mydb`.`docente`


CREATE TABLE IF NOT EXISTS `mydb`.`docente` (
  `idDocente` INT NOT NULL AUTO_INCREMENT,
  `ID_DOCENTE` VARCHAR(45) NULL,
  `NU_MES` INT NULL,
  `NU_ANO` INT NULL,
  `NU_IDADE` INT NULL,
  `TP_SEXO` INT NULL,
  `TP_COR_RACA` INT NULL,
  `TP_NACIONALIDADE` INT NULL,
  `CO_UF_NASC` INT NULL,
  `IN_NECESSIDADE_ESPECIAL` INT NULL,
  `IN_DEF_FISICA` INT NULL,
  `IN_SUPERDOTACAO` INT NULL,
  `IN_ESPECIALIZACAO` INT NULL,
  `IN_MESTRADO` INT NULL,
  `IN_DOUTORADO` INT NULL,
  `IN_POS_NENHUM` INT NULL,
  `TP_TIPO_DOCENTE` INT NULL,
  `TP_TIPO_CONTRATACAO` INT NULL,
  PRIMARY KEY (`idDocente`)
)
ENGINE = InnoDB;

-- -----
-- Table `mydb`.`docente_has_turma`


CREATE TABLE IF NOT EXISTS `mydb`.`docente_has_turma` (
  `docente_idDocente` INT NOT NULL,
  `turma_ID_TURMA` INT NOT NULL,
  PRIMARY KEY (`docente_idDocente`, `turma_ID_TURMA`),
  INDEX `fk_docente_has_turma_turma1_idx` (`turma_ID_TURMA` ASC) VISIBLE,
  INDEX `fk_docente_has_turma_docente1_idx` (`docente_idDocente` ASC) VISIBLE,
  CONSTRAINT `fk_docente_has_turma_docente1`
    FOREIGN KEY (`docente_idDocente`)
    REFERENCES `mydb`.`docente` (`idDocente`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_docente_has_turma_turma1`
    FOREIGN KEY (`turma_ID_TURMA`)
    REFERENCES `mydb`.`turma` (`ID_TURMA`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----
-- Table `mydb`.`docente_has_escola`
```

```

-----  

CREATE TABLE IF NOT EXISTS `mydb`.`docente_has_escola` (  

    `docente_idDocente` INT NOT NULL,  

    `escola_CO_ENTIDADE` INT NOT NULL,  

    PRIMARY KEY (`docente_idDocente`, `escola_CO_ENTIDADE`),  

    INDEX `fk_docente_has_escola_escola1_idx` (`escola_CO_ENTIDADE` ASC)  

VISIBLE,  

    INDEX `fk_docente_has_escola_docente1_idx` (`docente_idDocente` ASC)  

VISIBLE,  

    CONSTRAINT `fk_docente_has_escola_docente1`  

        FOREIGN KEY (`docente_idDocente`)  

        REFERENCES `mydb`.`docente` (`idDocente`)  

        ON DELETE NO ACTION  

        ON UPDATE NO ACTION,  

    CONSTRAINT `fk_docente_has_escola_escola1`  

        FOREIGN KEY (`escola_CO_ENTIDADE`)  

        REFERENCES `mydb`.`escola` (`CO_ENTIDADE`)  

        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;

```

In [2]:

```
# Importação das bibliotecas responsáveis com a conexão e manipulação do banco de dados
import mysql.connector as msql
from mysql.connector import Error
```

Script para inserção dos dados na tabela escola, desenvolvido em Python

In [8]:

```
# Exceção para caso haja algum erro na inserção dos dados.
try:
    # Conexão com o banco de dados MySQL criado localmente para o trabalho.
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'
    conn = msql.connect(host='localhost', database='mydb', user='root', password='root')

    # Procede caso a conexão com o banco tenha sido bem sucedida
    if conn.is_connected():
        # Cursor é a instância criada para a manipulação do banco
        cursor = conn.cursor()
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("You're connected to database: ", record)

    # For Loop em toda a tabela csv escolas_df gerada anteriormente para que possam
    for i, row in escolas_df.iterrows():
        # Execução do comando INSERT para inserção dos elementos no banco MySQL, de
        # A inserção é feita conforme parâmetros oferecidos e é rodado para cada li
        # Dessa forma inserimos todas as 1372 escolas no banco MySQL
```

```

cursor.execute("""INSERT INTO escola (CO_ENTIDADE, NO_ENTIDADE, CO_ORGAO_RE

    # A conexão não faz o commit das inserções como padrão, para salvar os dados
    conn.commit()
    print("Registros inseridos")
except Error as e:
    print("Error while connecting to MySQL", e)

```

You're connected to database: ('mydb',)
Registros inseridos

Script para inserção dos dados na tabela turma, desenvolvido em Python

In [9]:

```

# Exceção para caso haja algum erro na inserção dos dados.
try:
    # Conexão com o banco de dados MySQL criado Localmente para o trabalho.
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root'

    # Procede caso a conexão com o banco tenha sido bem sucedida
    if conn.is_connected():
        # Cursor é a instância criada para a manipulação do banco
        cursor = conn.cursor()
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("You're connected to database: ", record)

        # For Loop em toda a tabela csv escolas_df gerada anteriormente para que possam
        for i, row in turma_df.iterrows():
            # Execução do comando INSERT para inserção dos elementos no banco MySQL, de
            # A inserção é feita conforme parâmetros oferecidos e é rodado para cada linha
            # Dessa forma inserimos todas as 1372 escolas no banco MySQL.
            cursor.execute("""INSERT INTO turma (ID_TURMA, NO_TURMA, TP_MEDIACAO_DIDATI
                # A conexão não faz o commit das inserções como padrão, para salvar os dados
                conn.commit()
                print("Registros inseridos")
except Error as e:
    print("Error while connecting to MySQL", e)

```

You're connected to database: ('mydb',)
Registros inseridos

Script para inserção dos dados na tabela aluno, desenvolvido em Python

In [10]:

```

# Exceção para caso haja algum erro na inserção dos dados.
try:
    # Conexão com o banco de dados MySQL criado Localmente para o trabalho.
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root'

    # Procede caso a conexão com o banco tenha sido bem sucedida
    if conn.is_connected():
        # Cursor é a instância criada para a manipulação do banco
        cursor = conn.cursor()
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente

```

```

cursor.execute("select database();")
record = cursor.fetchone()
print("You're connected to database: ", record)

# For Loop em toda a tabela csv escolas_df gerada anteriormente para que possam
for i, row in alunos_df.iterrows():
    # Execução do comando INSERT para inserção dos elementos no banco MySQL, de
    # A inserção é feita conforme parâmetros oferecidos e é rodado para cada Li
    # Dessa forma inserimos todas as 1372 escolas no banco MySQL.
    cursor.execute("""INSERT INTO aluno (ID_ALUNO, ID_MATRICULA, NU_MES, NU_ANO)
    # A conexão não faz o commit das inserções como padrão, para salvar os dado
    conn.commit()
    print("Registros inseridos")
except Error as e:
    print("Error while connecting to MySQL", e)

```

You're connected to database: ('mydb',)
Registros inseridos

Script para inserção dos dados na tabela docente, turma_has_docente e docente_has_escola, desenvolvido em Python

In [11]:

```

# Exceção para caso haja algum erro na inserção dos dados.
try:
    # Conexão com o banco de dados MySQL criado localmente para o trabalho.
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root')

    # Procede caso a conexão com o banco tenha sido bem sucedida
    if conn.is_connected():
        # Cursor é a instância criada para a manipulação do banco
        cursor = conn.cursor()
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("You're connected to database: ", record)

        # For Loop em toda a tabela csv escolas_df gerada anteriormente para que possam
        for i, row in docentes_df.iterrows():
            # Execução do comando INSERT para inserção dos elementos no banco MySQL, de
            # A inserção é feita conforme parâmetros oferecidos e é rodado para cada Li
            # Dessa forma inserimos todas as 1372 escolas no banco MySQL.
            cursor.execute("""INSERT INTO docente (ID_DOCENTE, NU_MES, NU_ANO, NU_IDADE
            # Como aqui estamos trabalhando com tabelas que contém associação precisamo
            primary_key = cursor.lastrowid
            cursor.execute("""INSERT INTO docente_has_turma (docente_idDocente, turma_I
            cursor.execute("""INSERT INTO docente_has_escola (docente_idDocente, escola_
            # A conexão não faz o commit das inserções como padrão, para salvar os dado
            conn.commit()
            print("Registros inseridos")
except Error as e:
    print("Error while connecting to MySQL", e)

```

You're connected to database: ('mydb',)
Registros inseridos

Printscreen dos dados nas tabelas

Após esse extenso trabalho somos capazes de observar a inserção dos dados no banco por meio da ferramenta Workbench e conforme imagens a seguir:

- Tabela Aluno

```
1 • SELECT * FROM mydb.aluno;
```

ID_ALUNO	ID_MATRICULA	NU_MES	NU_ANO	NU_IDADE	TP_SEXO	TP_COR_RACA	TP_NACIONALIDADE	IN_NECESSIDADE_ESPECIAL	IN_DEF_FISICA	IN_SUPERDOTACAO	turma_ID_TUF
000005ED95C92554CC88D8C4ECE7C72489	423128638	11	2003	17	2	0	1	0	0	0	17560541
00002723B58A75E6CC0B93FBA2F6C68C	439932727	9	2008	12	1	1	1	0	0	0	286076
000028380A36A968EE258EF0922DEF	419451656	4	2017	3	2	3	1	0	0	0	16654257
00002A32C0650EF1425113FE7738732E	386156007	8	2003	17	2	3	1	0	0	0	17597844
0000391D258F084939608AA72DEFACFA	429647295	5	2010	10	2	2	1	0	0	0	19808283
000058367EFC97976F7769A3F75C5C6	396145279	5	2003	17	1	1	1	0	0	0	385022
00006F8CB9F058D600AD14131CE1879	331581792	4	2006	14	2	1	1	0	0	0	6797813
00009478CF0A2046C61182D9485FETE53	435343973	3	2014	6	1	0	1	0	0	0	18949310
000090ED77A10F5F3725F138824C4B04	384622897	7	2005	15	2	1	1	0	0	0	17374174
0000A588648C4C2AA2E7802468819F07	363211385	11	2012	8	1	3	1	0	0	0	9040524
00001043518DE8C8A330902A4E5C6795A	284761099	4	2005	15	1	0	1	0	0	0	17328656
00010696C02F954186FPC001AB6F38D2	405801426	1	2005	15	2	1	1	0	0	0	15064934
0001231D5091C6662E1D5435700A90D	440167496	12	2009	11	1	3	1	0	0	0	19808414
00012987800A64637645B433EAD1AD	387702483	11	2004	16	1	0	1	0	0	0	158218
000131P0D831990D385246FCD274E7	249402744	2	2010	10	1	3	1	0	0	0	728608
000139988E73A72DC30FFC6B27813B4B	352006122	4	2000	20	2	2	1	0	0	0	3197443
00014780C5D65F93C85DC4603D6588D	400397642	8	1992	28	2	0	1	0	0	0	18956014
00014E9E8924ABA8CFED33D50831CCD0	356259363	5	2016	4	1	1	1	0	0	0	16761944
0001568FCA6625118743C3580E1D5	326947784	6	2011	9	2	1	1	0	0	0	14553538
00017205091C67C151679A3594C1D5	419828822	8	2008	12	1	3	1	0	0	0	18927981
0001A6A33C712F0384C4E3622914C62D	396114840	12	2008	12	1	0	1	0	0	0	17691391
000188E4E29FAB2F4E2D2E33C51E745A	338393912	5	1999	21	2	3	1	0	0	0	3055987
0001E1C0F1B22DF2DC89121D000E13	332630355	2	2015	5	2	1	1	0	0	0	17370668
000217A50B5131C1DD0259963D68EAC1D	440472233	4	2017	3	1	0	1	0	0	0	16608172
000221D32E5CEC1417D47A860FC07D9A95	383635053	1	2007	13	1	3	1	0	0	0	17373291
0002265E83F2A3238895345F7A50	411172510	10	2010	10	2	3	1	0	0	0	1198826
0002265E83F2A3238895345F7A50B	395687936	10	2010	10	2	3	1	0	0	0	14573766
0002349F7F792C952F7AF1BF29692321	351983136	7	1999	21	2	3	1	0	0	0	3047984
0002441B7F854F9953E88A00FB0454	165814849	2	2009	11	2	1	1	0	0	0	14633083
0002820B2000D2BAE3D413A8F281AF	449437837	11	2004	16	1	0	1	0	0	0	17496570
00028E174FF48D30C01177EA9FD0F94	381000362	6	2008	12	1	3	1	0	0	0	1732749
0002A738F7A5E2A30EABF8E8DEFAAC	382259851	7	2005	15	1	3	1	0	0	0	17350738
0002C0AB7D576D279FDD5C68323A3BAA	425702289	8	2013	7	1	0	1	0	0	0	19434510
0002CFC950F0A225D4C409099A92D16	331600472	2	2007	13	2	0	1	0	0	0	17420390

- Tabela Docente

```
1 • SELECT * FROM mydb.docente;
```

idDocente	ID_DOCENTE	NU_MES	NU_ANO	NU_IDADE	TP_SEXO	TP_COR_RACA	TP_NACIONALIDADE	CO_UF_NASC	IN_NECESSIDADE_ESPECIAL	IN_DEF_FISICA	IN_SUPERDOTACAO	IN
1	0001D106AC54F2E10CEEC8AD85336E3	4	1983	37	2	3	1	31	0	0	0	0
2	0001D106AC54F2E10CEEC8AD85336E3	4	1983	37	2	3	1	31	0	0	0	0
3	0001D106AC54F2E10CEEC8AD85336E3	4	1983	37	2	3	1	31	0	0	0	0
4	0001D106AC54F2E10CEEC8AD85336E3	4	1983	37	2	3	1	31	0	0	0	0
5	0001D106AC54F2E10CEEC8AD85336E3	4	1983	37	2	3	1	31	0	0	0	0
6	000239A6D52D1458ACF2ZD047D0E01	1	1974	46	2	0	1	53	0	0	0	0
7	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
8	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
9	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
10	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
11	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
12	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
13	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
14	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
15	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
16	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
17	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
18	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
19	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
20	0002B50B05A7D512E3968C8C45466C	10	1968	52	1	0	1	53	0	0	0	0
21	0008DFEB0C3CCB0351C866805E0C79	10	1979	41	2	1	1	53	0	0	0	1
22	0008DFEB0C3CCB0351C866805E0C79	10	1979	41	2	1	1	53	0	0	0	1
23	0008DFEB0C3CCB0351C866805E0C79	10	1979	41	2	1	1	53	0	0	0	1
24	0008DFEB0C3CCB0351C866805E0C79	10	1979	41	2	1	1	53	0	0	0	1
25	0008DFEB0C3CCB0351C866805E0C79	10	1979	41	2	1	1	53	0	0	0	1
26	00085989F7C7068ECE37FEC78078F	2	1971	49	2	1	1	52	0	0	0	1
27	00086F54018C8213E5600CD42A4B3D9	9	1963	57	2	0	1	53	0	0	0	1
28	0008C821A1DFBD0641D45F94622502085	6	1981	39	2	1	1	22	0	0	0	0
29	0008C821A1DFBD0641D45F94622502085	6	1981	39	2	1	1	22	0	0	0	0
30	0008C821A1DFBD0641D45F94622502085	6	1981	39	2	1	1	22	0	0	0	0
31	0008C821A1DFBD0641D45F94622502085	6	1981	39	2	1	1	22	0	0	0	0
32	0008C821A1DFBD0641D45F94622502085	6	1981	39	2	1	1	22	0	0	0	0
33	0008C821A1DFBD0641D45F94622502085	6	1981	39	2	1	1	22	0	0	0	0
34	0008C821A1DFBD0641D45F94622502085	6	1981	39	2	1	1	22	0	0	0	0
35	0008C821A1DFBD0641D45F94622502085	6	1981	39	2	1	1	22	0	0	0	0

- Tabela Docente_has_escola

The screenshot shows the MySQL Workbench interface with a query editor at the top containing the SQL command: `SELECT * FROM mydb.docente_has_escola;`. Below the editor is a result grid titled "Result Grid". The grid has two columns: "docente_idDocente" and "escola_CO_ENTIDADE". The data consists of approximately 100 rows, all showing the value 53000013 for the "escola_CO_ENTIDADE" column. The right side of the interface features a vertical toolbar with icons for various database operations like "Result Grid", "Form Editor", "Field Types", "Query Stats", and "Execution Plan".

docente_idDocente	escola_CO_ENTIDADE
15403	53000013
15404	53000013
27658	53000013
27659	53000013
27660	53000013
27661	53000013
27662	53000013
27663	53000013
27664	53000013
41374	53000013
41375	53000013
41376	53000013
60208	53000013
60209	53000013
60210	53000013
67848	53000013
67849	53000013
71384	53000013
71385	53000013
71386	53000013
71387	53000013
71388	53000013
88171	53000013
88172	53000013
88173	53000013
147849	53000013
6104	53000030
8085	53000030
14843	53000030
20198	53000030
29068	53000030
34420	53000030
46146	53000030
48662	53000030
87246	53000030
110987	53000030

- Tabela Docente_has_turma

The screenshot shows the MySQL Workbench interface with a query editor at the top containing the SQL command: `SELECT * FROM mydb.docente_has_turma;`. Below the editor is a result grid titled "Result Grid". The grid has two columns: "docente_idDocente" and "turma_ID_TURMA". The data consists of approximately 100 rows, all showing the value 4439 for the "turma_ID_TURMA" column. The right side of the interface features a vertical toolbar with icons for various database operations like "Result Grid", "Form Editor", "Field Types", "Query Stats", and "Execution Plan".

docente_idDocente	turma_ID_TURMA
130134	643
1542	4439
17008	4439
23538	4439
59740	4439
62812	4439
66332	4439
75210	4439
109878	4439
1543	4440
17009	4440
23539	4440
59741	4440
62813	4440
66333	4440
75211	4440
109879	4440
1544	4441
17010	4441
23540	4441
59742	4441
62814	4441
66334	4441
75212	4441
109880	4441
1545	4442
17011	4442
23541	4442
59743	4442
62815	4442
66335	4442
75213	4442
109881	4442
1546	4443
17012	4443
23542	4443

- Tabela Escola

```
1 • SELECT * FROM mydb.escola;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Fetch rows: | Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

CO_ENTIDADE	NO_ENTIDADE	CO_ORGAO_REGIONAL	TP_SITUACAO_FUNCIONAMENTO	DT_ANO_LETIVO_INICIO	DT_ANO_LETIVO_TERMINO	CO_UF	TP_DEPENDENCIA	TP_LOCALIZACAO
5300013	ASSOC DE MAES PAIS AMIGOS REABILITADOR...	1.0	1	10/02/2020	29/01/2021	53	4	1
5300030	ASSOC DE PAIS E AMIGOS DOS EXCEP - APAE-DF	1.0	1	10/02/2020	16/12/2020	53	4	1
53000102	ASSOC PESTALOZZI DE BRASILIA	1.0	1	10/02/2020	19/01/2021	53	4	1
53000200	CEE 02 DE BRASILIA	1.0	1	10/02/2020	28/01/2021	53	2	1
53000226	ESC CETEB DE JOVENS E ADULTOS	99.0	1	02/01/2020	30/12/2020	53	4	1
53000234	CEJA ASA SUL - CESAS	1.0	1	10/02/2020	28/01/2021	53	2	1
53000331	CED DA AUDICAO E LINGUAGEM LUDOVICO PA...	1.0	1	10/02/2020	16/12/2020	53	4	1
53000439	CEE 01 DE BRASILIA	1.0	1	10/02/2020	28/01/2021	53	2	1
53000714	CEE DE DEFICIENTES VISUAIS	1.0	1	10/02/2020	16/12/2020	53	2	1
53000781	ARVENSE - CED	99.0	1	03/02/2020	23/12/2020	53	4	1
53000790	ESC URSINHO FELIZ	99.0	1	27/01/2020	11/12/2020	53	4	1
53000803	C INF REINO ENCANTADO	99.0	2	0	0	53	4	1
53000811	CASA DA CRIANCA PAO DE SANTO ANTONIO	1.0	1	10/02/2020	29/01/2021	53	4	1
53000820	ESC INF CASA DE ISMAEL	1.0	1	10/02/2020	29/01/2021	53	4	1
53000846	CEF 01 DE BRASILIA	1.0	1	10/02/2020	28/01/2021	53	2	1
53000854	CEF 01 DO PLANALT	1.0	1	10/02/2020	28/01/2021	53	2	1
53000862	CEF 02 DE BRASILIA	1.0	1	10/02/2020	28/01/2021	53	2	1
53000870	CEF 03 DE BRASILIA	1.0	1	10/02/2020	28/01/2021	53	2	1
53000889	CEF 04 DE BRASILIA	1.0	1	10/02/2020	28/01/2021	53	2	1
53000897	CEF 05 DE BRASILIA	1.0	1	10/02/2020	29/01/2021	53	2	1
53000919	CEF GAN	1.0	1	10/02/2020	16/12/2020	53	2	1
53000927	CEF POLIVALENTE	1.0	1	10/02/2020	28/01/2021	53	2	1
53000951	ESC SEB DINATOS	99.0	1	28/01/2020	11/12/2020	53	4	1
53000960	COL MAURICIO SALLES DE MELLO	99.0	1	23/01/2020	15/12/2020	53	4	1
53000986	CEM PAULO FREIRE	1.0	1	10/02/2020	28/01/2021	53	2	1
53001010	CEM ASA NORTE - CEN	1.0	1	10/02/2020	28/01/2021	53	2	1
53001036	CEM ELEFANTE BRANCO	1.0	1	10/02/2020	28/01/2021	53	2	1
53001044	CED GISON	1.0	1	10/02/2020	28/01/2021	53	2	1
53001060	COL LA SALLE BRASILIA	99.0	1	27/01/2020	21/12/2020	53	4	1
53001087	CED LEONARDO DA VINCI	99.0	1	03/02/2020	08/12/2020	53	4	1
53001095	CED LEONARDO DA VINCI - UNID NORTE	99.0	1	03/02/2020	15/12/2020	53	4	1
53001109	CED MARIA AUXILIADORA	99.0	1	27/01/2020	15/12/2020	53	4	1
53001117	CED NOSSA SENHORA DO ROSARIO	99.0	2	0	0	53	4	1
53001125	CED OBJETIVO SP-B	99.0	2	0	0	53	4	1
53001141	CED OBJETIVO SP-B	99.0	2	0	0	53	4	1

- Tabela Turma

```
1 • SELECT * FROM mydb.turma;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Fetch rows: | Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

ID_TURMA	NO_TURMA	TP_MEDIACAO_DIDATICO_PEDAGO	TX_HR_INICIAL	TX_ML_INICIAL	NU_DIAS_ATIVIDADE	QT_MATRICULAS
643	PRE II VESPERTINO	1	13.0	0.0	5	3
4439	8º FM	1	7.0	15.0	5	30
4440	8º EM	1	7.0	15.0	5	33
4441	8º DM	1	7.0	15.0	5	33
4442	8º CM	1	7.0	15.0	5	33
4443	8º BM	1	7.0	15.0	5	33
4444	8º AM	1	7.0	15.0	5	33
4452	6º EV	1	13.0	15.0	5	33
4453	6º DV	1	13.0	15.0	5	34
4454	6º CV	1	13.0	15.0	5	34
4455	6º BV	1	13.0	15.0	5	30
4456	6º AV	1	13.0	15.0	5	28
4464	ATENDIMENTO AN...	1	14.0	0.0	4	9
4465	ATENDIMENTO AN...	1	8.0	0.0	4	12
6946	19-A	1	7.0	30.0	5	28
7300	PE1A	1	8.0	0.0	5	21
7701	PE1B	1	8.0	0.0	5	21
7702	PE1C	1	8.0	0.0	5	17
7704	PE1E	1	13.0	40.0	5	21
7705	PE1F	1	13.0	40.0	5	21
7706	PE2A	1	8.0	0.0	5	25
7707	PE2B	1	8.0	0.0	5	24
7708	PE2C	1	8.0	0.0	5	23
7711	PE2F	1	13.0	40.0	5	24
7712	PE2G	1	13.0	40.0	5	23
7714	1º ANO C	1	8.0	0.0	5	26
7715	1º ANO D	1	8.0	0.0	5	20
7718	1º ANO G	1	8.0	0.0	5	22
7719	2º ANO A	1	8.0	0.0	5	27
7720	2º ANO B	1	8.0	0.0	5	28
7721	2º ANO C	1	8.0	0.0	5	29
7722	2º ANO D	1	8.0	0.0	5	29
7723	2 ANO E	1	8.0	0.0	5	29
7724	2º ANO F	1	13.0	40.0	5	29
7725	2º ANO G	1	13.0	40.0	5	29
7726	3º ANO A	1	8.0	0.0	5	25

Consultas SQL

1) Quantidade de Professores do Distrito Federal com algum tipo de necessidade especial (deficiência física, autismo, etc):

In [6]:

try:

Conexão com o banco de dados MySQL criado Localmente para o trabalho

```

# O banco tem nome 'mydb' e tanto login quanto senha são 'root'
conn = mysql.connect(host='localhost', database='mydb', user='root', password='root')

# Procede caso a conexão com o banco tenha sido bem sucedida
if conn.is_connected():
    # Cursor é a instância criada para a manipulação do banco
    cursor = conn.cursor()
    # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente
    cursor.execute("select database();")
    record = cursor.fetchone()

    # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada
    cursor.execute("""SELECT * FROM mydb.docente WHERE IN_NECESSIDADE_ESPECIAL = 1""")
    myresult = cursor.fetchall()

    print("Quantidade de professores com algum tipo de necessidade especial (defici
except Error as e:
    print("Error while connecting to MySQL", e)

```

Quantidade de professores com algum tipo de necessidade especial (deficiência física, autismo, etc) no DF: 1160

2) Quantidade de Professores do Distrito Federal que tenham mestrado ou doutorado:

In [7]:

```

try:
    # Conexão com o banco de dados MySQL criado localmente para o trabalho
    # O banco tem nome 'mydb' e tanto login quanto senha são 'root'
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root')

    # Procede caso a conexão com o banco tenha sido bem sucedida
    if conn.is_connected():
        # Cursor é a instância criada para a manipulação do banco
        cursor = conn.cursor()
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente
        cursor.execute("select database();")
        record = cursor.fetchone()

        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada
        cursor.execute("""SELECT * FROM mydb.docente WHERE IN_MESTRADO = 1 or IN_DOUTOR
        myresult = cursor.fetchall()

        print("Quantidade de professores com mestrado ou doutorado no DF: " + str(len(m
except Error as e:
    print("Error while connecting to MySQL", e)

```

Quantidade de professores com mestrado ou doutorado no DF: 9244

3) Quantidade de Escolas do Distrito Federal que não tenham água potável ou não tenham energia elétrica:

In [8]:

```

try:
    # Conexão com o banco de dados MySQL criado localmente para o trabalho
    # O banco tem nome 'mydb' e tanto login quanto senha são 'root'
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root')

    # Procede caso a conexão com o banco tenha sido bem sucedida
    if conn.is_connected():
        # Cursor é a instância criada para a manipulação do banco

```

```

cursor = conn.cursor()
# Aqui damos um SELECT no banco e nos conectamos à ele efetivamente
cursor.execute("select database();")
record = cursor.fetchone()

# Uma vez dentro do banco, executamos o comando SELECT com a condição desejada
cursor.execute("""SELECT * FROM mydb.escola WHERE IN_AGUA_POTAVEL = 0 or IN_ENE
myresult = cursor.fetchall()

print("Quantidade de escolas que não tenham água potável ou não tenham energia
except Error as e:
    print("Error while connecting to MySQL", e)

```

Quantidade de escolas que não tenham água potável ou não tenham energia elétrica no DF:
119

4) Quantidade de Escolas do Distrito Federal que tenham Professor Psicólogo, Nutricionista ou Fonoaudiólogo:

In [9]:

```

try:
    # Conexão com o banco de dados MySQL criado Localmente para o trabalho
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'
    conn = msq.connect(host='localhost', database='mydb', user='root', password='root'

    # Procede caso a conexão com o banco tenha sido bem sucedida
    if conn.is_connected():
        # Cursor é a instância criada para a manipulação do banco
        cursor = conn.cursor()
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente
        cursor.execute("select database();")
        record = cursor.fetchone()

        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada
        cursor.execute("""SELECT * FROM mydb.escola WHERE QT_PROF_FONAUDIOLOGO = 0 or Q
myresult = cursor.fetchall()

        print("Quantidade de escolas que tenham psicólogo, nutricionista ou fonoaudiólogo
except Error as e:
    print("Error while connecting to MySQL", e)

```

Quantidade de escolas que tenham psicólogo, nutricionista ou fonoaudiólogo no DF: 1342

5) Quantidade de Escolas do Distrito Federal que tenham computadores, notebooks ou tablets a disposição dos estudantes:

In [10]:

```

try:
    # Conexão com o banco de dados MySQL criado Localmente para o trabalho
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'
    conn = msq.connect(host='localhost', database='mydb', user='root', password='root'

    # Procede caso a conexão com o banco tenha sido bem sucedida
    if conn.is_connected():
        # Cursor é a instância criada para a manipulação do banco
        cursor = conn.cursor()
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente
        cursor.execute("select database();")
        record = cursor.fetchone()

        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada

```

```

cursor.execute("""SELECT * FROM mydb.escola WHERE IN_DESKTOP_ALUNO = 1 or IN_CO
myresult = cursor.fetchall()

print("Quantidade de escolas que tenham computadores, notebooks ou tablets a di
except Error as e:
    print("Error while connecting to MySQL", e)

```

Quantidade de escolas que tenham computadores, notebooks ou tablets a disposição dos estudantes no DF: 889

6) Quantidade de docentes brancos, negros, pardos e indígenas no Distrito Federal:

In [11]:

```

try:
    # Conexão com o banco de dados MySQL criado localmente para o trabalho
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root'

    # Procede caso a conexão com o banco tenha sido bem sucedida
    if conn.is_connected():
        # Cursor é a instância criada para a manipulação do banco
        cursor = conn.cursor()
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente
        cursor.execute("select database();")
        record = cursor.fetchone()

        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada
        cursor.execute("""SELECT COUNT(ID_DOCENTE) FROM mydb.docente WHERE TP_COR_RACA
myresult = cursor.fetchone()
prof_branco = myresult[0]

        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada
        cursor.execute("""SELECT COUNT(ID_DOCENTE) FROM mydb.docente WHERE TP_COR_RACA
myresult = cursor.fetchone()
prof_negro = myresult[0]

        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada
        cursor.execute("""SELECT COUNT(ID_DOCENTE) FROM mydb.docente WHERE TP_COR_RACA
myresult = cursor.fetchone()
prof_pardo = myresult[0]

        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada
        cursor.execute("""SELECT COUNT(ID_DOCENTE) FROM mydb.docente WHERE TP_COR_RACA
myresult = cursor.fetchone()
prof_indigena = myresult[0]

    print("Quantidade de professores brancos no DF: " + str(prof_branco))
    print("Quantidade de professores negros no DF: " + str(prof_negro))
    print("Quantidade de professores pardos no DF: " + str(prof_pardo))
    print("Quantidade de professores indigenas no DF: " + str(prof_indigena))
except Error as e:
    print("Error while connecting to MySQL", e)

```

Quantidade de professores brancos no DF: 44611
 Quantidade de professores negros no DF: 7485
 Quantidade de professores pardos no DF: 44868
 Quantidade de professores indigenas no DF: 1079

7) Quantidade de alunos no Distrito Federal com mais de 18 anos, bem como de menores:

In [12]:

```
try:  
    # Conexão com o banco de dados MySQL criado Localmente para o trabalho  
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'  
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root'  
  
    # Procede caso a conexão com o banco tenha sido bem sucedida  
    if conn.is_connected():  
        # Cursor é a instância criada para a manipulação do banco  
        cursor = conn.cursor()  
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente  
        cursor.execute("select database();")  
        record = cursor.fetchone()  
  
        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada  
        cursor.execute("""SELECT COUNT(ID_ALUNO) FROM mydb.aluno WHERE NU_IDADE > 18""")  
        myresult = cursor.fetchone()  
        qtd_alunos_maior = myresult[0]  
  
        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada  
        cursor.execute("""SELECT COUNT(ID_ALUNO) FROM mydb.aluno WHERE NU_IDADE < 18""")  
        myresult = cursor.fetchone()  
        qtd_alunos_menor = myresult[0]  
  
        print("Quantidade de alunos com mais de 18 anos no DF: " + str(qtd_alunos_maior)  
        print("Quantidade de alunos com menos de 18 anos no DF: " + str(qtd_alunos_menor)  
except Error as e:  
    print("Error while connecting to MySQL", e)
```

Quantidade de alunos com mais de 18 anos no DF: 64818
Quantidade de alunos com menos de 18 anos no DF: 626825

8) Quantidade de Escolas vinculadas à Segurança Pública e quantidade de escolas vinculadas à Secretaria de Educação no Distrito Federal:

In [13]:

```
try:  
    # Conexão com o banco de dados MySQL criado Localmente para o trabalho  
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'  
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root'  
  
    # Procede caso a conexão com o banco tenha sido bem sucedida  
    if conn.is_connected():  
        # Cursor é a instância criada para a manipulação do banco  
        cursor = conn.cursor()  
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente  
        cursor.execute("select database();")  
        record = cursor.fetchone()  
  
        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada  
        cursor.execute("""SELECT COUNT(CO_ENTIDADE) FROM mydb.escola WHERE IN_VINCULO_S  
        myresult = cursor.fetchone()  
        secretaria = myresult[0]  
  
        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada  
        cursor.execute("""SELECT COUNT(CO_ENTIDADE) FROM mydb.escola WHERE IN_VINCULO_S  
        myresult = cursor.fetchone()  
        segurança = myresult[0]  
  
        print("Quantidade de escolas vinculadas à Secretaria de Estado de Educação do D  
        print("Quantidade de escolas vinculadas à Secretaria de Estado de Segurança Púb
```

```
except Error as e:  
    print("Error while connecting to MySQL", e)
```

Quantidade de escolas vinculadas à Secretaria de Estado de Educação do Distrito Federal: 694

Quantidade de escolas vinculadas à Secretaria de Estado de Segurança Pública do Distrito Federal (conhecidas enquanto "escolas militarizadas"): 14

9) Quantidade de Escolas do Distrito Federal que tenham internet para os estudantes:

In [14]:

```
try:  
    # Conexão com o banco de dados MySQL criado Localmente para o trabalho  
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'  
    conn = msql.connect(host='localhost', database='mydb', user='root', password='root'  
    if conn.is_connected():  
        # Cursor é a instância criada para a manipulação do banco  
        cursor = conn.cursor()  
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente  
        cursor.execute("select database();")  
        record = cursor.fetchone()  
  
        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada  
        cursor.execute("""SELECT * FROM mydb.escola WHERE IN_INTERNET_ALUNOS = 1""")  
        myresult = cursor.fetchall()  
  
        print("Quantidade de escolas que com internet disponível para os estudantes no DF: ", len(myresult))  
    except Error as e:  
        print("Error while connecting to MySQL", e)
```

Quantidade de escolas que com internet disponível para os estudantes no DF: 543

10) Quantidade de Escolas do Distrito Federal que tenham investido em energias renováveis:

In [15]:

```
try:  
    # Conexão com o banco de dados MySQL criado Localmente para o trabalho.  
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'  
    conn = msql.connect(host='localhost', database='mydb', user='root', password='root'  
  
    # Procede caso a conexão com o banco tenha sido bem sucedida  
    if conn.is_connected():  
        # Cursor é a instância criada para a manipulação do banco  
        cursor = conn.cursor()  
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente  
        cursor.execute("select database();")  
        record = cursor.fetchone()  
  
        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada  
        cursor.execute("""SELECT * FROM mydb.escola WHERE IN_ENERGIA_RENOVAVEL = 1""")  
        myresult = cursor.fetchall()  
  
        print("Quantidade de escolas que façam uso de algum tipo de energia renovável no DF: ", len(myresult))  
    except Error as e:  
        print("Error while connecting to MySQL", e)
```

Quantidade de escolas que façam uso de algum tipo de energia renovável no DF: 7

11) Lista dos alunos do Distrito Federal com mais de 18 anos, ordenado pelo mais velho:

In [20]:

```
try:  
    # Conexão com o banco de dados MySQL criado localmente para o trabalho  
    # O banco tem nome 'mydb' e tanto login quanto senha são 'root'  
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root')  
  
    # Procede caso a conexão com o banco tenha sido bem sucedida  
    if conn.is_connected():  
        # Cursor é a instância criada para a manipulação do banco  
        cursor = conn.cursor()  
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente  
        cursor.execute("select database();")  
        record = cursor.fetchone()  
  
        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada  
        cursor.execute("""SELECT ID_ALUNO, NU_IDADE, TP_SEXO, TP_COR_RACA FROM mydb.alu  
                      myresult = cursor.fetchall()  
  
        print("Lista com os 5 alunos mais velhos da rede pública do Distrito Federal: ")  
  
        myresult_sliced = myresult[:5]  
  
        print("(ID_ALUNO, NU_IDADE, TP_SEXO, TP_COR_RACA)")  
  
        for x in myresult_sliced:  
            print(x)  
except Error as e:  
    print("Error while connecting to MySQL", e)
```

Lista com os 5 alunos mais velhos da rede pública do Distrito Federal:
(ID_ALUNO, NU_IDADE, TP_SEXO, TP_COR_RACA)
('5B062CBFF66FE95F91935BA858F6EBB1', 86, 2, 0)
('54FBDE565028C012F5D674D9D61A3DB8', 85, 2, 0)
('C05B49A3F05A32BC72F443E9B87FF353', 85, 2, 1)
('C05B49A3F05A32BC72F443E9B87FF353', 85, 2, 1)
('872F7C1B91F91B613966D67A79F9A7E0', 85, 2, 0)

12) Lista dos alunos do Distrito Federal com mais de 18 anos, ordenado pelo mais jovem:

In [21]:

```
try:  
    # Conexão com o banco de dados MySQL criado localmente para o trabalho  
    # O banco tem nome 'mydb' e tanto login quanto senha são 'root'  
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root')  
  
    # Procede caso a conexão com o banco tenha sido bem sucedida  
    if conn.is_connected():  
        # Cursor é a instância criada para a manipulação do banco  
        cursor = conn.cursor()  
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente  
        cursor.execute("select database();")  
        record = cursor.fetchone()  
  
        cursor.execute("""SELECT ID_ALUNO, NU_IDADE, TP_SEXO, TP_COR_RACA FROM mydb.alu  
                      myresult = cursor.fetchall()  
  
        print("Lista com os 5 alunos mais jovens da rede pública do Distrito Federal: ")
```

```

myresult_sliced = myresult[:5]

print("(ID_ALUNO, NU_IDADE, TP_SEXO, TP_COR_RACA)")

for x in myresult_sliced:
    print(x)
except Error as e:
    print("Error while connecting to MySQL", e)

```

Lista com os 5 alunos mais jovens da rede pública do Distrito Federal:

ID_ALUNO	NU_IDADE	TP_SEXO	TP_COR_RACA
'CC73694467F35EE268125095D810C0CC'	0	2	0
'AD6531FB642E38D1EA52774B34E10920'	0	2	0
'0202BA795A5D33DD212B6E1DC17D31BF'	0	2	0
'7115FF78B7289746FC9F022CCD3A1F78'	0	1	3
'70569456CD7C41D2FA891CC9AAF5C0C9'	1	2	1

13) Lista dos Professores do Distrito Federal com mais de 40 anos, ordenado pelo mais velho:

In [22]:

```

try:
    # Conexão com o banco de dados MySQL criado Localmente para o trabalho
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'
    conn = msql.connect(host='localhost', database='mydb', user='root', password='root')

    # Procede caso a conexão com o banco tenha sido bem sucedida
    if conn.is_connected():
        # Cursor é a instância criada para a manipulação do banco
        cursor = conn.cursor()
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente
        cursor.execute("select database();")
        record = cursor.fetchone()

        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada
        cursor.execute("""SELECT ID_DOCENTE, NU_IDADE, TP_SEXO, TP_COR_RACA FROM mydb.d

myresult = cursor.fetchall()

print("Quantidade de professores com MAIS de 40 anos no DF: " + str(len(myresul
print("Lista com os 5 professores mais velhos da rede pública do Distrito Feder

myresult_sliced = myresult[:5]

print("(ID_DOCENTE, NU_IDADE, TP_SEXO, TP_COR_RACA)")

for x in myresult_sliced:
    print(x)
except Error as e:
    print("Error while connecting to MySQL", e)

```

Quantidade de professores com mais de 40 anos no DF: 72688
 Lista com os 5 professores mais velhos da rede pública do Distrito Federal:

ID_DOCENTE	NU_IDADE	TP_SEXO	TP_COR_RACA
'620A55967E054357CDF3DC5C570D8F18'	78	2	3
'620A55967E054357CDF3DC5C570D8F18'	78	2	3
'620A55967E054357CDF3DC5C570D8F18'	78	2	3
'620A55967E054357CDF3DC5C570D8F18'	78	2	3
'3D8859B35F4A780E675AA0D46064122E'	78	1	3

14) Lista dos Professores do Distrito Federal com menos de 40 anos, ordenado pelo mais jovem:

In [23]:

```
try:  
    # Conexão com o banco de dados MySQL criado Localmente para o trabalho  
    # O banco tem nome 'mydb' e tanto Login quanto senha são 'root'  
    conn = mysql.connect(host='localhost', database='mydb', user='root', password='root')  
  
    # Procede caso a conexão com o banco tenha sido bem sucedida  
    if conn.is_connected():  
        # Cursor é a instância criada para a manipulação do banco  
        cursor = conn.cursor()  
        # Aqui damos um SELECT no banco e nos conectamos à ele efetivamente  
        cursor.execute("select database();")  
        record = cursor.fetchone()  
  
        # Uma vez dentro do banco, executamos o comando SELECT com a condição desejada  
        cursor.execute("""SELECT ID_DOCENTE, NU_IDADE, TP_SEXO, TP_COR_RACA FROM mydb.d  
myresult = cursor.fetchall()  
  
print("Quantidade de professores com MENOS de 40 anos no DF: " + str(len(myresu  
print("Lista com os 5 professores mais jovens da rede pública do Distrito Feder  
  
myresult_sliced = myresult[:5]  
  
print("(ID_DOCENTE, NU_IDADE, TP_SEXO, TP_COR_RACA)")  
  
for x in myresult_sliced:  
    print(x)  
except Error as e:  
    print("Error while connecting to MySQL", e)
```

Quantidade de professores com MENOS de 40 anos no DF: 73454
Lista com os 5 professores mais jovens da rede pública do Distrito Federal:
(ID_DOCENTE, NU_IDADE, TP_SEXO, TP_COR_RACA)
('3BB0076FF8453AFB9B2E77544DBDA47B', 17, 2, 3)
('11C448A363D53A8866D72B7154D0619E', 17, 2, 3)
('43FF000167415B3BF977821BB296E5F4', 18, 2, 3)
('F4246B3EC63ED2323B7CA097D600D744', 18, 2, 3)
('F4246B3EC63ED2323B7CA097D600D744', 18, 2, 3)