Trabalho_Individual

November 16, 2021

1 Trabalho individual da disciplina FBD 2021.1

1.0.1 Aluno: Eduardo de Oliveira Castro

1.0.2 Matrícula: 210008164

1.0.3 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.

1.0.4 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.

1.0.5 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:

O INEP também oferece, juntamente com os dados, um excelente dicionários 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:

1.0.6 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:

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

[2]:
```
# Carregamento do maior csv, o de alunos/matrículas. Isso é feito agora para␣
 ↪que não haja estouro de memória posteriormente
alunos = pd.read_csv(r'C:
 ↪\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.
 ↪1_FBD\Trabalho_Individual\Dados\matricula_co.csv', sep='|',␣
 ↪encoding='ISO-8859-1')
```

[3]:
```
# Carregando do arquivo referente as escolas
escolas = pd.read_csv(r'C:
 ↪\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.
 ↪1_FBD\Trabalho_Individual\Dados\escolas.csv', sep='|', encoding='ISO-8859-1')

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

```
escolas = escolas.drop(columns=['NU_ANO_CENSO', 'CO_REGIAO', 'CO_MESORREGIAO',␣
 ↪'CO_MICRORREGIAO', 'CO_MUNICIPIO', 'CO_DISTRITO',␣
 ↪'TP_LOCALIZACAO_DIFERENCIADA', 'IN_VINCULO_SECRETARIA_SAUDE',␣
 ↪'IN_VINCULO_OUTRO_ORGAO', 'TP_CATEGORIA_ESCOLA_PRIVADA', 'IN_CONVENIADA_PP',␣
 ↪'TP_CONVENIO_PODER_PUBLICO', 'IN_MANT_ESCOLA_PRIVADA_EMP',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_ONG', 'IN_MANT_ESCOLA_PRIVADA_OSCIP',␣
 ↪'IN_MANT_ESCOLA_PRIV_ONG_OSCIP', 'IN_MANT_ESCOLA_PRIVADA_SIND',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_SIST_S', 'IN_MANT_ESCOLA_PRIVADA_S_FINS',␣
 ↪'TP_REGULAMENTACAO', 'TP_RESPONSAVEL_REGULAMENTACAO',␣
 ↪'CO_ESCOLA_SEDE_VINCULADA', 'CO_IES_OFERTANTE', 'IN_LOCAL_FUNC_GALPAO',␣
 ↪'TP_OCUPACAO_GALPAO', 'IN_LOCAL_FUNC_SALAS_OUTRA_ESC',␣
 ↪'IN_LOCAL_FUNC_OUTROS', 'IN_PREDIO_COMPARTILHADO', 'IN_AGUA_REDE_PUBLICA',␣
 ↪'IN_AGUA_POCO_ARTESIANO', 'IN_AGUA_CACIMBA', 'IN_AGUA_FONTE_RIO',␣
 ↪'IN_AGUA_INEXISTENTE', 'IN_ENERGIA_GERADOR_FOSSIL',␣
 ↪'IN_ENERGIA_INEXISTENTE', 'IN_ESGOTO_FOSSA_SEPTICA',␣
 ↪'IN_ESGOTO_FOSSA_COMUM', 'IN_ESGOTO_FOSSA', 'IN_ESGOTO_INEXISTENTE',␣
 ↪'IN_LIXO_QUEIMA', 'IN_LIXO_ENTERRA', 'IN_LIXO_DESTINO_FINAL_PUBLICO',␣
 ↪'IN_LIXO_DESCARTA_OUTRA_AREA', 'IN_TRATAMENTO_LIXO_SEPARACAO',␣
 ↪'IN_TRATAMENTO_LIXO_REUTILIZA', 'IN_TRATAMENTO_LIXO_RECICLAGEM',␣
 ↪'IN_TRATAMENTO_LIXO_INEXISTENTE', 'IN_ALMOXARIFADO', 'IN_BANHEIRO_EI',␣
 ↪'IN_BANHEIRO_FUNCIONARIOS', 'IN_BANHEIRO_CHUVEIRO',␣
 ↪'IN_BIBLIOTECA_SALA_LEITURA', 'IN_DESPENSA', 'IN_DORMITORIO_ALUNO',␣
 ↪'IN_DORMITORIO_PROFESSOR', 'IN_SALA_MUSICA_CORAL', 'IN_SALA_ESTUDIO_DANCA',␣
 ↪'IN_SALA_MULTIUSO', 'IN_SALA_DIRETORIA', 'IN_SALA_LEITURA',␣
 ↪'IN_SALA_PROFESSOR', 'IN_SALA_REPOUSO_ALUNO', 'IN_SECRETARIA',␣
 ↪'IN_SALA_ATENDIMENTO_ESPECIAL', 'IN_TERREIRAO', 'IN_VIVEIRO',␣
 ↪'IN_DEPENDENCIAS_OUTRAS', 'IN_ACESSIBILIDADE_CORRIMAO',␣
 ↪'IN_ACESSIBILIDADE_ELEVADOR', 'IN_ACESSIBILIDADE_PISOS_TATEIS',␣
```

```
# Fateia apenas parte as escolas cuja UF seja equivalente ao número  53, que␣
↪representa do DF
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 será muito utilizado para lidar com as demais tabelas
escolas_df_codigos = escolas_df['CO_ENTIDADE'].values.tolist()

# Salva uma nova versão do CSV, contendo apenas as colunas consideradas␣
↪relevantes e os dados do DF
escolas_df.to_csv(r'C:
↪\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.
↪1_FBD\Trabalho_Individual\Dados\escolas_DF.csv', sep=';',␣
↪encoding='ISO-8859-1', index = False)
```

```
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,
```

[4]:
```
# Carregando do arquivo referente aos docentes
docentes = pd.read_csv(r'C:
↪\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.
↪1_FBD\Trabalho_Individual\Dados\docentes_co.csv', sep='|',␣
↪encoding='ISO-8859-1')

# 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', 'CO_MUNICIPIO_NASC', 'CO_UF_END', 'CO_MUNICIPIO_END',␣
 ↪'TP_ZONA_RESIDENCIAL', 'TP_LOCAL_RESID_DIFERENCIADA', 'IN_BAIXA_VISAO',␣
 ↪'IN_CEGUEIRA', 'IN_DEF_AUDITIVA', 'IN_DEF_INTELECTUAL', 'IN_SURDEZ',␣
 ↪'IN_SURDOCEGUEIRA', 'IN_DEF_MULTIPLA', 'IN_AUTISMO', 'TP_ESCOLARIDADE',␣
 ↪'TP_ENSINO_MEDIO', 'TP_SITUACAO_CURSO_1', 'CO_AREA_CURSO_1', 'CO_CURSO_1',␣
 ↪'IN_LICENCIATURA_1', 'NU_ANO_CONCLUSAO_1', 'TP_TIPO_IES_1', 'CO_IES_1',␣
 ↪'TP_SITUACAO_CURSO_2', 'CO_AREA_CURSO_2', 'CO_CURSO_2', 'IN_LICENCIATURA_2',␣
 ↪'NU_ANO_CONCLUSAO_2', 'TP_TIPO_IES_2', 'CO_IES_2', 'TP_SITUACAO_CURSO_3',␣
 ↪'CO_AREA_CURSO_3', 'CO_CURSO_3', 'IN_LICENCIATURA_3', 'NU_ANO_CONCLUSAO_3',␣
 ↪'TP_TIPO_IES_3', 'CO_IES_3', 'IN_COMPLEMENTACAO_PEDAGOGICA',␣
 ↪'CO_AREA_COMPL_PEDAGOGICA_1', 'CO_AREA_COMPL_PEDAGOGICA_2',␣
 ↪'CO_AREA_COMPL_PEDAGOGICA_3', 'IN_ESPECIFICO_CRECHE',␣
 ↪'IN_ESPECIFICO_PRE_ESCOLA', 'IN_ESPECIFICO_ANOS_INICIAIS',␣
 ↪'IN_ESPECIFICO_ANOS_FINAIS', 'IN_ESPECIFICO_ENS_MEDIO', 'IN_ESPECIFICO_EJA',␣
 ↪'IN_ESPECIFICO_ED_ESPECIAL', 'IN_ESPECIFICO_ED_INDIGENA',␣
 ↪'IN_ESPECIFICO_CAMPO', 'IN_ESPECIFICO_AMBIENTAL',␣
 ↪'IN_ESPECIFICO_DIR_HUMANOS', 'IN_ESPECIFICO_DIV_SEXUAL',␣
 ↪'IN_ESPECIFICO_DIR_ADOLESC', 'IN_ESPECIFICO_AFRO', 'IN_ESPECIFICO_GESTAO',␣
 ↪'IN_ESPECIFICO_OUTROS', 'IN_ESPECIFICO_NENHUM', 'IN_DISC_LINGUA_PORTUGUESA',␣
 ↪'IN_DISC_EDUCACAO_FISICA', 'IN_DISC_ARTES', 'IN_DISC_LINGUA_INGLES',␣
 ↪'IN_DISC_LINGUA_ESPANHOL', 'IN_DISC_LINGUA_FRANCES', 'IN_DISC_LINGUA_OUTRA',␣
 ↪'IN_DISC_LIBRAS', 'IN_DISC_LINGUA_INDIGENA', 'IN_DISC_PORT_SEGUNDA_LINGUA',␣
 ↪'IN_DISC_MATEMATICA', 'IN_DISC_CIENCIAS', 'IN_DISC_FISICA',␣
 ↪'IN_DISC_QUIMICA', 'IN_DISC_BIOLOGIA', 'IN_DISC_HISTORIA',␣
 ↪'IN_DISC_GEOGRAFIA', 'IN_DISC_SOCIOLOGIA', 'IN_DISC_FILOSOFIA',␣
 ↪'IN_DISC_ESTUDOS_SOCIAIS', 'IN_DISC_EST_SOCIAIS_SOCIOLOGIA',␣
 ↪'IN_DISC_INFORMATICA_COMPUTACAO', 'IN_DISC_ENSINO_RELIGIOSO',␣
 ↪'IN_DISC_PROFISSIONALIZANTE', 'IN_DISC_ESTAGIO_SUPERVISIONADO',␣
 ↪'IN_DISC_PEDAGOGICAS', 'IN_DISC_OUTRAS', 'TP_TIPO_ATENDIMENTO_TURMA',␣
 ↪'TP_TIPO_LOCAL_TURMA', 'TP_MEDIACAO_DIDATICO_PEDAGO', 'TP_ETAPA_ENSINO',␣
 ↪'CO_CURSO_EDUC_PROFISSIONAL', 'IN_ESPECIAL_EXCLUSIVA', 'IN_REGULAR',␣
 ↪'IN_EJA', 'IN_PROFISSIONALIZANTE', 'CO_REGIAO', 'CO_MESORREGIAO',␣
 ↪'CO_MICRORREGIAO', 'CO_MUNICIPIO', 'CO_DISTRITO', 'TP_DEPENDENCIA',␣
 ↪'TP_LOCALIZACAO', 'TP_CATEGORIA_ESCOLA_PRIVADA', 'IN_CONVENIADA_PP',␣
 ↪'TP_CONVENIO_PODER_PUBLICO', 'IN_MANT_ESCOLA_PRIVADA_EMP',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_ONG', 'IN_MANT_ESCOLA_PRIVADA_OSCIP',␣
 ↪'IN_MANT_ESCOLA_PRIV_ONG_OSCIP', 'IN_MANT_ESCOLA_PRIVADA_SIND',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_SIST_S', 'IN_MANT_ESCOLA_PRIVADA_S_FINS',␣
 ↪'TP_REGULAMENTACAO', 'TP_LOCALIZACAO_DIFERENCIADA', 'IN_EDUCACAO_INDIGENA'])

# Cria um novo dataframe apenas contendo os docentes cuja escola seja do DF,␣
 ↪para tal comparação utilizamos a lista com cs códigos das escolas criada␣
 ↪anteriormente
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 dados do DF
docentes_df.to_csv(r'C:
 ↪\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.
 ↪1_FBD\Trabalho_Individual\Dados\docentes_DF.csv', sep=';',␣
 ↪encoding='ISO-8859-1', index = False)
```

```
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,
```

[5]:
```
# Carregando do arquivo referente aos gestores
gestor = pd.read_csv(r'C:
 ↪\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.
 ↪1_FBD\Trabalho_Individual\Dados\gestor.csv', sep='|', encoding='ISO-8859-1')

# 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_UF_NASC', 'CO_MUNICIPIO_NASC', 'IN_BAIXA_VISAO',␣
 ↪'IN_CEGUEIRA', 'IN_DEF_AUDITIVA', 'IN_DEF_FISICA', 'IN_DEF_INTELECTUAL',␣
 ↪'IN_SURDEZ', 'IN_SURDOCEGUEIRA', 'IN_DEF_MULTIPLA', 'IN_AUTISMO',␣
 ↪'TP_ESCOLARIDADE', 'TP_ENSINO_MEDIO', 'CO_AREA_CURSO_1', 'CO_CURSO_1',␣
 ↪'IN_LICENCIATURA_1', 'NU_ANO_CONCLUSAO_1', 'TP_TIPO_IES_1', 'CO_IES_1',␣
 ↪'CO_AREA_CURSO_2', 'CO_CURSO_2', 'IN_LICENCIATURA_2', 'NU_ANO_CONCLUSAO_2',␣
 ↪'TP_TIPO_IES_2', 'CO_IES_2', 'IN_ESPECIFICO_CRECHE',␣
 ↪'IN_ESPECIFICO_PRE_ESCOLA', 'IN_ESPECIFICO_ANOS_INICIAIS',␣
 ↪'IN_ESPECIFICO_ANOS_FINAIS', 'IN_ESPECIFICO_ENS_MEDIO', 'IN_ESPECIFICO_EJA',␣
 ↪'IN_ESPECIFICO_ED_ESPECIAL', 'IN_ESPECIFICO_ED_INDIGENA',␣
 ↪'IN_ESPECIFICO_CAMPO', 'IN_ESPECIFICO_AMBIENTAL',␣
 ↪'IN_ESPECIFICO_DIR_HUMANOS', 'IN_ESPECIFICO_DIV_SEXUAL',␣
 ↪'IN_ESPECIFICO_DIR_ADOLESC', 'IN_ESPECIFICO_AFRO', 'IN_ESPECIFICO_GESTAO',␣
 ↪'IN_ESPECIFICO_OUTROS', 'IN_ESPECIFICO_NENHUM', 'TP_CARGO_GESTOR',␣
 ↪'TP_TIPO_ACESSO_CARGO', 'CO_REGIAO', 'CO_MESORREGIAO', 'CO_MICRORREGIAO',␣
 ↪'CO_MUNICIPIO', 'CO_DISTRITO', 'TP_DEPENDENCIA', 'TP_LOCALIZACAO',␣
 ↪'TP_CATEGORIA_ESCOLA_PRIVADA', 'IN_CONVENIADA_PP',␣
 ↪'TP_CONVENIO_PODER_PUBLICO', 'IN_MANT_ESCOLA_PRIVADA_EMP',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_ONG', 'IN_MANT_ESCOLA_PRIV_ONG_OSCIP',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_OSCIP', 'IN_MANT_ESCOLA_PRIVADA_SIND',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_SIST_S', 'IN_MANT_ESCOLA_PRIVADA_S_FINS',␣
 ↪'TP_REGULAMENTACAO', 'TP_LOCALIZACAO_DIFERENCIADA', 'IN_EDUCACAO_INDIGENA'])
```

```
# Cria um novo dataframe apenas contendo os gestores cuja escola seja do DF,␣
 ↪para tal comparação utilizamos a lista com cs códigos das escolas criada␣
 ↪anteriormente
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 dados do DF
gestor_df.to_csv(r'C:
 ↪\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.
 ↪1_FBD\Trabalho_Individual\Dados\gestor_DF.csv', sep=';',␣
 ↪encoding='ISO-8859-1', index = False)
```

[6]:
```
# Carregando do arquivo referente as turmas
turma = pd.read_csv(r'C:
 ↪\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.
 ↪1_FBD\Trabalho_Individual\Dados\turmas.csv', sep='|', encoding='ISO-8859-1')

# 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', 'TP_TIPO_LOCAL_TURMA', 'CO_TIPO_ATIVIDADE_1',␣
 ↪'CO_TIPO_ATIVIDADE_2', 'CO_TIPO_ATIVIDADE_3', 'CO_TIPO_ATIVIDADE_4',␣
 ↪'CO_TIPO_ATIVIDADE_5', 'CO_TIPO_ATIVIDADE_6', 'TP_ETAPA_ENSINO',␣
 ↪'CO_CURSO_EDUC_PROFISSIONAL', 'IN_ESPECIAL_EXCLUSIVA', 'IN_REGULAR',␣
 ↪'IN_EJA', 'IN_PROFISSIONALIZANTE', 'IN_DISC_LINGUA_PORTUGUESA',␣
 ↪'IN_DISC_EDUCACAO_FISICA', 'IN_DISC_ARTES', 'IN_DISC_LINGUA_INGLES',␣
 ↪'IN_DISC_LINGUA_ESPANHOL', 'IN_DISC_LINGUA_FRANCES', 'IN_DISC_LINGUA_OUTRA',␣
 ↪'IN_DISC_LIBRAS', 'IN_DISC_LINGUA_INDIGENA', 'IN_DISC_PORT_SEGUNDA_LINGUA',␣
 ↪'IN_DISC_MATEMATICA', 'IN_DISC_CIENCIAS', 'IN_DISC_FISICA',␣
 ↪'IN_DISC_QUIMICA', 'IN_DISC_BIOLOGIA', 'IN_DISC_HISTORIA',␣
 ↪'IN_DISC_GEOGRAFIA', 'IN_DISC_SOCIOLOGIA', 'IN_DISC_FILOSOFIA',␣
 ↪'IN_DISC_ESTUDOS_SOCIAIS', 'IN_DISC_EST_SOCIAIS_SOCIOLOGIA',␣
 ↪'IN_DISC_INFORMATICA_COMPUTACAO', 'IN_DISC_ENSINO_RELIGIOSO',␣
 ↪'IN_DISC_PROFISSIONALIZANTE', 'IN_DISC_ESTAGIO_SUPERVISIONADO',␣
 ↪'IN_DISC_PEDAGOGICAS', 'IN_DISC_OUTRAS', 'CO_REGIAO', 'CO_MESORREGIAO',␣
 ↪'CO_MICRORREGIAO', 'CO_MUNICIPIO', 'CO_DISTRITO', 'TP_DEPENDENCIA',␣
 ↪'TP_LOCALIZACAO', 'TP_CATEGORIA_ESCOLA_PRIVADA', 'IN_CONVENIADA_PP',␣
 ↪'TP_CONVENIO_PODER_PUBLICO', 'IN_MANT_ESCOLA_PRIVADA_EMP',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_ONG', 'IN_MANT_ESCOLA_PRIVADA_OSCIP',␣
 ↪'IN_MANT_ESCOLA_PRIV_ONG_OSCIP', 'IN_MANT_ESCOLA_PRIVADA_SIND',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_SIST_S', 'IN_MANT_ESCOLA_PRIVADA_S_FINS',␣
 ↪'TP_REGULAMENTACAO', 'TP_LOCALIZACAO_DIFERENCIADA', 'IN_EDUCACAO_INDIGENA',␣
 ↪'IN_DIA_SEMANA_DOMINGO', 'IN_DIA_SEMANA_SEGUNDA', 'IN_DIA_SEMANA_TERCA',␣
 ↪'IN_DIA_SEMANA_QUARTA', 'IN_DIA_SEMANA_QUINTA', 'IN_DIA_SEMANA_SEXTA',␣
 ↪'IN_DIA_SEMANA_SABADO'])

# Cria um novo dataframe apenas contendo as turmas cuja escola seja do DF, para␣
 ↪tal comparação utilizamos a lista com cs códigos das escolas criada␣
 ↪anteriormente
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 dados do DF
turma_df.to_csv(r'C:
 ↪\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.
 ↪1_FBD\Trabalho_Individual\Dados\turmas_DF.csv', sep=';',␣
 ↪encoding='ISO-8859-1', index = False)
```

[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', 'CO_UF_NASC', 'CO_MUNICIPIO_NASC', 'CO_UF_END',␣
 ↪'CO_MUNICIPIO_END', 'TP_ZONA_RESIDENCIAL', 'TP_LOCAL_RESID_DIFERENCIADA',␣
 ↪'IN_BAIXA_VISAO', 'IN_CEGUEIRA', 'IN_DEF_AUDITIVA', 'IN_DEF_INTELECTUAL',␣
 ↪'IN_SURDEZ', 'IN_SURDOCEGUEIRA', 'IN_DEF_MULTIPLA', 'IN_AUTISMO',␣
 ↪'IN_RECURSO_LEDOR', 'IN_RECURSO_TRANSCRICAO', 'IN_RECURSO_INTERPRETE',␣
 ↪'IN_RECURSO_LIBRAS', 'IN_RECURSO_LABIAL', 'IN_RECURSO_AMPLIADA_18',␣
 ↪'IN_RECURSO_AMPLIADA_24', 'IN_RECURSO_CD_AUDIO',␣
 ↪'IN_RECURSO_PROVA_PORTUGUES', 'IN_RECURSO_VIDEO_LIBRAS',␣
 ↪'IN_RECURSO_BRAILLE', 'IN_RECURSO_NENHUM', 'IN_AEE_LIBRAS',␣
 ↪'IN_AEE_LINGUA_PORTUGUESA', 'IN_AEE_INFORMATICA_ACESSIVEL',␣
 ↪'IN_AEE_BRAILLE', 'IN_AEE_CAA', 'IN_AEE_SOROBAN', 'IN_AEE_VIDA_AUTONOMA',␣
 ↪'IN_AEE_OPTICOS_NAO_OPTICOS', 'IN_AEE_ENRIQ_CURRICULAR',␣
 ↪'IN_AEE_DESEN_COGNITIVO', 'IN_AEE_MOBILIDADE', 'TP_OUTRO_LOCAL_AULA',␣
 ↪'IN_TRANSPORTE_PUBLICO', 'TP_RESPONSAVEL_TRANSPORTE', 'IN_TRANSP_BICICLETA',␣
 ↪'IN_TRANSP_MICRO_ONIBUS', 'IN_TRANSP_ONIBUS', 'IN_TRANSP_TR_ANIMAL',␣
 ↪'IN_TRANSP_VANS_KOMBI', 'IN_TRANSP_OUTRO_VEICULO', 'IN_TRANSP_EMBAR_ATE5',␣
 ↪'IN_TRANSP_EMBAR_5A15', 'IN_TRANSP_EMBAR_15A35', 'IN_TRANSP_EMBAR_35',␣
 ↪'TP_ETAPA_ENSINO', 'IN_ESPECIAL_EXCLUSIVA', 'IN_REGULAR', 'IN_EJA',␣
 ↪'IN_PROFISSIONALIZANTE', 'CO_CURSO_EDUC_PROFISSIONAL',␣
 ↪'TP_MEDIACAO_DIDATICO_PEDAGO', 'NU_DURACAO_TURMA',␣
 ↪'NU_DUR_ATIV_COMP_MESMA_REDE', 'NU_DUR_ATIV_COMP_OUTRAS_REDES',␣
 ↪'NU_DUR_AEE_MESMA_REDE', 'NU_DUR_AEE_OUTRAS_REDES', 'NU_DIAS_ATIVIDADE',␣
 ↪'TP_UNIFICADA', 'TP_TIPO_ATENDIMENTO_TURMA', 'TP_TIPO_LOCAL_TURMA',␣
 ↪'CO_REGIAO', 'CO_MESORREGIAO', 'CO_MICRORREGIAO', 'CO_MUNICIPIO',␣
 ↪'CO_DISTRITO', 'TP_DEPENDENCIA', 'TP_LOCALIZACAO',␣
 ↪'TP_CATEGORIA_ESCOLA_PRIVADA', 'IN_CONVENIADA_PP',␣
 ↪'TP_CONVENIO_PODER_PUBLICO', 'IN_MANT_ESCOLA_PRIVADA_EMP',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_ONG', 'IN_MANT_ESCOLA_PRIVADA_OSCIP',␣
 ↪'IN_MANT_ESCOLA_PRIV_ONG_OSCIP', 'IN_MANT_ESCOLA_PRIVADA_SIND',␣
 ↪'IN_MANT_ESCOLA_PRIVADA_SIST_S', 'IN_MANT_ESCOLA_PRIVADA_S_FINS',␣
 ↪'TP_REGULAMENTACAO', 'TP_LOCALIZACAO_DIFERENCIADA', 'IN_EDUCACAO_INDIGENA'])

# Cria um novo dataframe apenas contendo os alunos cuja escola seja do DF, para␣
 ↪tal comparação utilizamos a lista com cs códigos das escolas criada␣
 ↪anteriormente
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 dados do DF
alunos_df.to_csv(r'C:
 ↪\Users\eduar\Documents\programming\python\Mestrado_PPCA_UnB\2021.
 ↪1_FBD\Trabalho_Individual\Dados\matricula_df.csv', sep=';',␣
 ↪encoding='ISO-8859-1', index = False)
```

1.0.7 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:

1.0.8 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:

1.0.9 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.

1.0.10 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

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


-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;
```

```sql
-- -----------------------------------------------------
-- 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;
```

[3]:
```
import mysql.connector as msql
from mysql.connector import Error
```

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

[8]:
```
import mysql.connector as msql
from mysql.connector import Error
```

```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("You're connected to database: ", record)

        #loop through the data frame
        for i,row in escolas_df.iterrows():
            #here %S means string values
```

```
            cursor.execute("""INSERT INTO escola (CO_ENTIDADE, NO_ENTIDADE,␣
↪CO_ORGAO_REGIONAL, TP_SITUACAO_FUNCIONAMENTO, DT_ANO_LETIVO_INICIO,␣
↪DT_ANO_LETIVO_TERMINO, CO_UF, TP_DEPENDENCIA, TP_LOCALIZACAO,␣
↪IN_VINCULO_SECRETARIA_EDUCACAO, IN_VINCULO_SEGURANCA_PUBLICA,␣
↪IN_LOCAL_FUNC_PREDIO_ESCOLAR, TP_OCUPACAO_PREDIO_ESCOLAR,␣
↪IN_LOCAL_FUNC_SOCIOEDUCATIVO, IN_LOCAL_FUNC_UNID_PRISIONAL,␣
↪IN_LOCAL_FUNC_PRISIONAL_SOCIO, IN_AGUA_POTAVEL, IN_ENERGIA_REDE_PUBLICA,␣
↪IN_ENERGIA_RENOVAVEL, IN_ESGOTO_REDE_PUBLICA, IN_LIXO_SERVICO_COLETA,␣
↪IN_AREA_VERDE,  IN_AUDITORIO, IN_BANHEIRO, IN_BANHEIRO_PNE, IN_BIBLIOTECA,␣
↪IN_COZINHA, IN_LABORATORIO_CIENCIAS, IN_LABORATORIO_INFORMATICA,␣
↪IN_PATIO_COBERTO, IN_PATIO_DESCOBERTO, IN_PARQUE_INFANTIL, IN_PISCINA,␣
↪IN_QUADRA_ESPORTES, IN_QUADRA_ESPORTES_COBERTA,␣
↪IN_QUADRA_ESPORTES_DESCOBERTA, IN_REFEITORIO, IN_SALA_ATELIE_ARTES,␣
↪IN_DESKTOP_ALUNO, QT_DESKTOP_ALUNO, IN_COMP_PORTATIL_ALUNO,␣
↪QT_COMP_PORTATIL_ALUNO, IN_TABLET_ALUNO, QT_TABLET_ALUNO, IN_INTERNET,␣
↪IN_INTERNET_ALUNOS, TP_REDE_LOCAL, IN_BANDA_LARGA, QT_PROF_FONAUDIOLOGO,␣
↪QT_PROF_NUTRICIONISTA, QT_PROF_PSICOLOGO, IN_ALIMENTACAO) VALUES (%s, %s,␣
↪%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,␣
↪%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,␣
↪%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)""", (row["CO_ENTIDADE"],␣
↪row["NO_ENTIDADE"], row["CO_ORGAO_REGIONAL"],␣
↪row["TP_SITUACAO_FUNCIONAMENTO"], row["DT_ANO_LETIVO_INICIO"],␣
↪row["DT_ANO_LETIVO_TERMINO"], row["CO_UF"], row["TP_DEPENDENCIA"],␣
↪row["TP_LOCALIZACAO"], row["IN_VINCULO_SECRETARIA_EDUCACAO"],␣
↪row["IN_VINCULO_SEGURANCA_PUBLICA"], row["IN_LOCAL_FUNC_PREDIO_ESCOLAR"],␣
↪row["TP_OCUPACAO_PREDIO_ESCOLAR"], row["IN_LOCAL_FUNC_SOCIOEDUCATIVO"],␣
↪row["IN_LOCAL_FUNC_UNID_PRISIONAL"], row["IN_LOCAL_FUNC_PRISIONAL_SOCIO"],␣
↪row["IN_AGUA_POTAVEL"], row["IN_ENERGIA_REDE_PUBLICA"],␣
↪row["IN_ENERGIA_RENOVAVEL"], row["IN_ESGOTO_REDE_PUBLICA"],␣
↪row["IN_LIXO_SERVICO_COLETA"], row["IN_AREA_VERDE"], row["IN_AUDITORIO"],␣
↪row["IN_BANHEIRO"], row["IN_BANHEIRO_PNE"], row["IN_BIBLIOTECA"],␣
↪row["IN_COZINHA"], row["IN_LABORATORIO_CIENCIAS"],␣
↪row["IN_LABORATORIO_INFORMATICA"], row["IN_PATIO_COBERTO"],␣
↪row["IN_PATIO_DESCOBERTO"], row["IN_PARQUE_INFANTIL"], row["IN_PISCINA"],␣
↪row["IN_QUADRA_ESPORTES"], row["IN_QUADRA_ESPORTES_COBERTA"],␣
↪row["IN_QUADRA_ESPORTES_DESCOBERTA"], row["IN_REFEITORIO"],␣
↪row["IN_SALA_ATELIE_ARTES"], row["IN_DESKTOP_ALUNO"],␣
↪row["QT_DESKTOP_ALUNO"], row["IN_COMP_PORTATIL_ALUNO"],␣
↪row["QT_COMP_PORTATIL_ALUNO"], row["IN_TABLET_ALUNO"],␣
↪row["QT_TABLET_ALUNO"], row["IN_INTERNET"], row["IN_INTERNET_ALUNOS"],␣
↪row["TP_REDE_LOCAL"], row["IN_BANDA_LARGA"], row["QT_PROF_FONAUDIOLOGO"],␣
↪row["QT_PROF_NUTRICIONISTA"], row["QT_PROF_PSICOLOGO"],␣
↪row["IN_ALIMENTACAO"]))
            # the connection is not auto committed by default, so we must␣
↪commit to save our changes
            conn.commit()
        print("Registros inseridos")
```

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

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

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

[9]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("You're connected to database: ", record)

        #loop through the data frame
        for i,row in turma_df.iterrows():
            #here %S means string values
            cursor.execute("""INSERT INTO turma (ID_TURMA, NO_TURMA,␣
 ↪TP_MEDIACAO_DIDATICO_PEDAGO, TX_HR_INICIAL, TX_MI_INICIAL,␣
 ↪NU_DIAS_ATIVIDADE, QT_MATRICULAS) VALUES (%s, %s, %s, %s, %s, %s, %s)""",␣
 ↪(row["ID_TURMA"], row["NO_TURMA"], row["TP_MEDIACAO_DIDATICO_PEDAGO"],␣
 ↪row["TX_HR_INICIAL"], row["TX_MI_INICIAL"], row["NU_DIAS_ATIVIDADE"],␣
 ↪row["QT_MATRICULAS"]))
            # the connection is not auto committed by default, so we must␣
 ↪commit to save our changes
            conn.commit()
        print("Registros inseridos")
except Error as e:
            print("Error while connecting to MySQL", e)
```

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

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

[10]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("You're connected to database: ", record)

        #loop through the data frame
```

```
        for i,row in alunos_df.iterrows():
            #here %S means string values
            cursor.execute("""INSERT INTO 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_TURMA,␣
↪escola_CO_ENTIDADE) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,␣
↪%s)""", (row["ID_ALUNO"], row["ID_MATRICULA"], row["NU_MES"], row["NU_ANO"],␣
↪row["NU_IDADE"], row["TP_SEXO"], row["TP_COR_RACA"],␣
↪row["TP_NACIONALIDADE"], row["IN_NECESSIDADE_ESPECIAL"],␣
↪row["IN_DEF_FISICA"], row["IN_SUPERDOTACAO"], row["ID_TURMA"],␣
↪row["CO_ENTIDADE"]))
            # the connection is not auto committed by default, so we must␣
↪commit to save our changes
            conn.commit()
        print("Registros inseridos")
except Error as e:
            print("Error while connecting to MySQL", e)
```

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

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

[11]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("You're connected to database: ", record)

        #loop through the data frame
        for i,row in docentes_df.iterrows():
            #here %S means string values
            cursor.execute("""INSERT INTO docente (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_ESPECIALIZACAO,␣
↪IN_MESTRADO, IN_DOUTORADO, IN_POS_NENHUM, TP_TIPO_DOCENTE,␣
↪TP_TIPO_CONTRATACAO) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,␣
↪%s, %s, %s, %s, %s)""", (row["ID_DOCENTE"], row["NU_MES"], row["NU_ANO"],␣
↪row["NU_IDADE"], row["TP_SEXO"], row["TP_COR_RACA"],␣
↪row["TP_NACIONALIDADE"], row["CO_UF_NASC"], row["IN_NECESSIDADE_ESPECIAL"],␣
↪row["IN_DEF_FISICA"], row["IN_SUPERDOTACAO"], row["IN_ESPECIALIZACAO"],␣
↪row["IN_MESTRADO"], row["IN_DOUTORADO"], row["IN_POS_NENHUM"],␣
↪row["TP_TIPO_DOCENTE"], row["TP_TIPO_CONTRATACAO"]))
```

```
            primary_key = cursor.lastrowid
            cursor.execute("""INSERT INTO docente_has_turma (docente_idDocente,␣
 ↪turma_ID_TURMA) VALUES (%s, %s)""", (primary_key, row["ID_TURMA"]))
            cursor.execute("""INSERT INTO docente_has_escola␣
 ↪(docente_idDocente, escola_CO_ENTIDADE) VALUES (%s, %s)""", (primary_key,␣
 ↪row["CO_ENTIDADE"]))
            # the connection is not auto committed by default, so we must␣
 ↪commit to save our changes
            conn.commit()
        print("Registros inseridos")
except Error as e:
            print("Error while connecting to MySQL", e)
```

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

1.0.15 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:

1.0.16 - Tabela Aluno

1.0.17 - Tabela Docente

1.0.18 - Tabela Docente_has_escola

1.0.19 - Tabela Docente_has_turma

1.0.20 - Tabela Escola

1.0.21 - Tabela Turma

1.0.22 Consultas SQL

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

[6]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT * FROM mydb.docente WHERE␣
 ↪IN_NECESSIDADE_ESPECIAL = 1""")
        myresult = cursor.fetchall()

        print("Quantidade de professores com algum tipo de necessidade especial␣
 ↪(deficiência física, autismo, etc) no DF: " + str(len(myresult)))
```

```
        #for x in myresult:
        #    print(x)
        # the connection is not auto committed by default, so we must commit to␣
 ↪save our changes
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:

[7]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT * FROM mydb.docente WHERE IN_MESTRADO = 1 or␣
 ↪IN_DOUTORADO = 1""")
        myresult = cursor.fetchall()

        print("Quantidade de professores com mestrado ou doutorado no DF: " +␣
 ↪str(len(myresult)))
        #for x in myresult:
        #    print(x)
        # the connection is not auto committed by default, so we must commit to␣
 ↪save our changes
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:

[8]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT * FROM mydb.escola WHERE IN_AGUA_POTAVEL = 0␣
 ↪or IN_ENERGIA_REDE_PUBLICA = 0""")
```

```
        myresult = cursor.fetchall()

        print("Quantidade de escolas que não tenham água potável ou não tenham␣
 ↪energia elétrica no DF: " + str(len(myresult)))
        #for x in myresult:
        #    print(x)
        # the connection is not auto committed by default, so we must commit to␣
 ↪save our changes
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 Psicologo, Nutricionista ou Fonoaudiologo:

[9]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT * FROM mydb.escola WHERE QT_PROF_FONAUDIOLOGO␣
 ↪= 0 or QT_PROF_NUTRICIONISTA = 0 or QT_PROF_PSICOLOGO = 0""")
        myresult = cursor.fetchall()

        print("Quantidade de escolas que tenham psicológo, nutricionista ou␣
 ↪fonoaudiólogo no DF: " + str(len(myresult)))

        #for x in myresult:
        #    print(x)
        # the connection is not auto committed by default, so we must commit to␣
 ↪save our changes
except Error as e:
            print("Error while connecting to MySQL", e)
```

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

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

[10]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
```

```
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT * FROM mydb.escola WHERE IN_DESKTOP_ALUNO = 1␣
↪or IN_COMP_PORTATIL_ALUNO = 1 or IN_TABLET_ALUNO = 1""")
        myresult = cursor.fetchall()

        print("Quantidade de escolas que tenham computadores, notebooks ou␣
↪tablets a disposição dos estudantes no DF: " + str(len(myresult)))

        #for x in myresult:
        #    print(x)
        # the connection is not auto committed by default, so we must commit to␣
↪save our changes
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:

[11]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT COUNT(ID_DOCENTE) FROM mydb.docente WHERE␣
↪TP_COR_RACA = 1""")
        myresult = cursor.fetchone()
        prof_branco = myresult[0]

        cursor.execute("""SELECT COUNT(ID_DOCENTE) FROM mydb.docente WHERE␣
↪TP_COR_RACA = 2""")
        myresult = cursor.fetchone()
        prof_negro = myresult[0]

        cursor.execute("""SELECT COUNT(ID_DOCENTE) FROM mydb.docente WHERE␣
↪TP_COR_RACA = 3""")
        myresult = cursor.fetchone()
        prof_pardo = myresult[0]

        cursor.execute("""SELECT COUNT(ID_DOCENTE) FROM mydb.docente WHERE␣
↪TP_COR_RACA = 4""")
```

```
        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))


        # the connection is not auto committed by default, so we must commit to␣
↪save our changes
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:

[12]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT COUNT(ID_ALUNO) FROM mydb.aluno WHERE NU_IDADE␣
↪> 18""")
        myresult = cursor.fetchone()
        qtd_alunos_maior = myresult[0]

        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))

        # the connection is not auto committed by default, so we must commit to␣
↪save our changes
```

```
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:

[13]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT COUNT(CO_ENTIDADE) FROM mydb.escola WHERE␣
↪IN_VINCULO_SECRETARIA_EDUCACAO = 1""")
        myresult = cursor.fetchone()
        secretaria = myresult[0]

        cursor.execute("""SELECT COUNT(CO_ENTIDADE) FROM mydb.escola WHERE␣
↪IN_VINCULO_SEGURANCA_PUBLICA = 1""")
        myresult = cursor.fetchone()
        seguranca = myresult[0]

        print("Quantidade de escolas vinculadas à Secretaria de Estado de␣
↪Educação do Distrito Federal: " + str(secretaria))
        print("Quantidade de escolas vinculadas à Secretaria de Estado de␣
↪Segurança Pública do Distrito Federal (conhecidas enquanto \"escolas␣
↪militarizadas\"): " + str(seguranca))



        # the connection is not auto committed by default, so we must commit to␣
↪save our changes
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) Lista de Escolas do Distrito Federal que tenham internet para os estudantes:

[14]:
```
try:
```

```
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        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: " + str(len(myresult)))
        #for x in myresult:
        #    print(x)
        # the connection is not auto committed by default, so we must commit to␣
↪save our changes
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) Lista de Escolas do Distrito Federal que tenham investido em energias renováveis:

[15]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        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: " + str(len(myresult)))
        #for x in myresult:
        #    print(x)
        # the connection is not auto committed by default, so we must commit to␣
↪save our changes
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:

[3]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT ID_ALUNO, NU_IDADE, TP_SEXO, TP_COR_RACA FROM␣
↪mydb.aluno WHERE NU_IDADE > 18 ORDER BY NU_IDADE DESC""")

        myresult = cursor.fetchall()

        print("Quantidade de alunos com mais de 18 anos: " + str(len(myresult)))
        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)

        # the connection is not auto committed by default, so we must commit to␣
↪save our changes
except Error as e:
            print("Error while connecting to MySQL", e)
```

```
Quantidade de alunos com mais de 18 anos: 64818
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 menos de 18 anos, ordenado pelo mais jovem:

[4]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
```

```
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT ID_ALUNO, NU_IDADE, TP_SEXO, TP_COR_RACA FROM␣
↪mydb.aluno WHERE NU_IDADE < 18 ORDER BY NU_IDADE ASC""")

        myresult = cursor.fetchall()

        print("Quantidade de alunos com menos de 18 anos: " +␣
↪str(len(myresult)))
        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)

        # the connection is not auto committed by default, so we must commit to␣
↪save our changes
except Error as e:
            print("Error while connecting to MySQL", e)
```

```
Quantidade de alunos com menos de 18 anos: 626825
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:

[22]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT ID_DOCENTE, NU_IDADE, TP_SEXO, TP_COR_RACA␣
↪FROM mydb.docente WHERE NU_IDADE > 40 ORDER BY NU_IDADE DESC""")
```

```
        myresult = cursor.fetchall()

        print("Quantidade de professores com MAIS de 40 anos no DF: " +␣
↪str(len(myresult)))
        print("Lista com os 5 professores mais velhos da rede pública do␣
↪Distrito Federal: ")

        myresult_sliced = myresult[:5]

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

        for x in myresult_sliced:
            print(x)

        # the connection is not auto committed by default, so we must commit to␣
↪save our changes
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:

[113]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT ID_DOCENTE, NU_IDADE, TP_SEXO, TP_COR_RACA␣
↪FROM mydb.docente WHERE NU_IDADE < 40 ORDER BY NU_IDADE ASC""")

        myresult = cursor.fetchall()

        print("Quantidade de professores com MENOS de 40 anos no DF: " +␣
↪str(len(myresult)))
        print("Lista com os 5 professores mais jovens da rede pública do␣
↪Distrito Federal: ")
```

```
        myresult_sliced = myresult[:5]

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

        for x in myresult_sliced:
            print(x)

        # the connection is not auto committed by default, so we must commit to␣
 ↪save our changes
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)
```

15) Quantidade de professores do Leonardo da Vinci e lista ordenada com os 5 mais velhos:

[76]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT DISTINCT ID_DOCENTE, NU_IDADE, TP_SEXO,␣
 ↪TP_COR_RACA FROM mydb.docente docente
        JOIN mydb.docente_has_escola ats ON docente.idDocente = ats.
 ↪docente_idDocente WHERE
        ats.escola_CO_ENTIDADE = '53001087' ORDER BY NU_IDADE DESC""")

        myresult = cursor.fetchall()

        print("Quantidade de professores no Leonardo da Vinci: " +␣
 ↪str(len(myresult)))
        print("Lista com os 5 professores mais velhos da escola Leonardo da␣
 ↪Vinci: ")

        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 no Leonardo da Vinci: 108
Lista com os 5 professores mais velhos da escola Leonardo da Vinci:
(ID_DOCENTE, NU_IDADE, TP_SEXO, TP_COR_RACA)
('37B754B9337F6B41EA56DAE55C42C751', 65, 2, 0)
('30E4CD5D80907F8001766003504C4010', 63, 1, 3)
('0D430747EB2CEDF739C56BB5E8EEE933', 59, 1, 0)
('54D1555B556849FCB9F0EC0751D71B6F', 56, 2, 0)
('917FB4F2182D35316CA5E7C01E70186D', 56, 1, 1)
```

16) Quantidade de professores da Escola das Nações e lista ordenada com os 5 mais jovens:

[78]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT DISTINCT ID_DOCENTE, NU_IDADE, TP_SEXO,␣
↪TP_COR_RACA FROM mydb.docente docente
        JOIN mydb.docente_has_escola ats ON docente.idDocente = ats.
↪docente_idDocente WHERE
        ats.escola_CO_ENTIDADE = '53009541' ORDER BY NU_IDADE ASC""")

        myresult = cursor.fetchall()

        print("Quantidade de professores na Escola das Nações: " +␣
↪str(len(myresult)))
        print("Lista com os 5 professores mais novos da escola Escola das␣
↪Nações: ")

        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 na Escola das Nações: 66
Lista com os 5 professores mais novos da escola Escola das Nações:
(ID_DOCENTE, NU_IDADE, TP_SEXO, TP_COR_RACA)
('1D668089D8944873BA38E652DA52A308', 26, 1, 0)
('A0A02C991557C705AA02CDB5193B205D', 26, 2, 1)
('43BF4DB881B59713EEB67D6F8EE29401', 26, 2, 0)
('91D45F5B8D30E58CAF0CB8D58A271E1C', 28, 2, 0)
('DD7868FD1DC2FBBFA395B114F276F47B', 28, 1, 3)
```

17) Professor com a maior quantidade de turmas do DF e escolas onde o mesmo trabalha:

[105]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""select *, count(*) as c FROM mydb.docente GROUP BY␣
↪ID_DOCENTE ORDER BY c DESC""")

        myresult = cursor.fetchone()
        qtdDeTurmas = myresult[-1]
        ID_DOCENTE = myresult[1]
        idade = myresult[4]
        sexo = myresult[5]

        # the connection is not auto committed by default, so we must commit to␣
↪save our changes
except Error as e:
            print("Error while connecting to MySQL", e)
```

[106]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT idDocente FROM mydb.docente WHERE ID_DOCENTE =␣
↪'""" + ID_DOCENTE + """'""")
```

```
        myresult = cursor.fetchall()

        ids = []

        for element in myresult:
            ids.append(str(element[0]))
        # the connection is not auto committed by default, so we must commit to␣
 ↪save our changes
except Error as e:
            print("Error while connecting to MySQL", e)
```

[107]:
```
escolas = []
for element in ids:
    try:
        conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
        if conn.is_connected():
            cursor = conn.cursor()
            cursor.execute("select database();")
            record = cursor.fetchone()


            #cursor.execute("""SELECT escola_CO_ENTIDADE FROM mydb.escola JOIN␣
 ↪mydb.docente_has_escola ats WHERE ats.docente_idDocente = '121584'""")
            cursor.execute("""SELECT escola_CO_ENTIDADE FROM mydb.escola JOIN␣
 ↪mydb.docente_has_escola ats WHERE ats.docente_idDocente = '""" + element +␣
 ↪"""'""")
            myresult = cursor.fetchone()

            escolas.append(myresult[0])
            # the connection is not auto committed by default, so we must␣
 ↪commit to save our changes
    except Error as e:
                print("Error while connecting to MySQL", e)
```

17) Professor com a maior quantidade de turmas do DF e escolas onde o mesmo trabalha:

[108]:
```
escolas = set(escolas)

print("Quem é o professor com a maior quantidade de turmas no Distrito Federal:␣
 ↪")

print("ID: " + str(ID_DOCENTE))
print("IDADE: " + str(idade))
if sexo == 1:
    print("SEXO: MASCULINO")
```

```
if sexo == 2:
    print("SEXO: FEMININO")
print("QUANTIDADE DE TURMAS: " + str(qtdDeTurmas))
print("ESCOLAS ONDE ELE OU ELA TRABALHA: ")

for escola in escolas:
    try:
        conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
        if conn.is_connected():
            cursor = conn.cursor()
            cursor.execute("select database();")
            record = cursor.fetchone()

            cursor.execute("""SELECT NO_ENTIDADE FROM mydb.escola WHERE␣
 ↪CO_ENTIDADE = '""" + str(escola) + """'""")
            myresult = cursor.fetchone()

            print("- " + myresult[0])

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

```
Quem é o professor com a maior quantidade de turmas no Distrito Federal:
ID: CF7150E2A634CD8D5563D998020DAF62
IDADE: 29
SEXO: MASCULINO
QUANTIDADE DE TURMAS: 57
ESCOLAS ONDE ELE OU ELA TRABALHA:
- COL OBJETIVO DF - UNIDADE VI
- COL OBJETIVO DF - UNIDADE III
- COL OBJETIVO DF - UNIDADE V
- COL OBJETIVO DF - UNIDADE II
- COL OBJETIVO DF - UNIDADE VII
```

17) Professor com a menor quantidade de turmas do DF e escolas onde o mesmo trabalha:

[109]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""select *, count(*) as c FROM mydb.docente GROUP BY␣
 ↪ID_DOCENTE ORDER BY c ASC""")
```

```
        myresult = cursor.fetchone()
        qtdDeTurmas = myresult[-1]
        ID_DOCENTE = myresult[1]
        idade = myresult[4]
        sexo = myresult[5]

        # the connection is not auto committed by default, so we must commit to␣
 ↪save our changes
except Error as e:
            print("Error while connecting to MySQL", e)
```

[110]:
```
try:
    conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()

        cursor.execute("""SELECT idDocente FROM mydb.docente WHERE ID_DOCENTE =␣
 ↪'""" + ID_DOCENTE + """'""")

        myresult = cursor.fetchall()

        ids = []

        for element in myresult:
            ids.append(str(element[0]))
        # the connection is not auto committed by default, so we must commit to␣
 ↪save our changes
except Error as e:
            print("Error while connecting to MySQL", e)
```

[111]:
```
escolas = []
for element in ids:
    try:
        conn = msql.connect(host='localhost', database='mydb', user='root',␣
 ↪password='root')
        if conn.is_connected():
            cursor = conn.cursor()
            cursor.execute("select database();")
            record = cursor.fetchone()


            #cursor.execute("""SELECT escola_CO_ENTIDADE FROM mydb.escola JOIN␣
 ↪mydb.docente_has_escola ats WHERE ats.docente_idDocente = '121584'""")
```

```
            cursor.execute("""SELECT escola_CO_ENTIDADE FROM mydb.escola JOIN␣
↪mydb.docente_has_escola ats WHERE ats.docente_idDocente = '""" + element +␣
↪"""'""")
            myresult = cursor.fetchone()

            escolas.append(myresult[0])
            # the connection is not auto committed by default, so we must␣
↪commit to save our changes
    except Error as e:
                print("Error while connecting to MySQL", e)
```

17) Professor com a menor quantidade de turmas do DF e escolas onde o mesmo trabalha:

[112]:
```
escolas = set(escolas)

print("Quem é o professor com a maior quantidade de turmas no Distrito Federal:␣
↪")

print("ID: " + str(ID_DOCENTE))
print("IDADE: " + str(idade))
if sexo == 1:
    print("SEXO: MASCULINO")
if sexo == 2:
    print("SEXO: FEMININO")
print("QUANTIDADE DE TURMAS: " + str(qtdDeTurmas))
print("ESCOLAS ONDE ELE OU ELA TRABALHA: ")

for escola in escolas:
    try:
        conn = msql.connect(host='localhost', database='mydb', user='root',␣
↪password='root')
        if conn.is_connected():
            cursor = conn.cursor()
            cursor.execute("select database();")
            record = cursor.fetchone()

            cursor.execute("""SELECT NO_ENTIDADE FROM mydb.escola WHERE␣
↪CO_ENTIDADE = '""" + str(escola) + """'""")
            myresult = cursor.fetchone()

            print("- " + myresult[0])
    #         print("Quem é o professor com a maior quantidade de turmas no␣
↪Distrito Federal: ")

    #         print("ID: " + str(x[0]))
    #         print("ID: " + str(x[1]))
```

```
    #         print("IDADE: " + str(x[4]))
    #         print("SEXO (1 - MASCULINO; 2 - FEMININO): " + str(x[5]))
    #         print("QUANTIDADE DE TURMAS: " + str(x[-1]))

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

```
Quem é o professor com a maior quantidade de turmas no Distrito Federal:
ID: F6BF550E658CFAB20D429148C044BBE9
IDADE: 40
SEXO: FEMININO
QUANTIDADE DE TURMAS: 1
ESCOLAS ONDE ELE OU ELA TRABALHA:
- EC 38 DE CEILANDIA
```

[ ]: