Entrega 1: Construção de base de consultas

Consultas

CCB

Quantas pessoas estão cadastradas no cadastro único?

```
SELECT
SUM(qtde_pessoas)
FROM tb_familia
```

Quantos dos beneficiários são homens?

```
SELECT
COUNT(DISTINCT(id_pessoa))
FROM tb_pessoa
WHERE cod_sexo_pessoa = 1
```

Quantas das beneficiárias são mulheres?

```
SELECT
COUNT(DISTINCT(id_pessoa))
FROM tb_pessoa
WHERE cod_sexo_pessoa = 2
```

Quais os graus de parentesco possíveis?

```
SELECT
DISTINCT(cod_parentesco_rf_pessoa)
FROM tb_pessoa
```

Quais os níveis de escolaridade dos beneficiários do cadastro único?

```
SELECT
DISTINCT(COD_CURSO_FREQUENTOU_PESSOA_MEMB)
FROM tb_esc
```

CCI

Qual a média de cômodos em relação a quantidade de pessoas na família?

```
SELECT
qtde_pessoas,
AVG(qtd_comodos_dormitorio_fam)
FROM tb_familia
JOIN tb_domicilio ON tb_familia.id_familia=tb_domicilio.id_familia
GROUP BY qtde_pessoas
```

Qual o principal trabalho de uma mulher?

```
SELECT
cod_sexo_pessoa,
tb_trab.cod_principal_trab_memb
```

```
FROM tb_pessoa
JOIN tb_trab ON tb_pessoa.id_pessoa=tb_trab.id_pessoa
WHERE tb_pessoa.cod_sexo_pessoa=2
```

Qual o número de famílias cadastradas por estado?

```
SELECT
tb_mun.uf,
COUNT(DISTINCT(tb_familia.id_familia)) AS numero_de_familias
FROM tb_familia
JOIN tb_mun ON tb_familia.cd_ibge=tb_mun.cd_ibge
GROUP BY tb_mun.uf
```

Qual o grau de escolaridade dos filhos?

```
SELECT
DISTINCT(tb_pessoa.id_pessoa),
tb_esc.cod_ano_serie_frequenta_memb
FROM tb_pessoa
JOIN tb_esc ON tb_pessoa.id_pessoa=tb_esc.id_pessoa
WHERE tb_pessoa.cod_parentesco_rf_pessoa=3
```

Quantas famílias possuem saneamento básico e um banheiro?

```
SELECT
COUNT(DISTINCT(tb_domicilio.id_familia))
FROM tb_domicilio
WHERE cod_banheiro_domic_fam=1 AND cod_escoa_sanitario_domic_fam=1
```

CCA

Qual a média da renda bruta anual de pessoas do nordeste?

```
SELECT
regiao,
AVG(val_renda_bruta_12_meses_memb)
FROM tb_trab
JOIN tb_pessoa ON tb_trab.id_pessoa = tb_pessoa.id_pessoa
JOIN tb_familia ON tb_pessoa.id_familia = tb_familia.id_familia
JOIN tb_mun ON tb_mun.cd_ibge = tb_familia.cd_ibge
WHERE tb_mun.regiao = '2 - Nordeste' GROUP BY tb_mun.regiao
```

Quantas pessoas negras do sudeste frequentam escola pública?

```
SELECT
COUNT(DISTINCT(tb_pessoa.id_pessoa))
FROM tb_mun
JOIN tb_familia ON tb_mun.cd_ibge=tb_familia.cd_ibge
JOIN tb_pessoa ON tb_familia.id_familia = tb_pessoa.id_familia
JOIN tb_esc ON tb_esc.id_pessoa=tb_pessoa .id_pessoa
WHERE tb_pessoa.cod_raca_cor_pessoa = 2 AND
tb_esc.ind_frequenta_escola_memb = 1 AND tb_mun.regiao = '3 - Sudeste'
```

```
GROUP BY tb_mun.regiao
```

Qual a média salarial dos estados da região Norte?

```
WITH tb_regiao AS (
SELECT
tb_familia.id_familia,
tb_mun.uf,
tb_mun.regiao
FROM tb_familia
JOIN tb_mun ON tb_familia.cd_ibge = tb_mun.cd_ibge)

SELECT
tb_regiao.uf,
AVG(tb_trab.val_remuner_emprego_memb)
FROM tb_trab
JOIN tb_regiao ON tb_trab.id_familia = tb_regiao.id_familia
WHERE tb_regiao.regiao = '1 - Norte'
GROUP BY tb_regiao.uf
```

Quais a quantidade de pessoas por faixas salariais?

```
SELECT
'Não recebeu' AS salario,
COUNT(val_remuner_emprego_memb)
FROM tb_trab
WHERE val_remuner_emprego_memb = 0
UNION
SELECT
'Recebeu até meio salário mínimo',
COUNT(val_remuner_emprego_memb)
FROM tb_trab
WHERE val_remuner_emprego_memb > 0 AND val_remuner_emprego_memb < 759
UNION
SELECT
'Recebeu até um salário mínimo',
COUNT(val_remuner_emprego_memb)
FROM tb_trab
WHERE val_remuner_emprego_memb > 758 AND val_remuner_emprego_memb < 1518
UNION
SELECT
'Recebeu até um e meio salário mínimo',
COUNT(val_remuner_emprego_memb)
FROM tb_trab
WHERE val_remuner_emprego_memb > 1517 AND val_remuner_emprego_memb <
2276
UNION
```

```
SELECT
'Recebeu mais de um e meio salário mínimo',
COUNT(val_remuner_emprego_memb)
FROM tb_trab
WHERE val_remuner_emprego_memb > 2275
```

Imigração afeta o acesso a água encanada?

```
SELECT tb_mun.nome_municipio,
COUNT(CASE WHEN tb_pessoa.cod_local_nascimento_pessoa = 1 AND
tb_domicilio.cod_agua_canalizada_fam = 1 THEN 1 END) AS originario,
COUNT(CASE WHEN tb_pessoa.cod_local_nascimento_pessoa = 2 AND
tb_domicilio.cod_agua_canalizada_fam = 1 THEN 1 END) AS imigrante,
COUNT(CASE WHEN tb_pessoa.cod_local_nascimento_pessoa = 3 AND
tb_domicilio.cod_agua_canalizada_fam = 1 THEN 1 END) AS estramgeiro
FROM tb_mun
JOIN tb_familia ON tb_mun.cd_ibge = tb_familia.cd_ibge
JOIN tb_pessoa ON tb_familia.id_familia = tb_pessoa.id_familia
JOIN tb_domicilio ON tb_pessoa.id_familia = tb_domicilio.id_familia
GROUP BY tb_mun.nome_municipio
```

Entrega 2: Plano de tuning e indexação

Tuning

Baixa alocação de memória

effective_cache_size = 2GB

Alocação de memória conservadora

effective_cache_size = 4GB

Alta alocação de memória

effective_cache_size = 8GB

Index

Indexar joins

```
CREATE INDEX idx_fam_familia ON tb_familia (id_familia)
CREATE INDEX idx_fam_cd_ibge ON tb_familia (cd_ibge)

CREATE INDEX idx_dom_familia ON tb_domicilio (id_familia)

CREATE INDEX idx_pes_familia ON tb_pessoa (id_familia)
CREATE INDEX idx_pes_pessoa ON tb_pessoa (id_pessoa)

CREATE INDEX idx_trab_familia ON tb_trab (id_familia)
CREATE INDEX idx_trab_pessoa ON tb_trab (id_pessoa)
```

```sql
CREATE INDEX idx_mun_cd_igbe ON tb_mun (cd_ibge)

CREATE INDEX idx_esc_familia ON tb_esc (id_familia)
CREATE INDEX idx_esc_pessoa ON tb_esc (id_pessoa)
```

Indexar where

```sql
CREATE INDEX idx_trab_remuner ON tb_trab (val_remuner_emprego_memb)

CREATE INDEX idx_reg_regiao ON tb_regiao (regiao)

CREATE INDEX idx_pes_raca ON tb_pessoa (cod_raca_cor_pessoa)
```

Indexar group by

```sql
CREATE INDEX idx_mun_municipio ON tb_mun (nome_municipio)

CREATE INDEX idx_mun_uf ON tb_mun (uf)

CREATE INDEX idx_mun_regiao ON tb_mun (regiao)
```

Rotina

```python
import time
import psycopg2

# Configuração de conexão
conn = psycopg2.connect(
    dbname='MATB09',
    user='postgres',
    password='admin',
    host='localhost',
    port='5432'
)

# Caminho do arquivo de configuração
tuning1 = "configs/postgresql1.conf"
tuning2 = "configs/postgresql2.conf"
tuning3 = "configs/postgresql3.conf"
conf_destino = "C:/Program Files/PostgreSQL/17/data/postgresql.conf"


# Queries a serem executadas
queries_consulta = [
   """SELECT
```

```
   SUM(qtde_pessoas)
   FROM tb_familia;""",
   """SELECT
   COUNT(DISTINCT(id_pessoa))
   FROM tb_pessoa
   WHERE cod_sexo_pessoa = 1;""",
   """SELECT
   COUNT(DISTINCT(id_pessoa))
   FROM tb_pessoa
   WHERE cod_sexo_pessoa = 2;""",
   """SELECT
   DISTINCT(cod_parentesco_rf_pessoa)
   FROM tb_pessoa;""",
   """SELECT
   DISTINCT(COD_CURSO_FREQUENTOU_PESSOA_MEMB)
   FROM tb_esc;""",
   """SELECT
   qtde_pessoas,
   AVG(qtd_comodos_dormitorio_fam)
   FROM tb_familia
   JOIN tb_domicilio ON tb_familia.id_familia=tb_domicilio.id_familia
   GROUP BY qtde_pessoas;""",
   """SELECT
   cod_sexo_pessoa,
   tb_trab.cod_principal_trab_memb
   FROM tb_pessoa
   JOIN tb_trab ON tb_pessoa.id_pessoa=tb_trab.id_pessoa
   WHERE tb_pessoa.cod_sexo_pessoa=2;""",
   """SELECT
   tb_mun.uf,
   COUNT(DISTINCT(tb_familia.id_familia)) AS numero_de_familias
   FROM tb_familia
   JOIN tb_mun ON tb_familia.cd_ibge=tb_mun.cd_ibge
   GROUP BY tb_mun.uf;""",
   """"SELECT
   DISTINCT(tb_pessoa.id_pessoa),
   tb_esc.cod_ano_serie_frequenta_memb
   FROM tb_pessoa
   JOIN tb_esc ON tb_pessoa.id_pessoa=tb_esc.id_pessoa
   WHERE tb_pessoa.cod_parentesco_rf_pessoa=3;""",
   """SELECT
   COUNT(DISTINCT(tb_domicilio.id_familia))
   FROM tb_domicilio
   WHERE cod_banheiro_domic_fam=1 AND
cod_escoa_sanitario_domic_fam=1;""",
   """SELECT
```

```
    regiao,
    AVG(val_renda_bruta_12_meses_memb)
    FROM tb_trab
    JOIN tb_pessoa ON tb_trab.id_pessoa = tb_pessoa.id_pessoa
    JOIN tb_familia ON tb_pessoa.id_familia = tb_familia.id_familia
    JOIN tb_mun ON tb_mun.cd_ibge = tb_familia.cd_ibge
    WHERE tb_mun.regiao = '2 - Nordeste' GROUP BY tb_mun.regiao;""",
    """SELECT
    COUNT(DISTINCT(tb_pessoa.id_pessoa))
    FROM tb_mun
    JOIN tb_familia ON tb_mun.cd_ibge=tb_familia.cd_ibge
    JOIN tb_pessoa ON tb_familia.id_familia = tb_pessoa.id_familia
    JOIN tb_esc ON tb_esc.id_pessoa=tb_pessoa .id_pessoa
    WHERE tb_pessoa.cod_raca_cor_pessoa = 2 AND
tb_esc.ind_frequenta_escola_memb = 1 AND tb_mun.regiao = '3 - Sudeste'
    GROUP BY tb_mun.regiao;""",
    """WITH tb_regiao AS (
    SELECT
    tb_familia.id_familia,
    tb_mun.uf,
    tb_mun.regiao
    FROM tb_familia
    JOIN tb_mun ON tb_familia.cd_ibge = tb_mun.cd_ibge)

    SELECT
    tb_regiao.uf,
    AVG(tb_trab.val_remuner_emprego_memb)
    FROM tb_trab
    JOIN tb_regiao ON tb_trab.id_familia = tb_regiao.id_familia
    WHERE tb_regiao.regiao = '1 - Norte'
    GROUP BY tb_regiao.uf;""",
    """SELECT
    'Não recebeu' AS salario,
    COUNT(val_remuner_emprego_memb)
    FROM tb_trab
    WHERE val_remuner_emprego_memb = 0
    UNION
    SELECT
    'Recebeu até meio salário mínimo',
    COUNT(val_remuner_emprego_memb)
    FROM tb_trab
    WHERE val_remuner_emprego_memb > 0 AND val_remuner_emprego_memb < 759
    UNION
    SELECT
    'Recebeu até um salário mínimo',
    COUNT(val_remuner_emprego_memb)
```

```
   FROM tb_trab
   WHERE val_remuner_emprego_memb > 758 AND val_remuner_emprego_memb <
1518
   UNION
   SELECT
   'Recebeu até um e meio salário mínimo',
   COUNT(val_remuner_emprego_memb)
   FROM tb_trab
   WHERE val_remuner_emprego_memb > 1517 AND val_remuner_emprego_memb <
2276
   UNION
   SELECT
   'Recebeu mais de um e meio salário mínimo',
   COUNT(val_remuner_emprego_memb)
   FROM tb_trab
   WHERE val_remuner_emprego_memb > 2275;""",
   """SELECT tb_mun.nome_municipio,
   COUNT(CASE WHEN tb_pessoa.cod_local_nascimento_pessoa = 1 AND
tb_domicilio.cod_agua_canalizada_fam = 1 THEN 1 END) AS originario,
   COUNT(CASE WHEN tb_pessoa.cod_local_nascimento_pessoa = 2 AND
tb_domicilio.cod_agua_canalizada_fam = 1 THEN 1 END) AS imigrante,
   COUNT(CASE WHEN tb_pessoa.cod_local_nascimento_pessoa = 3 AND
tb_domicilio.cod_agua_canalizada_fam = 1 THEN 1 END) AS estramgeiro
   FROM tb_mun
   JOIN tb_familia ON tb_mun.cd_ibge = tb_familia.cd_ibge
   JOIN tb_pessoa ON tb_familia.id_familia = tb_pessoa.id_familia
   JOIN tb_domicilio ON tb_pessoa.id_familia = tb_domicilio.id_familia
   GROUP BY tb_mun.nome_municipio;"""
]

# Cria indices joins
indices_joins_create = [
    "CREATE INDEX idx_fam_familia ON tb_familia (id_familia)",
    "CREATE INDEX idx_fam_cd_ibge ON tb_familia (cd_ibge)",
    "CREATE INDEX idx_dom_familia ON tb_domicilio (id_familia)",
    "CREATE INDEX idx_pes_familia ON tb_pessoa (id_familia)",
    "CREATE INDEX idx_pes_pessoa ON tb_pessoa (id_pessoa)",
    "CREATE INDEX idx_trab_familia ON tb_trab (id_familia)",
    "CREATE INDEX idx_trab_pessoa ON tb_trab (id_pessoa)",
    "CREATE INDEX idx_mun_cd_igbe ON tb_mun (cd_ibge)",
    "CREATE INDEX idx_esc_familia ON tb_esc (id_familia)",
    "CREATE INDEX idx_esc_pessoa ON tb_esc (id_pessoa)"
]
# Cria indices where
indices_where_create = [
    "CREATE INDEX idx_trab_remuner ON tb_trab
```

```
(val_remuner_emprego_memb)",
    "CREATE INDEX idx_reg_regiao ON tb_regiao (regiao)",
    "CREATE INDEX idx_pes_raca ON tb_pessoa (cod_raca_cor_pessoa)"
]
# Cria indices group by
indices_group_create = [
    "CREATE INDEX idx_mun_municipio ON tb_mun (nome_municipio)",
    "CREATE INDEX idx_mun_uf ON tb_mun (uf)",
    "CREATE INDEX idx_mun_regiao ON tb_mun (regiao)"
]
# Dropa indices joins
indices_joins_drop = [
    "DROP INDEX idx_fam_familia ON tb_familia",
    "DROP INDEX idx_fam_cd_ibge ON tb_familia",
    "DROP INDEX idx_dom_familia ON tb_domicilio",
    "DROP INDEX idx_pes_familia ON tb_pessoa",
    "DROP INDEX idx_pes_pessoa ON tb_pessoa",
    "DROP INDEX idx_trab_familia ON tb_trab",
    "DROP INDEX idx_trab_pessoa ON tb_trab",
    "DROP INDEX idx_mun_cd_igbe ON tb_mun",
    "DROP INDEX idx_esc_familia ON tb_esc",
    "DROP INDEX idx_esc_pessoa ON tb_esC"
]
# Dropa indices where
indices_where_drop = [
    "DROP INDEX idx_trab_remuner ON tb_trab",
    "DROP INDEX idx_reg_regiao ON tb_regiao",
    "DROP INDEX idx_pes_raca ON tb_pessoa"
]
# Dropa indices group by
indices_group_drop = [
    "DROP INDEX idx_mun_municipio ON tb_mun",
    "DROP INDEX idx_mun_uf ON tb_mun",
    "DROP INDEX idx_mun_regiao ON tb_mun"
]

# Sobrescrever a configuração atual com o tuning
def tuning_bd(conf_origem):
    with open(conf_origem, 'r') as origem, open(conf_destino, 'w') as
destino:
        destino.write(origem.read())

def execute_queries(queries):
    for i, query in enumerate(queries, 1):
        print(f"Executando Query {i}: {query[:50]}...")  # Mostra o
início da query para identificar
```

```
        times = []

        for _ in range(10):  # Executa 10 vezes
            start_time = time.time()
            cursor.execute(query)
            cursor.fetchall()  # Pega os resultados (evita cache)
            elapsed_time = time.time() - start_time
            times.append(elapsed_time)

        print(f"Query {i} - Tempos de execução: {times}")
        print(f"Média: {sum(times) / len(times):.4f}s | Mínimo:
{min(times):.4f}s | Máximo: {max(times):.4f}s\n")

if __name__ == "__main__":
    cursor = conn.cursor()

    # Benchmark
    execute_queries()

    # Tuning 1
    tuning_bd(tuning1)

    # Indices 1
    execute_queries(indices_joins_create)
    execute_queries(queries_consulta)
    execute_queries(indices_joins_drop)

    # Indices 2
    execute_queries(indices_where_create)
    execute_queries(queries_consulta)
    execute_queries(indices_where_drop)

    # Indices 3
    execute_queries(indices_group_create)
    execute_queries(queries_consulta)
    execute_queries(indices_group_drop)

    # Tuning 2
    tuning_bd(tuning2)

    # Indices 1
    execute_queries(indices_joins_create)
    execute_queries(queries_consulta)
    execute_queries(indices_joins_drop)

    # Indices 2
```

```
    execute_queries(indices_where_create)
    execute_queries(queries_consulta)
    execute_queries(indices_where_drop)

    # Indices 3
    execute_queries(indices_group_create)
    execute_queries(queries_consulta)
    execute_queries(indices_group_drop)

    # Tuning 3
    tuning_bd(tuning3)

    # Indices 1
    execute_queries(indices_joins_create)
    execute_queries(queries_consulta)
    execute_queries(indices_joins_drop)

    # Indices 2
    execute_queries(indices_where_create)
    execute_queries(queries_consulta)
    execute_queries(indices_where_drop)

    # Indices 3
    execute_queries(indices_group_create)
    execute_queries(queries_consulta)
    execute_queries(indices_group_drop)

    cursor.close()
    conn.close()
```