## 1. Tabela `Taxpayer`:

Esta tabela representa os contribuintes (empresas).

**Create**

```
CREATE TABLE IF NOT EXISTS public."Taxpayers"
(
    "Id" integer NOT NULL GENERATED BY DEFAULT AS IDENTITY (
INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
    "Cnpj" bigint NOT NULL,
    "CompanyName" text COLLATE pg_catalog."default" NOT NULL,
    "OpeningDate" date NOT NULL,
    "TaxationRegime" text COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "PK_Taxpayers" PRIMARY KEY ("Id")
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Taxpayers"
    OWNER to postgres;
-- Index: IX_Taxpayers_Cnpj

-- DROP INDEX IF EXISTS public."IX_Taxpayers_Cnpj";

CREATE UNIQUE INDEX IF NOT EXISTS "IX_Taxpayers_Cnpj"
    ON public."Taxpayers" USING btree
    ("Cnpj" ASC NULLS LAST)
    TABLESPACE pg_default;
```

**Select**

```
SELECT "Id", "Cnpj", "CompanyName", "OpeningDate", "TaxationRegime"
     FROM public."Taxpayers";
```

**Insert**

```
INSERT INTO public."Taxpayers"(
     "Id", "Cnpj", "CompanyName", "OpeningDate", "TaxationRegime")
     VALUES (?, ?, ?, ?, ?);
```

**Update**

```
UPDATE public."Taxpayers"
      SET "Id"=?, "Cnpj"=?, "CompanyName"=?, "OpeningDate"=?,
"TaxationRegime"=?
      WHERE <condition>;
```

**Delete**

```
DELETE FROM public."Taxpayers"
      WHERE <condition>;
```

- `Cnpj`: Identificador único da empresa.
- `CompanyName`: Nome da empresa.
- `OpeningDate`: Data de abertura da empresa.
- `TaxationRegime`: Regime tributário da empresa.

## 2. Tabela `Benefit`:

Esta tabela contém os benefícios que podem ser associados aos contribuintes.

**Create**

```
CREATE TABLE IF NOT EXISTS public."Benefits"
(
    "Id" integer NOT NULL GENERATED BY DEFAULT AS IDENTITY (
INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
    "Name" text COLLATE pg_catalog."default" NOT NULL,
    "DiscountPercentage" real NOT NULL,
    CONSTRAINT "PK_Benefits" PRIMARY KEY ("Id")
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Benefits"
    OWNER to postgres;
```

**Select**

```
SELECT "Id", "Name", "DiscountPercentage"
      FROM public."Benefits";
```

**Insert**

```
INSERT INTO public."Benefits"(
      "Id", "Name", "DiscountPercentage")
      VALUES (?, ?, ?);
```

**Update**

```
UPDATE public."Benefits"
      SET "Id"=?, "Name"=?, "DiscountPercentage"=?
      WHERE <condition>;
```

**Delete**

```
DELETE FROM public."Benefits"
      WHERE <condition>;
```

- `Name`: Nome do benefício.
- `DiscountPercentage`: Percentual de desconto associado ao benefício.

## 3. Tabela `TaxpayerBenefit`:

Essa tabela faz a associação entre contribuintes e benefícios. Cada contribuinte pode ter múltiplos benefícios, e cada benefício pode ser associado a múltiplos contribuintes.

**Create**

```
CREATE TABLE IF NOT EXISTS public."TaxpayerBenefits"
(
    "TaxpayerId" integer NOT NULL,
    "BenefitId" integer NOT NULL,
    CONSTRAINT "PK_TaxpayerBenefits" PRIMARY KEY ("TaxpayerId",
"BenefitId"),
    CONSTRAINT "FK_TaxpayerBenefits_Benefits_BenefitId" FOREIGN KEY
("BenefitId")
        REFERENCES public."Benefits" ("Id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE CASCADE,
    CONSTRAINT "FK_TaxpayerBenefits_Taxpayers_TaxpayerId" FOREIGN
KEY ("TaxpayerId")
        REFERENCES public."Taxpayers" ("Id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE CASCADE
)
```

```
TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."TaxpayerBenefits"
    OWNER to postgres;
-- Index: IX_TaxpayerBenefits_BenefitId

-- DROP INDEX IF EXISTS public."IX_TaxpayerBenefits_BenefitId";

CREATE INDEX IF NOT EXISTS "IX_TaxpayerBenefits_BenefitId"
    ON public."TaxpayerBenefits" USING btree
    ("BenefitId" ASC NULLS LAST)
    TABLESPACE pg_default;
```

**Select**

```
SELECT "TaxpayerId", "BenefitId"
     FROM public."TaxpayerBenefits";
```

**Insert**

```
INSERT INTO public."TaxpayerBenefits"(
     "TaxpayerId", "BenefitId")
     VALUES (?, ?);
```

**Update**

```
UPDATE public."TaxpayerBenefits"
     SET "TaxpayerId"=?, "BenefitId"=?
     WHERE <condition>;
```

**Delete**

```
DELETE FROM public."TaxpayerBenefits"
     WHERE <condition>;
```

- Esta tabela usa uma chave composta para associar um contribuinte a um benefício.
- `TaxpayerId` faz referência ao `Id` da tabela `Taxpayer`.
- `BenefitId` faz referência ao `Id` da tabela `Benefit`.

## 4. Tabela `Payment`:

Esta tabela representa os pagamentos realizados pelos contribuintes. Cada pagamento está associado a um benefício e a um contribuinte.

**Create**

```
CREATE TABLE IF NOT EXISTS public."Payments"
(
    "Id" integer NOT NULL GENERATED BY DEFAULT AS IDENTITY (
INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
    "InitialValue" real NOT NULL,
    "FinalValue" real NOT NULL,
    "TaxpayerId" integer NOT NULL,
    "BenefitId" integer,
    CONSTRAINT "PK_Payments" PRIMARY KEY ("Id"),
    CONSTRAINT "FK_Payments_Benefits_BenefitId" FOREIGN KEY
("BenefitId")
        REFERENCES public."Benefits" ("Id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE RESTRICT,
    CONSTRAINT "FK_Payments_Taxpayers_TaxpayerId" FOREIGN KEY
("TaxpayerId")
        REFERENCES public."Taxpayers" ("Id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE CASCADE
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Payments"
    OWNER to postgres;
-- Index: IX_Payments_BenefitId

-- DROP INDEX IF EXISTS public."IX_Payments_BenefitId";

CREATE INDEX IF NOT EXISTS "IX_Payments_BenefitId"
    ON public."Payments" USING btree
    ("BenefitId" ASC NULLS LAST)
    TABLESPACE pg_default;
-- Index: IX_Payments_TaxpayerId

-- DROP INDEX IF EXISTS public."IX_Payments_TaxpayerId";

CREATE INDEX IF NOT EXISTS "IX_Payments_TaxpayerId"
    ON public."Payments" USING btree
    ("TaxpayerId" ASC NULLS LAST)
```

```
    TABLESPACE pg_default;
```

**Select**

```
SELECT "Id", "InitialValue", "FinalValue", "TaxpayerId", "BenefitId"
     FROM public."Payments";
```

```
Insert
```

```
INSERT INTO public."Payments"(
     "Id", "InitialValue", "FinalValue", "TaxpayerId", "BenefitId")
     VALUES (?, ?, ?, ?, ?);
```

**Update**

```
UPDATE public."Payments"
     SET "Id"=?, "InitialValue"=?, "FinalValue"=?, "TaxpayerId"=?,
"BenefitId"=?
     WHERE <condition>;
```

**Delete**

```
DELETE FROM public."Payments"
     WHERE <condition>;
```

- `InitialValue`: Valor inicial do pagamento.
- `FinalValue`: Valor com desconto (calculado).
- `TaxpayerId`: Refere-se ao `Id` da tabela `Taxpayer`, associando o pagamento a um contribuinte.
- `BenefitId`: Refere-se ao `Id` da tabela `Benefit`, associando o pagamento a um benefício específico.

## Relacionamentos e Chaves Estrangeiras:

- **`Taxpayer` a `Benefit`**: Relacionamento muitos para muitos. Isso é feito pela tabela `TaxpayerBenefit`.
- **`Payment` a `Taxpayer`**: Relacionamento muitos para um. Um pagamento pertence a um único contribuinte.
- **`Payment` a `Benefit`**: Relacionamento muitos para um. Um pagamento pode estar associado a um benefício, mas não é obrigatório.

## Modelagem final:

- A tabela `Taxpayer` armazena informações dos contribuintes.
- A tabela `Benefit` armazena informações sobre os benefícios.
- A tabela `TaxpayerBenefit` faz a associação entre contribuintes e benefícios.
- A tabela `Payment` armazena os pagamentos realizados, com valores de desconto aplicados.

Com essa modelagem, você pode realizar as operações de buscar os benefícios de um contribuinte, calcular os pagamentos com desconto e associá-los corretamente no banco de dados sem precisar salvar cada pagamento antes de ser realizado.