```sql
select COUNT(*) from inep.participante;

-- 1. projetar total de participantes por tipo de escola excluindo-se os treineiros
select
    case tp_escola
        when 1 then 'Não respondeu'
        when 2 then 'Pública'
        when 3 then 'Privada'
        else 'Não informado'
    end as tipo_escola,
    COUNT(*) as quantidade
from participante
where
    not in_treineiro
group by
    tp_escola;

-- 2. Selecionar os 10 municípios com maior média de notas
with
    pontuacao_participante as (
        select p.id_participante, sum(coalesce(nu_nota, 0)) as nota_final
        from
            participante p
            inner join resposta_participante rp on (
                rp.id_participante = p.id_participante
            )
        where
            rp.tp_presenca = 1
            and not p.in_treineiro
        group by
            p.id_participante
    )
select e.no_municipio_esc, avg(pp.nota_final) as media_nota
from
    participante p
    inner join pontuacao_participante pp on (
        p.id_participante = pp.id_participante
    )
    inner join escola e on (p.id_escola = e.id_escola)
group by
    e.no_municipio_esc
order by avg(pp.nota_final) desc
limit 10;

-- 3. projetar os locais de prova com maior índice de abstenção
select lp.no_municipio_prova, lp.sg_uf_prova, count(*) as quantidade
from
    participante p
    inner join local_prova lp on (
        p.id_local_prova = lp.id_local_prova
    )
    inner join resposta_participante rp on (
        rp.id_participante = p.id_participante
        and rp.tp_prova = 'CN'
        and rp.tp_presenca = 0
    )
group by
    lp.no_municipio_prova,
    lp.sg_uf_prova
order by count(*) desc
limit 10;

-- 4. projetar as maiores médias de redação por município e tipo de escola
select
    e.no_municipio_esc as municipio,
```

```
    uf."SIGLA" as uf,
    case e.tp_dependencia_adm_esc
        when 1 then 'Federal'
        when 2 then 'Estadual'
        when 3 then 'Municipal'
        when 4 then 'Privada'
        else 'Não informado'
    end as esfera,
    round(avg(rp.nu_nota_redacao), 2) as media_redacao
from
    participante p
    inner join redacao rp on (
        rp.id_participante = p.id_participante
    )
    inner join escola e on (p.id_escola = e.id_escola)
    left join estados uf on (uf."COD" = e.co_uf_esc)
where
    not p.in_treineiro
    and rp.nu_nota_redacao is not null
group by
 uf."SIGLA",
    e.no_municipio_esc,
    e.tp_dependencia_adm_esc
order by avg(rp.nu_nota_redacao) desc
limit 10;

-- 5. projetar as maiores médias por faixa etária
with
    nota_total_participante as (
        select p.id_participante, sum(
                coalesce(rp.nu_nota, 0) + coalesce(r.nu_nota_redacao, 0)
            ) as nota_final
        from
            participante p
            left join resposta_participante rp on (
                rp.id_participante = p.id_participante
                and rp.tp_presenca = 1
            )
            left join redacao r on (
                r.id_participante = p.id_participante
            )
        where
            not p.in_treineiro
        group by
            p.id_participante
    )
select tfe.descricao, round(avg(ntp.nota_final)) as media_nota
from
    participante p
    inner join tipo_faixa_etaria tfe on (
        p.tp_faixa_etaria = tfe.cd_tipo
    )
    inner join nota_total_participante ntp on (
        p.id_participante = ntp.id_participante
    )
where
    not p.in_treineiro
group by
    tfe.descricao
order by avg(ntp.nota_final) desc
limit 10;

create MATERIALIZED view if not exists inep.estatisticas_participante as
with
    nota_total_participante as (
```

```
        select p.id_participante, sum(
                coalesce(rp.nu_nota, 0) + coalesce(r.nu_nota_redacao, 0)
            ) as nota_final
        from
            participante p
            left join resposta_participante rp on (
                rp.id_participante = p.id_participante
                and rp.tp_presenca = 1
            )
            left join redacao r on (
                r.id_participante = p.id_participante
            )
        where
            not p.in_treineiro
        group by
            p.id_participante
    )
select
    p.tp_faixa_etaria,
    count(*) as quantidade,
    round(avg(ntp.nota_final)) as media_nota,
    stddev(ntp.nota_final) as desvio_padrao,
    min(ntp.nota_final) as menor_nota,
    max(ntp.nota_final) as maior_nota,
    percentile_cont(0.25) within group (order by ntp.nota_final) as primeiro_quartil,
    percentile_cont(0.5) within group (order by ntp.nota_final) as mediana,
    percentile_cont(0.75) within group (order by ntp.nota_final) as terceiro_quartil
from
    participante p
    inner join nota_total_participante ntp on (
        p.id_participante = ntp.id_participante
    )
group by
    p.tp_faixa_etaria;

select
 tfe.descricao,
 ep.quantidade,
 ep.media_nota as media,
 round(ep.desvio_padrao) as desvio_padrao,
 round(ep.menor_nota) as menor_nota,
 round(ep.maior_nota) as maior_nota,
 round(ep.primeiro_quartil ) as primeiro_quartil,
 round(ep.mediana) as mediana,
 round(ep.terceiro_quartil ) as terceiro_quartil
from
 estatisticas_participante ep
inner join
 tipo_faixa_etaria tfe on (ep.tp_faixa_etaria = tfe.cd_tipo );

-- -- função para cálculo do total da nota
create or replace function calcular_nota_total(p_nu_inscricao bigint)
returns numeric as $$
declare
    l_nota_total numeric;
begin
    select sum(coalesce(rp.nu_nota, 0) + coalesce(r.nu_nota_redacao, 0))
    into l_nota_total
    from
        participante p
        left join resposta_participante rp on (
            rp.id_participante = p.id_participante
            and rp.tp_presenca = 1
        )
        left join redacao r on (
```

```
            r.id_participante = p.id_participante
        )
    where
        p.nu_inscricao = p_nu_inscricao;

    return l_nota_total;
end;
$$ language plpgsql;

-- função para cálculo do z-score
create or replace function comparar_participantes(
    p_nu_inscrição_a bigint,
    p_nu_inscrição_b bigint
) returns numeric as $$
declare
    l_participante_a participante%rowtype;
    l_participante_b participante%rowtype;
    l_nota_total_a numeric;
    l_nota_total_b numeric;
    l_stats_a estatisticas_participante%rowtype;
    l_stats_b estatisticas_participante%rowtype;
    l_media_b numeric;
    l_z_a numeric;
    l_z_b numeric;
begin
    select * into l_participante_a
    from participante
    where nu_inscricao = p_nu_inscrição_a;

    select * into l_participante_b
    from participante
    where nu_inscricao = p_nu_inscrição_b;

    if l_participante_a.tp_faixa_etaria = l_participante_b.tp_faixa_etaria then
        return calcular_nota_total(l_participante_a.nu_inscricao) -
    else
        select * into l_stats_a from estatisticas_participante where tp_faixa_etaria =
        select * into l_stats_b from estatisticas_participante where tp_faixa_etaria =

        l_z_a := (calcular_nota_total(l_participante_a.nu_inscricao) - l_stats_a.media_nota) /
        l_z_b := (calcular_nota_total(l_participante_b.nu_inscricao) - l_stats_b.media_nota) /
        return l_z_a - l_z_b;
    end if;
end;
$$ language plpgsql;

select
    calcular_nota_total(210060925335) nota_a,
    calcular_nota_total(210060978194) nota_b,
    comparar_participantes(210060925335, 210060978194);

select tfe.descricao, p.* from participante p inner join tipo_faixa_etaria tfe on (tfe.cd_tipo
= p.tp_faixa_etaria) where p.nu_inscricao in (210060925335, 210060978194);
```