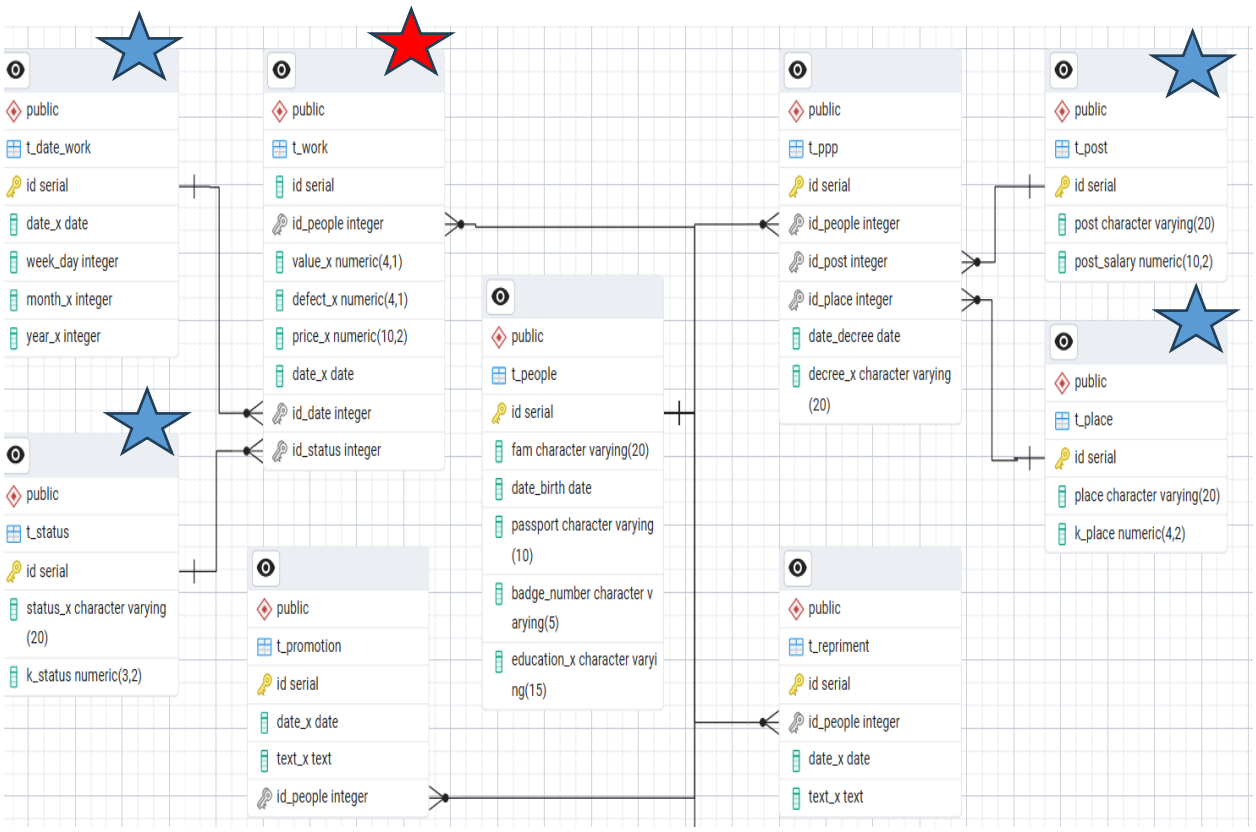

1. Выполнить проектирование хранимой процедуру (ХП), обеспечивающих начисление заработной платы (ЗП) каждому работнику в указанный месяц заданного года.

Формулы начисления ЗП

$$ЗП_{смена} = \frac{оклад * k_{status} * k_{place}}{количество\ смен\ в\ месяц} -$$

для начальников и тех рабочих, которые отсутствовали на рабочем месте по уважительной причине.

Для рабочих, которые трудились за станком

$$ЗП_{смена} = \frac{стоимость\ одного\ изделия(объем - брак)}{8}$$

Логика работы ХП должна предусматривать:

— организацию корректного выхода из процедуры, предусматривающую освобождение памяти от временной таблицы;

— после обработки всех рабочих смен выполнение в процедуре операции переноса данных в таблицу *t_salary*.

```
CREATE OR REPLACE PROCEDURE sp_calculate_salary (
    p_month INT,  -- Месяц (1-12)
    p_year INT    -- Год (например, 2025)
)
LANGUAGE plpgsql
AS $$
DECLARE
    v_shift_count INT;  -- Количество рабочих дней в месяце
BEGIN
    -- 1. Подсчет рабочих дней в месяце
    SELECT COUNT(*) INTO v_shift_count
    FROM t_date_work
    WHERE date_part('month', date_x) = p_month
      AND date_part('year', date_x) = p_year;

    -- 2. Создание временной таблицы для расчетов по всем цехам
    DROP TABLE IF EXISTS t_temp_salary_detail;
    CREATE TEMP TABLE t_temp_salary_detail AS
    SELECT
        p.id AS id_people,
        p.fam,
        p.passport,
        po.post,
        po.post_salary,
        pl.k_place,
        st.id AS status_id,
        st.k_status,
        w.value_x,
        w.defect_x,
        w.price_x,
        d.date_x
    FROM t_people p
    JOIN t_ppp pp ON p.id = pp.id_people
    JOIN t_post po ON pp.id_post = po.id
    JOIN t_place pl ON pp.id_place = pl.id
    JOIN t_work w ON p.id = w.id_people
    JOIN t_status st ON w.id_status = st.id
    JOIN t_date_work d ON w.id_date = d.id
    WHERE
        pp.date_decree = (
```

```
            SELECT MAX(px.date_decree)
            FROM t_ppp px
            WHERE px.id_people = p.id AND px.date_decree <= d.date_x
        )
        AND date_part('month', d.date_x) = p_month
        AND date_part('year', d.date_x) = p_year;

    -- 3. Агрегация данных по сотрудникам
    DROP TABLE IF EXISTS t_temp_salary;
    CREATE TEMP TABLE t_temp_salary AS
    SELECT
        id_people,
        fam,
        passport,
        -- Берем последний оклад
        (SELECT post_salary FROM t_temp_salary_detail d
         WHERE d.id_people = t.id_people
         ORDER BY date_x DESC LIMIT 1) AS post_salary,
        -- Берем последний коэффициент цеха
        (SELECT k_place FROM t_temp_salary_detail d
         WHERE d.id_people = t.id_people
         ORDER BY date_x DESC LIMIT 1) AS k_place,
        -- Суммируем дни по всем цехам
        SUM(CASE WHEN status_id = 2 THEN 1 ELSE 0 END) AS
day_bus_trip,
        SUM(CASE WHEN status_id = 3 THEN 1 ELSE 0 END) AS
day_disease,
        SUM(CASE WHEN status_id = 4 THEN 1 ELSE 0 END) AS
day_vacation,
        COUNT(status_id) AS day_all,
        -- Расчет зарплаты по компонентам
        ROUND(SUM(
            CASE
                WHEN status_id = 2 THEN (post_salary * k_status *
k_place) / v_shift_count
                ELSE 0
            END
        ), 2) AS salary_bt,
        ROUND(SUM(
            CASE
                WHEN status_id = 3 THEN (post_salary * k_status *
k_place) / v_shift_count
                ELSE 0
            END
        ), 2) AS salary_d,
```

```
        ROUND(SUM(
            CASE
                WHEN status_id = 4 THEN (post_salary * k_status *
k_place) / v_shift_count
                ELSE 0
            END
        ), 2) AS salary_v,
        -- Общая зарплата
        ROUND(SUM(
            CASE
                WHEN post = 'рабочий' AND status_id > 1 THEN
                    (post_salary * k_status * k_place) / v_shift_count
                WHEN post = 'рабочий' AND status_id = 1 THEN
                    (value_x - defect_x) * price_x
                WHEN post IN ('начальник','заместитель','бригадир')
THEN
                    (post_salary * k_status * k_place) / v_shift_count
                ELSE 0
            END
        ), 2) AS salary_all
    FROM t_temp_salary_detail t
    GROUP BY id_people, fam, passport;

    -- 4. Сохранение результатов в основную таблицу
    IF NOT EXISTS (SELECT * FROM information_schema.tables
WHERE table_name = 't_salary') THEN
        CREATE TABLE t_salary (
            fam character varying(20),
            passport character varying(10),
            month_x double precision,
            year_x double precision,
            day_bus_trip bigint,
            day_disease bigint,
            day_vacation bigint,
            day_all bigint,
            salary_bt numeric,
            salary_d numeric,
            salary_v numeric,
            salary_all numeric
        );
    END IF;-- Удаляем старые данные за этот месяц
    DELETE FROM t_salary
    WHERE month_x = p_month AND year_x = p_year;

    -- Вставляем новые данные
```

```
    INSERT INTO t_salary (
        fam, passport, month_x, year_x,
        day_bus_trip, day_disease, day_vacation, day_all,
        salary_bt, salary_d, salary_v, salary_all
    )
    SELECT
        fam, passport, p_month::double precision, p_year::double
precision,
        day_bus_trip, day_disease, day_vacation, day_all,
        salary_bt, salary_d, salary_v, salary_all
    FROM t_temp_salary;

    RAISE NOTICE 'Зарплата за %-% успешно рассчитана',
p_month, p_year;
END;
$$;


--Рассчитываем ЗП с января по май 2025 года.
CALL sp_calculate_salary(1, 2025);
CALL sp_calculate_salary(2, 2025);
CALL sp_calculate_salary(3, 2025);
CALL sp_calculate_salary(4, 2025);
CALL sp_calculate_salary(5, 2025);

--Выводим данные. Группируем по месяцу.

SELECT *
        FROM public.t_salary
        order by month_x;
```

	fam character varying (20)	passport character varying (10)	month_x double precision	year_x double precision	day_bus_trip bigint	day_disease bigint	day_vacation bigint	day_all bigint	salary_bt numeric	salary_d numeric	salary_v numeric	salary_all numeric
1	Алехин	4444411111	1	2025	0	0	0	23	0.00	0.00	0.00	58170.26
2	Деев	4444411112	1	2025	0	0	0	23	0.00	0.00	0.00	47775.69
3	Лосев	4444411113	1	2025	4	0	0	23	7976.26	0.00	0.00	41804.13
4	Фролов	4444411114	1	2025	0	5	0	23	0.00	2977.47	0.00	317534.39
5	Щукин	4444411115	1	2025	3	0	0	23	4908.66	0.00	0.00	332915.82
6	Гришин	7777711111	1	2025	0	4	0	23	0.00	2337.46	0.00	311421.13
7	Волков	7777711112	1	2025	0	0	0	23	0.00	0.00	0.00	382675.02
8	Воробьев	7777711113	1	2025	0	0	0	23	0.00	0.00	0.00	381273.28
9	Мухин	7777711114	1	2025	0	0	0	23	0.00	0.00	0.00	374965.45
10	Шишкин	7777711115	1	2025	0	0	0	23	0.00	0.00	0.00	376367.19
11	Иванов	1234567899	1	2025	0	0	0	23	0.00	0.00	0.00	63710.29
12	Сидоров	1234567891	1	2025	0	0	0	23	0.00	0.00	0.00	52325.76
13	Петров	1234567892	1	2025	4	0	0	23	8735.90	0.00	0.00	45785.47
14	Федоров	1234567894	1	2025	0	5	0	23	0.00	3144.44	0.00	343356.23
15	Шишкина	5202711115	1	2025	3	0	0	23	5376.15	0.00	0.00	402381.18
16	Иванова	2234567899	1	2025	0	4	0	23	0.00	2560.07	0.00	371175.41
17	Сидорова	2234567891	1	2025	0	0	0	23	0.00	0.00	0.00	447024.96
18	Петрова	2234567892	1	2025	0	0	0	23	0.00	0.00	0.00	452883.15
19	Федорова	2234567894	1	2025	0	0	0	23	0.00	0.00	0.00	453784.41
20	Щукина	5704411115	1	2025	0	0	0	23	0.00	0.00	0.00	447926.22
21	Алехин	4444411111	2	2025	0	0	0	20	0.00	0.00	0.00	58170.26
22	Деев	4444411112	2	2025	0	0	0	20	0.00	0.00	0.00	47775.69
23	Лосев	4444411113	2	2025	3	0	0	20	6879.52	0.00	0.00	41686.62
24	Фролов	4444411114	2	2025	0	4	0	20	0.00	2688.07	0.00	266916.06
25	Щукин	4444411115	2	2025	3	0	0	20	6182.57	0.00	0.00	312676.02
26	Гришин	7777711111	2	2025	0	4	0	20	0.00	2688.07	0.00	278129.98
27	Волков	7777711112	2	2025	0	0	0	20	0.00	0.00	0.00	330810.64
28	Воробьев	7777711113	2	2025	0	0	0	20	0.00	0.00	0.00	329408.90
29	Мухин	7777711114	2	2025	0	0	0	20	0.00	0.00	0.00	328007.16
30	Шишкин	7777711115	2	2025	0	0	0	20	0.00	0.00	0.00	318895.85

Total rows: 100 | Query complete 00:00:00.091

2. Выполнить проектирование ХП, обеспечивающей вывод результатов работников, которые отсутствовали на рабочем месте в виде:

```
CREATE OR REPLACE PROCEDURE sp_employee_absences_report(
   m_x INT,
   y_x INT
)
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
   cnt_shift INT := 1;
BEGIN
   -- Расчет количества рабочих дней в месяце
   SELECT COUNT(date_x)
   INTO cnt_shift
   FROM t_date_work
   WHERE date_part('month', date_x) = m_x AND
        date_part('year', date_x) = y_x;

   DROP TABLE IF EXISTS t_temp;

   -- Создание временной таблицы с данными о работе
   CREATE TEMP TABLE t_temp ON COMMIT DROP AS
   SELECT
      d.date_x,
      w.id_people,
      w.id_status,
      px.id_post,
```

```
        px.id_place,
        w.value_x,
        w.defect_x,
        w.price_x,
        ROUND(po.post_salary * pl.k_place * st.k_status / cnt_shift, 2) AS
day_salary,
        date_part('month', d.date_x) AS month_x,
        date_part('year', d.date_x) AS year_x,
        pl.place AS workshop
    FROM
        t_date_work AS d
        JOIN t_work AS w ON d.id = w.id_date
        JOIN t_status AS st ON st.id = w.id_status
        JOIN t_ppp AS px ON w.id_people = px.id_people
        JOIN t_post AS po ON px.id_post = po.id
        JOIN t_place AS pl ON px.id_place = pl.id
    WHERE
        px.date_decree = (
            SELECT MAX(q.date_decree)
            FROM t_ppp AS q
            WHERE px.id_people = q.id_people AND q.date_decree <= d.date_x
        ) AND
        date_part('month', d.date_x) = m_x AND
        date_part('year', d.date_x) = y_x;

    -- Создание или обновление таблицы отчетности
    IF NOT EXISTS (
        SELECT *
        FROM information_schema.tables
        WHERE table_name = 't_employee_absences'
```

```
        AND table_schema = 'public'
    ) THEN
        -- Создание таблицы отчетности, если она не существует
        CREATE TABLE t_employee_absences AS
        SELECT
            t.workshop,
            t.month_x,
            t.year_x AS year_x,
            COUNT(CASE WHEN t.id_status = 2 THEN 1 END) AS
days_business_trip,
            COUNT(CASE WHEN t.id_status = 3 THEN 1 END) AS days_sick_leave,
            COUNT(CASE WHEN t.id_status = 4 THEN 1 END) AS days_vacation,
            COUNT(CASE WHEN t.id_status > 1 THEN 1 END) AS
total_days_absence,
            COALESCE(SUM(CASE WHEN t.id_status = 2 THEN t.day_salary
END), 0) AS salary_business_trip,
            COALESCE(SUM(CASE WHEN t.id_status = 3 THEN t.day_salary
END), 0) AS salary_sick_leave,
            COALESCE(SUM(CASE WHEN t.id_status = 4 THEN t.day_salary
END), 0) AS salary_vacation,
            COALESCE(SUM(CASE WHEN t.id_status > 1 THEN t.day_salary
END), 0) AS total_salary_absences
        FROM
            t_temp t
        GROUP BY
            t.workshop, t.month_x, t.year_x;
    ELSE
        -- Проверка, существуют ли уже данные за указанный период
        IF NOT EXISTS (
            SELECT *
            FROM t_employee_absences
            WHERE month_x = m_x AND year_x = y_x
```

```
    ) THEN
        -- Вставка новых данных
        INSERT INTO t_employee_absences (
            workshop,
            month_x,
            year_x,
            days_business_trip,
            days_sick_leave,
            days_vacation,
            total_days_absence,
            salary_business_trip,
            salary_sick_leave,
            salary_vacation,
            total_salary_absences
        )
        SELECT
            t.workshop,
            t.month_x,
            t.year_x,
            COUNT(CASE WHEN t.id_status = 2 THEN 1 END) AS
days_business_trip,
            COUNT(CASE WHEN t.id_status = 3 THEN 1 END) AS
days_sick_leave,
            COUNT(CASE WHEN t.id_status = 4 THEN 1 END) AS
days_vacation,
            COUNT(CASE WHEN t.id_status > 1 THEN 1 END) AS
total_days_absence,
            COALESCE(SUM(CASE WHEN t.id_status = 2 THEN t.day_salary
END), 0) AS salary_business_trip,COALESCE(SUM(CASE WHEN t.id_status =
3 THEN t.day_salary END), 0) AS salary_sick_leave,
            COALESCE(SUM(CASE WHEN t.id_status = 4 THEN t.day_salary
END), 0) AS salary_vacation,
```

```
            COALESCE(SUM(CASE WHEN t.id_status > 1 THEN t.day_salary
END), 0) AS total_salary_absences
          FROM
           t_temp t
          GROUP BY
             t.workshop, t.month_x, t.year_x;
       ELSE
          RAISE NOTICE 'Данные за указанный период уже существуют!';
       END IF;
    END IF;

    -- Вывод результатов
    RAISE NOTICE 'Отчет по отсутствиям сотрудников за %/%:', m_x, y_x;
END;
$BODY$;
--Расчет и добавление данных
CALL sp_employee_absences_report(1, 2025);
CALL sp_employee_absences_report(2, 2025);
CALL sp_employee_absences_report(3, 2025);
CALL sp_employee_absences_report(4, 2025);
CALL sp_employee_absences_report(5, 2025);
--Просмотр данных
SELECT *
        FROM public.t_employee_absences;
```

Showing rows: 1 to 10 Page No: 1 of 1

	workshop character varying (20)	month_x double precision	year_x double precision	days_business_trip bigint	days_sick_leave bigint	days_vacation bigint	total_days_absence bigint	salary_business_trip numeric	salary_sick_leave numeric	salary_vacation numeric	total_salary_absences numeric
1	цех №71	1	2025	7	9	0	16	12884.90	5259.24	0	18144.14
2	цех №72	1	2025	7	9	0	16	14112.03	5760.18	0	19872.21
3	цех №72	2	2025	6	8	0	14	13717.29	5888.16	0	19605.45
4	цех №71	2	2025	6	8	0	14	12524.46	5376.16	0	17900.62
5	цех №72	3	2025	7	8	0	15	15026.79	5607.76	0	20634.55
6	цех №71	3	2025	7	8	0	15	13720.11	5120.16	0	18840.27
7	цех №71	4	2025	7	8	0	15	13470.57	4887.44	0	18358.01
8	цех №72	4	2025	7	8	0	15	14753.53	5352.88	0	20106.41
9	цех №71	5	2025	8	9	0	17	14807.05	5498.37	0	20305.42
10	цех №72	5	2025	8	9	0	17	16217.30	6021.99	0	22239.29

Total rows: 10 | Query complete 00:00:00.091 | CRLF | Ln 2, Col 34