ЛАБОРАТОРНАЯ РАБОТА №8

Вариант №4

Началась лабораторная работа №8 с небольшого обновления в БД. В таблице «Contract_Templates» были добавлены два новых поля:

1) Type of percentage – тип процентной ставки, отображающий вид кредита – аннуитетный или дифференцированный.
2) Penalty percentage – количество штрафных процентов за просрочку выплат.

Изменения можно увидеть на схеме рис. 1.

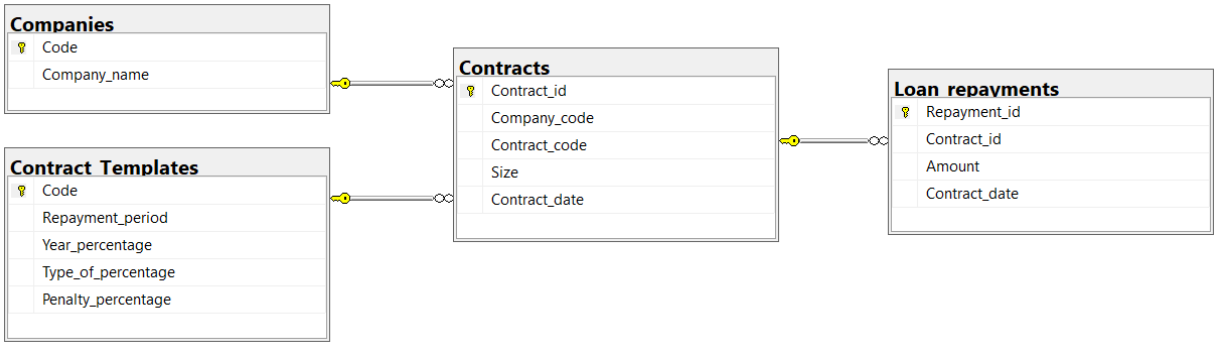

Рисунок 1. Физическая модель базы данных

В SQL изменения выглядят следующим образом:

```
-- Добавление поля для таблицы "Contract_Templates"
ALTER TABLE Contract_Templates
ADD Type_of_percentage NCHAR(15) NOT NULL;

-- Добавление поля для таблицы "Contract_Templates"
ALTER TABLE Contract_Templates
ADD Penalty_percentage DECIMAL(5,2) NOT NULL;
```

Код запроса на обновление таблицы «Contract_Templates»

Начнём реализацию необходимых выходных документов с создания нескольких простых агрегатных функций:

```
-- Создание функции для нахождения количества месяцев, прошедших с заключения договора
CREATE FUNCTION CalculateMonthsPassed
(
    @contractDate DATE,
    @currentDate DATE
)
RETURNS INT
AS
BEGIN
    DECLARE @monthsPassed INT;

    SET @monthsPassed = DATEDIFF(MONTH, @contractDate, @currentDate);

    RETURN @monthsPassed;
END;
```

```sql
-- Создание функции на поиск ближайшей даты обновления процентов по кредиту
CREATE FUNCTION GetNearestContractUpdateDate
(
    @Contract_Date DATE,
    @Current_Date DATE
)
RETURNS DATE
AS
BEGIN
    DECLARE @NextUpdateDate DATE;

    IF @Current_Date > @Contract_Date
    BEGIN
        SET @NextUpdateDate = @Contract_Date;

        WHILE @NextUpdateDate < @Current_Date
        BEGIN
            SET @NextUpdateDate = DATEADD(MONTH, 1, @NextUpdateDate);
        END;

        RETURN DATEADD(MONTH, -1, @NextUpdateDate);
    END
    ELSE
    BEGIN
        RETURN @Contract_Date;
    END;
       RETURN @Contract_Date;
END;

-- Создание функции для высчета суммы всех платежей по кредиту на дату
CREATE FUNCTION CalculateTotalPayments
(
    @contractId BIGINT,
    @date DATE
)
RETURNS DECIMAL(18, 2)
AS
BEGIN
    DECLARE @totalAmount DECIMAL(18, 2);

    -- Находим сумму всех выплат по контракту на заданную дату
    SELECT @totalAmount = SUM(Amount)
    FROM Loan_repayments
    WHERE Contract_id = @contractId
    AND Contract_date <= @date;

    -- Если нет выплат по контракту, возвращаем 0
    IF @totalAmount IS NULL
    BEGIN
        SET @totalAmount = 0;
    END

    RETURN @totalAmount;
END;
```

Код запросов на создание основных агрегатных функций

Отлично, теперь у нас есть все необходимые инструменты, чтобы написать ещё одну функцию для масштабирования итоговой суммы кредита по процентам на определённую дату с учётом:

1) Типа процентной ставки;

2) Возможных просрочек по выплатам;
3) Возможного истечения срока кредита (в таком случае начисляется только ***ШТРАФ***, без начисления процентов по основному кредиту)

Код на языке SQL этой функции будет выглядеть следующим образом:

```
-- Создание функции для масштабирования суммы кредита по процентам
CREATE FUNCTION ScaleLoanAmount
(
    @currentDate DATE,
    @contractId BIGINT
)
RETURNS DECIMAL(18,2)
AS
BEGIN
    DECLARE @scaledAmount DECIMAL(18,2);
    DECLARE @typeOfPercentage NCHAR(15);
       DECLARE @repaymentPeriod INT;
       DECLARE @contractDate DATE;
    DECLARE @monthPercentage DECIMAL(5,2);
       DECLARE @contractPenalty DECIMAL(5,2);

    -- Получаем информацию о контракте
    SELECT
       @typeOfPercentage = Ct.Type_of_percentage,
       @monthPercentage = Ct.Year_percentage / 12,
       @repaymentPeriod = Ct.Repayment_period,
       @contractDate = Co.Contract_date,
       @contractPenalty = Ct.Penalty_percentage
    FROM Contracts Co
    JOIN Contract_Templates Ct ON Co.Contract_code = Ct.Code
    WHERE Co.Contract_id = @contractId;

    -- Вычисляем количество прошедших месяцев
    DECLARE @monthsPassed INT;
       SET @monthsPassed = dbo.CalculateMonthsPassed(@contractDate, @currentDate);

       IF @monthsPassed < 0 OR @contractDate > @currentDate
       BEGIN
              RETURN 0;
       END
       IF @monthsPassed = 0 AND @contractDate <= @currentDate
       BEGIN
              RETURN (SELECT Size FROM Contracts WHERE Contract_id = @contractId);
       END
       IF @monthsPassed > @repaymentPeriod
       BEGIN
              DECLARE @nowDATE DATE = DATEADD(MONTH, @repaymentPeriod, @contractDate);
              SET @scaledAmount = dbo.ScaleLoanAmount(@nowDATE, @contractId);
              WHILE @scaledAmount >= dbo.CalculateTotalPayments(@contractId, @nowDATE)
AND @nowDATE <= @currentDate
        BEGIN
            SET @nowDATE = DATEADD(MONTH, 1, @nowDATE);
                    SET @scaledAmount = @scaledAmount * (1 + @contractPenalty / 100);
        END;

              RETURN @scaledAmount;
       END

       DECLARE @totalInterest DECIMAL(18,2);
       DECLARE @nearestDate DATE = dbo.GetNearestContractUpdateDate(@contractDate,
@currentDate);
    SET @totalInterest = dbo.ScaleLoanAmount(@nearestDate, @contractId);
```

```
        IF MONTH(@nearestDate) = MONTH(@currentDate)
                        AND YEAR(@nearestDate) = YEAR(@currentDate)
        BEGIN
                RETURN @totalInterest
        END

    -- Масштабируем сумму кредита в зависимости от типа процентной ставки
    IF @typeOfPercentage = N'fixed' -- фиксированная ставка
    BEGIN
        SET @scaledAmount = @totalInterest + (SELECT Size FROM Contracts WHERE
Contract_id = @contractId) * (@monthPercentage / 100);
    END
    ELSE -- дифференцированная ставка
    BEGIN
        SET @scaledAmount = @totalInterest * (1 + @monthPercentage / 100);
    END

        -- Вычисляем штраф
    SET @totalInterest = dbo.CalculateTotalPayments(@contractId, @currentDate);

        IF @totalInterest < @scaledAmount / @repaymentPeriod * @monthsPassed
        BEGIN
                SET @scaledAmount = @scaledAmount * (1 + @contractPenalty / 100);
        END

    RETURN @scaledAmount;
END;
```

Код запросов на создание агрегатной функции масштабирования итоговой суммы кредита по процентам на определённую дату

Для проверки функции заполним базу данных сведениями о видах кредитов, компаниях и договорах:

```
INSERT INTO Companies (code, Company_name)
VALUES
(2, 'Alpha Industries'),
(3, 'Beta Corporation'),
(5, 'Gamma Enterprises'),
(7, 'Delta Innovations'),
(11, 'Epsilon Technologies'),
(13, 'Zeta Solutions'),
(17, 'Eta Group'),
(19, 'Theta Systems'),
(23, 'Iota Ventures'),
(29, 'Kappa Enterprises');

INSERT INTO Contract_Templates (Code, Repayment_period, Year_percentage,
Type_of_percentage, Penalty_percentage)
VALUES
(1, 12, 3.50, 'fixed', 1.50),
(2, 24, 4.25, 'fixed', 2.00),
(3, 36, 5.00, 'fixed', 2.50),
(4, 12, 3.75, 'differentiated', 1.75),
(5, 24, 4.50, 'differentiated', 2.25),
(6, 36, 5.25, 'differentiated', 2.75),
(7, 48, 5.50, 'fixed', 3.00);


INSERT INTO Contracts (Contract_id, Company_Code, Contract_Code, Size, Contract_Date)
VALUES
(1, 2, 1, 100000.00, '2024-04-01'),
```

```
(2, 3, 2, 150000.00, '2023-02-02'),
(3, 5, 3, 200000.00, '2023-01-03'),
(4, 7, 4, 250000.00, '2023-11-04'),
(5, 11, 5, 300000.00, '2023-10-05'),
(6, 13, 6, 350000.00, '2022-09-06'),
(7, 17, 7, 400000.00, '2024-05-07'),
(8, 19, 1, 450000.00, '2023-06-08'),
(9, 23, 2, 500000.00, '2023-06-09'),
(10, 29, 3, 550000.00, '2022-08-10'),
(11, 2, 4, 600000.00, '2022-12-11'),
(12, 3, 5, 650000.00, '2023-11-12'),
(13, 5, 6, 700000.00, '2022-08-13'),
(14, 7, 7, 750000.00, '2023-07-14'),
(15, 11, 1, 800000.00, '2023-05-15'),
(16, 13, 2, 850000.00, '2022-08-16'),
(17, 17, 3, 900000.00, '2023-09-17'),
(18, 19, 4, 950000.00, '2023-10-18'),
(19, 23, 5, 1000000.00, '2023-11-19'),
(20, 29, 6, 1050000.00, '2023-08-20'),
(21, 2, 7, 1100000.00, '2024-02-21'),
(22, 3, 1, 1150000.00, '2024-01-22'),
(23, 5, 2, 1200000.00, '2023-02-23'),
(24, 7, 3, 1250000.00, '2024-03-24'),
(25, 11, 4, 1300000.00, '2023-04-25'),
(26, 13, 5, 1350000.00, '2024-04-26'),
(27, 17, 6, 1400000.00, '2022-06-27'),
(28, 19, 7, 1450000.00, '2023-12-28'),
(29, 23, 1, 1500000.00, '2023-04-29'),
(30, 29, 2, 1550000.00, '2023-06-30');
```

Код запроса на вставку данных

Теперь напишем запрос на проверку функции (для наглядности будем смотреть на результаты по месяцам):

```
PRINT N'Задолженность по кредиту №1 (выплаты пустые):';
PRINT dbo.ScaleLoanAmount(GETDATE(), 1);

PRINT N'Задолженность по кредиту №3 (выплаты пустые):';
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -17, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -16, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -15, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -14, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -13, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -12, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -11, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -10, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -9, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -8, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -7, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -6, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -5, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -4, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -3, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -2, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -1, GETDATE()), 3);
PRINT dbo.ScaleLoanAmount(GETDATE(), 3);

PRINT N'Задолженность по кредиту №11 (выплаты пустые):';
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -17, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -16, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -15, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -14, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -13, GETDATE()), 11);
```

```
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -12, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -11, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -10, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -9, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -8, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -7, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -6, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -5, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -4, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -3, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -2, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(DATEADD(MONTH, -1, GETDATE()), 11);
PRINT dbo.ScaleLoanAmount(GETDATE(), 11);
```

Код запроса-проверки функции

Видим следующий результат:

```
Задолженность по кредиту №1 (выплаты пустые):
101794.35
Задолженность по кредиту №3 (выплаты пустые):
0.00
200000.00
205861.00
211868.53
218026.24
224337.90
230807.35
237438.53
244235.49
251202.38
258343.44
265663.03
273165.61
280855.75
288738.14
296817.59
305099.03
313587.51
Задолженность по кредиту №11 (выплаты пустые):
600000.00
612392.55
625041.06
637950.82
651127.22
664575.76
678302.07
692311.89
706611.08
721205.60
736101.56
751305.18
766822.83
793896.47
807789.66
821925.98
836309.68
850945.10
```

Результат запроса-проверки функции

Как видно из теста кредита №1, функция увеличила изначальный размер кредита на 3,5 / 12 = 0,29% и затем начислила штраф в размере 1,5%. В тесте

кредита №3 можно увидеть, что если запрашивается информация по кредиту на дату, когда он ещё не был оформлен, то сумма кредита будет составлять 0, а если в месяц его заключения – его размер. В тесте кредита №11 ясно видно, что в последние 5 месяцев срок кредита уже истёк, и должны начисляться только штрафные проценты, что и видно из выражения:

$$836309.68 * (1 + 1{,}75 / 12 / 100) \approx 850945.10$$

Заполним данными таблицу «Loan_repayments»:

```
INSERT INTO Loan_repayments (Repayment_id, Contract_id, Contract_date, Amount)
VALUES
(1, 11, '10-01-2023', 50156.25),
(2, 11, '10-02-2023', 50469.73),
(3, 11, '10-03-2023', 50784.66),
(4, 11, '10-04-2023', 51000.00),
(5, 11, '10-05-2023', 51200.33),
(6, 11, '10-06-2023', 51784.66),
(7, 11, '10-07-2023', 52324.54),
(8, 11, '10-08-2023', 52784.78),
(9, 11, '10-09-2023', 53000.26),
(10, 11, '10-10-2023', 53784.36),
(11, 11, '10-11-2023', 54000.51),
(12, 11, '10-12-2023', 54321.67),
(13, 11, '11-12-2023', 543210.67);

INSERT INTO Loan_repayments (Repayment_id, Contract_id, Amount, Contract_Date)
VALUES
(151, 1, 250000.00, '2023-05-15'),
(211, 1, 200000.00, '2023-05-16'),
(31, 1, 180000.00, '2023-05-17'),
(41, 1, 220000.00, '2023-05-18'),
(52, 2, 300000.00, '2023-05-19'),
(61, 2, 270000.00, '2023-05-20'),
(71, 2, 250000.00, '2023-05-21'),
(81, 2, 280000.00, '2023-05-22'),
(91, 30, 320000.00, '2023-05-23'),
(101, 30, 300000.00, '2023-05-24'),
(111, 30, 270000.00, '2023-05-25'),
(121, 30, 230000.00, '2023-05-26'),
(131, 30, 240000.00, '2023-05-27'),
(14, 2, 260000.00, '2023-05-28'),
(15, 2, 290000.00, '2023-05-29'),
(16, 5, 310000.00, '2023-05-30'),
(17, 7, 320000.00, '2023-05-31'),
(18, 8, 330000.00, '2023-06-01'),
(19, 8, 280000.00, '2023-06-02'),
(20, 8, 270000.00, '2023-06-03'),
(21, 8, 240000.00, '2023-06-04'),
(22, 9, 250000.00, '2023-06-05'),
(23, 9, 260000.00, '2023-06-06'),
(24, 9, 280000.00, '2023-06-07'),
(25, 10, 290000.00, '2023-06-08'),
(26, 10, 310000.00, '2023-06-09'),
(27, 10, 330000.00, '2023-06-10'),
(28, 14, 340000.00, '2023-06-11'),
(29, 14, 350000.00, '2023-06-12'),
(30, 15, 360000.00, '2023-06-13');
```

Код запроса заполнения таблицы выплат

Теперь можно переходить к выходным формам. Форму №1 получаем запросом:

```
DECLARE @currentDate DATE = GETDATE();
SELECT
       C.Company_name,
       COUNT(Co.Contract_id) AS Count_Of_Credits,
       SUM(CASE WHEN dbo.ScaleLoanAmount(@currentDate, Co.Contract_id) <=
dbo.CalculateTotalPayments(Co.Contract_id, @currentDate)
                     THEN 1
                     ELSE 0
                     END) AS Count_Of_Paid_Credits,
       SUM(CASE WHEN dbo.ScaleLoanAmount(@currentDate, Co.Contract_id) <=
dbo.CalculateTotalPayments(Co.Contract_id, @currentDate)
                     THEN 0
                     ELSE 1
                     END) AS Count_Of_Non_Paid_Credits,
       SUM(CASE WHEN dbo.ScaleLoanAmount(@currentDate, Co.Contract_id) <=
dbo.CalculateTotalPayments(Co.Contract_id, @currentDate)
                     THEN 0
                     ELSE dbo.ScaleLoanAmount(@currentDate, Co.Contract_id) -
dbo.CalculateTotalPayments(Co.Contract_id, @currentDate)
                     END) AS Debt
FROM Companies C
JOIN Contracts Co ON C.Code = Co.Company_code
GROUP BY ROLLUP(C.Company_name);
```

Код запроса получения формы №1

Соответственно результат:

	Company_name	Count_Of_Credits	Count_Of_Paid_Credits	Count_Of_Non_Paid_Credits	Debt
1	Alpha Industries	3	2	1	1218108.85
2	Beta Corporation	3	1	2	1994350.82
3	Delta Innovations	3	0	3	1701304.13
4	Epsilon Technologies	3	1	2	2248519.12
5	Eta Group	3	0	3	4100853.14
6	Gamma Enterprises	3	0	3	3359643.99
7	Iota Ventures	3	1	2	3047903.23
8	Kappa Enterprises	3	1	2	1644070.95
9	Theta Systems	3	1	2	2813582.19
10	Zeta Solutions	3	0	3	3409589.94
11	NULL	30	7	23	25537926.36

Рисунок 2. Выходная форма №1

Для формы №2 напишем запрос:

```
DECLARE @currentDate DATE = GETDATE();
SELECT
       C.Company_name,
       Co.Contract_id AS Contract_id,
       Co.Contract_date AS Contract_date,
       Co.Size,
       Ct.Year_percentage,
       Ct.Type_of_percentage,
       Ct.Repayment_period,
       Ct.Penalty_percentage,
       dbo.ScaleLoanAmount(@currentDate, Co.Contract_id) AS Must_be_paid,
       dbo.CalculateTotalPayments(Co.Contract_id, @currentDate) AS Paid,
```

```
       CASE WHEN dbo.ScaleLoanAmount(@currentDate, Co.Contract_id) <=
dbo.CalculateTotalPayments(Co.Contract_id, @currentDate)
                    THEN 0
                    ELSE dbo.ScaleLoanAmount(@currentDate, Co.Contract_id) -
dbo.CalculateTotalPayments(Co.Contract_id, @currentDate)
                    END AS Debt
FROM Companies C
JOIN Contracts Co ON C.Code = Co.Company_code
JOIN Contract_Templates Ct ON Co.Contract_code = Ct.Code
ORDER BY C.Code;
```

Код запроса получения формы №2

Соответственно получаем результат:

	Company_name	Contract_id	Contract_date	Size	Year_percentage	Type_of_percentage	Repayment_period	Penalty_percentage	Must_be_paid	Paid	Debt
1	Alpha Industries	1	2024-04-01	100000.00	3.50	fixed	12	1.50	100290.00	850000.00	0.00
2	Alpha Industries	11	2022-12-11	600000.00	3.75	differentiated	12	1.75	715413.87	1168822.42	0.00
3	Alpha Industries	21	2024-02-21	1100000.00	5.50	fixed	48	3.00	1218108.85	0.00	1218108.85
4	Beta Corporation	22	2024-01-22	1150000.00	3.50	fixed	12	1.50	1234415.90	0.00	1234415.90
5	Beta Corporation	12	2023-11-12	650000.00	4.50	differentiated	24	2.25	759934.92	0.00	759934.92
6	Beta Corporation	2	2023-02-02	150000.00	4.25	fixed	24	2.00	167120.04	1650000.00	0.00
7	Gamma Enterprises	3	2023-01-03	200000.00	5.00	fixed	36	2.50	313587.51	0.00	313587.51
8	Gamma Enterprises	13	2022-08-13	700000.00	5.25	differentiated	36	2.75	1356929.47	0.00	1356929.47
9	Gamma Enterprises	23	2023-02-23	1200000.00	4.25	fixed	24	2.00	1689127.01	0.00	1689127.01
10	Delta Innovations	24	2024-03-24	1250000.00	5.00	fixed	36	2.50	1324178.28	0.00	1324178.28
11	Delta Innovations	14	2023-07-14	750000.00	5.50	fixed	48	3.00	784500.00	690000.00	94500.00
12	Delta Innovations	4	2023-11-04	250000.00	3.75	differentiated	12	1.75	282625.85	0.00	282625.85
13	Epsilon Technologies	5	2023-10-05	300000.00	4.50	differentiated	24	2.25	308071.56	310000.00	0.00
14	Epsilon Technologies	15	2023-05-15	800000.00	3.50	fixed	12	1.50	917994.31	360000.00	557994.31
15	Epsilon Technologies	25	2023-04-25	1300000.00	3.75	differentiated	12	1.75	1690524.81	0.00	1690524.81
16	Zeta Solutions	26	2024-04-26	1350000.00	4.50	differentiated	24	2.25	1385620.43	0.00	1385620.43
17	Zeta Solutions	16	2022-08-16	850000.00	4.25	fixed	24	2.00	1366555.86	0.00	1366555.86

Рисунок 3. Выходная форма №2

И ещё пару запросов:

```
-- Список компаний с задолженностями выше 3000000
DECLARE @currentDate DATE = GETDATE();
SELECT
    C.Company_name,
       SUM(CASE WHEN dbo.ScaleLoanAmount(@currentDate, Co.Contract_id) <=
dbo.CalculateTotalPayments(Co.Contract_id, @currentDate)
                    THEN 0
                    ELSE dbo.ScaleLoanAmount(@currentDate, Co.Contract_id) -
dbo.CalculateTotalPayments(Co.Contract_id, @currentDate)
                    END) AS Debt
FROM Companies C
LEFT JOIN Contracts Co ON C.Code = Co.Company_Code
LEFT JOIN Loan_repayments L ON Co.Contract_id = L.Contract_id
GROUP BY C.Company_name
HAVING SUM(Co.Size) - COALESCE(SUM(L.Amount), 0) > 3000000;

-- Список договоров обновляющихся на этой неделе
SELECT *
FROM Contracts
WHERE DATEDIFF(DAY, '2023-06-06', Contract_Date) <= 7 AND DATEDIFF(DAY, '2023-06-06',
Contract_Date) > 0;

-- Компании и списки их кредитов
SELECT C.Company_name,
    STUFF((SELECT ', ' + CAST(Co.Contract_id AS VARCHAR(MAX))
           FROM Contracts Co
           WHERE Co.Company_code = C.Code
           FOR XML PATH('')), 1, 2, '') AS Contracts
FROM Companies C
GROUP BY C.Company_name, C.Code
ORDER BY C.Code;
```

```sql
-- Сколько компании выплатили банку и дата последнего платежа
SELECT C.Company_name, SUM(L.Amount) AS Payments, MAX(L.Contract_date) AS Last_payment
FROM Companies C
JOIN Contracts Co ON C.Code = Co.Company_code
JOIN Loan_repayments L ON Co.Contract_id = L.Contract_id
GROUP BY C.Company_name, C.Code
ORDER BY C.Code;
```

Коды различных запросов

	Company_name	Debt
1	Alpha Industries	1218108.85
2	Iota Ventures	3047903.23
3	Kappa Enterprises	2642770.95
4	Theta Systems	2813582.19

	Contract_id	Company_code	Contract_code	Size	Contract_date
1	8	19	1	450000.00	2023-06-08
2	9	23	2	500000.00	2023-06-09

	Company_name	Contracts
1	Alpha Industries	1, 11, 21
2	Beta Corporation	2, 12, 22
3	Gamma Enterprises	3, 13, 23
4	Delta Innovations	4, 14, 24
5	Epsilon Technolog...	5, 15, 25
6	Zeta Solutions	6, 16, 26
7	Eta Group	7, 17, 27
8	Theta Systems	8, 18, 28
9	Iota Ventures	9, 19, 29
10	Kappa Enterprises	10, 20, ...

	Company_name	Payments	Last_payment
1	Alpha Industries	2018822.42	2023-12-11
2	Beta Corporation	1650000.00	2023-05-29
3	Delta Innovations	690000.00	2023-06-12
4	Epsilon Technologies	670000.00	2023-06-13
5	Eta Group	320000.00	2023-05-31
6	Theta Systems	1120000.00	2023-06-04
7	Iota Ventures	790000.00	2023-06-07
8	Kappa Enterprises	2290000.00	2023-06-10

Рисунок 4. Результаты запросов