МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«ОПТИМІЗАЦІЯ ВИКОРИСТАННЯ ПОВ'ЯЗАНИХ ДАНИХ В ПРОГРАМІ РОЗРАХУНКУ ЗАРОБІТНОЇ ПЛАТИ»**

Виконав: студент 2 курсу, групи 8.1212-іпз-1-дн
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми інженерія програмного забезпечення
(назва освітньої програми)
О.І. Кішкін
(ініціали та прізвище)
Керівник доцент кафедри програмної інженерії, Phd, Столярова А.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент доцент кафедри комп'ютерних наук, доцент, к.т.н. Матвіїшина Н.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти магістр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ ____ ” _____ 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Кішкіну Олександру Івановичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Оптимізація використання пов’язаних даних в програмі розрахунку заробітної плати

керівник роботи Столярова Анастасія Валеріївна, Phd
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 01 » травня 2023 року № 642-с

2. Строк подання студентом роботи 26.02.2024 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Реалізація та тестування системи.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень) презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 03.05.2023 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	08.05.2023	
2.	Збір вихідних даних.	09.06.2023	
3.	Обробка методичних та теоретичних	31.08.2023	
	джерел.		
4.	Розробка першого та другого розділу.	15.12.2023	
5.	Розробка третього розділу.	05.02.2024	
6.	Оформлення та нормоконтроль	19.02.2024	
	кваліфікаційної роботи магістра.		
7.	Захист кваліфікаційної роботи.	13.03.2024	

Студент _____ О.І. Кішкін
(підпис) (ініціали та прізвище)

Керівник роботи _____ А.В. Столярова
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Оптимізація використання пов’язаних даних в програмі розрахунку заробітної плати»: 76 с., 38 рис., 3 табл., 11 джерел, 7 додатків.

ВЗАЄМНІ БЛОКУВАННЯ, ЗАРОБІТНА ПЛАТА, ІНТЕГРАЦІЯ, ПАРАЛЕЛЬНІ ОБЧИСЛЕННЯ, MS SQL SERVER, SQL ЗАПИТ.

Об’єкт дослідження – програма розрахунку заробітної плати.

Мета роботи: оптимізація процесу обробки складних періодичних обчислень.

Метод дослідження – використовуючи інструменти профілювання визначити місце у коді, де витрачається найбільше часу. Оптимізувати алгоритми, запити, використати паралельні обчислення.

У цій роботі ми розглянемо причини, що впливають на поведінку програм, проаналізуємо методи підвищення продуктивності та збільшення кількості одночасно працюючих користувачів.

Для цього розглянемо фактори, які впливають на виконання робіт: вибрана технологія (клієнт-сервер, термінал-сервер і т.п.), мережеві налаштування, підбір конфігурації обладнання, параметри операційної системи сервера, налаштування бази даних, виконання регламентних робіт, аналіз програмного коду, аналіз запитів до бази даних, налаштування виключення блокування в результаті обчислень, підвищення ефективності роботи служби управління персоналом, автоматизація потоків і спрощення процесів інтеграції, підвищення якості даних, що передаються між внутрішньою і зовнішньою підсистемами.

В результаті ми повинні мати можливість одночасно працювати з великою кількістю користувачів, скоротити час на виконання обчислень і зменшити частку ручних операцій.

SUMMARY

Master’s qualifying paper «Optimization of the Connected Data Usage in the Accounting Software»: 76 pages, 38 figures, 3 tables, 11 references, 7 supplements.

DEADLOCKS, PAYROLL, INTEGRATION, CALCULATION OPTIMIZATION, CLIENT-SERVER, MS SQL, TERMINAL-SERVER.

In the course of the work of the enterprise, changes are made to the running programs, the functionality increases. Databases are growing in size. This significantly increases the load. Programs that worked well yesterday begin to lose speed, or even stop completely.

In this work, we will consider the causes that affect the behavior of programs, analyze methods for improving performance, and increasing the number of concurrent users.

To do this, consider such factors that affect the performance of work: selected technology (client-server, terminal-server, etc.), network settings, selection of equipment configuration, server operating system settings, database settings, carrying out routine work, program code analysis, analysis of queries to the database, settings to exclude blocking as a result of calculations, improving the efficiency of the personnel management service, automation of flows and simplification of integration processes, improving the quality of transmitted data between internal and external subsystems.

As a result, we should be able to simultaneously work with a large number of users, reduce the time for performing calculations, and reduce the proportion of manual operations.

ЗМІСТ

ВСТУП

Управління заробітною платою є важливою функцією будь-якої організації. Точна та своєчасна обробка заробітної плати має вирішальне значення для забезпечення задоволеності працівників і дотримання вимог законодавства. Дані для розрахунку заробітної плати отримують з різноманітних внутрішніх і зовнішніх джерел, таких як системи обліку робочого часу та відвідуваності, облікових записів працівників, податкових органів та постачальників пільг. Оптимізація набору пов'язаних даних у програмі нарахування заробітної плати може призвести до підвищення ефективності, точності та економічної ефективності. У цьому огляді розглядаються попередні дослідження, пов'язані з оптимізацією набору даних у програмах нарахування заробітної плати.

Обробка заробітної плати вимагає точного та своєчасного введення даних, пов'язаних із винагородами та відрахуваннями працівників. Неточності або затримки в наборі даних можуть призвести до помилок у обробці заробітної плати, що спричинить незадоволення працівників і потенційні правові наслідки. Автоматизація та інтеграція джерел даних може призвести до підвищення точності та ефективності обробки заробітної плати.

Інтеграція даних є критично важливим компонентом обробки заробітної плати. Інтеграція дозволяє безперешкодно передавати дані між системами, зменшуючи ймовірність помилок і підвищуючи ефективність. Використання інтегрованої системи нарахування заробітної плати призводить до зменшення кількості помилок і підвищення ефективності.

Аутсорсинг заробітної плати стає все більш популярним варіантом для організацій, які прагнуть оптимізувати обробку заробітної плати. Аутсорсинг дозволяє організаціям зосередитися на своїх основних компетенціях, залишаючи обробку заробітної плати експертам. Аутсорсинг призводить до підвищення точності та ефективності, а також до економії коштів для

організації.

Хмарні системи розрахунку заробітної плати є ще одним варіантом для організацій, які хочуть оптимізувати обробку заробітної плати. Хмарні системи дозволяють бездоганно інтегрувати дані з багатьох джерел і надають доступ до даних про заробітну плату в реальному часі. Використання хмарних систем призводить до підвищення ефективності, підвищення точності та економії коштів для організації.

Отже, оптимізація збору даних у програмах нарахування заробітної плати має вирішальне значення для забезпечення точної та своєчасної обробки заробітної плати. Попередні дослідження показали, що автоматизація та інтеграція джерел даних, аутсорсинг заробітної плати та хмарні системи розрахунку заробітної плати можуть призвести до підвищення ефективності, точності та економічної ефективності. Організації повинні розглянути ці варіанти, коли хочуть оптимізувати обробку заробітної плати. Необхідні подальші дослідження, щоб визначити оптимальний підхід для кожної організації на основі її конкретних потреб і вимог.

1 ТЕХНІЧНЕ ЗАВДАННЯ ТА РЕАЛІЗАЦІЯ

1.1 Постановка задачі

Раніше на великих підприємствах заробітну плату розраховували безпосередньо на місцях. В кожному місті була своя програма. Це призводило до необхідності підтримувати велику кількість програм, для отримання звіту по підприємству доводилось збирати та консолідувати звіти. Це ускладнювало роботу, збільшувало витрати на супровід. Звіти на підприємстві отримували після завершення звітного періоду. В той час інтернет був не стабільним та повільним, а вартість великою.

З часом швидкість та надійність зросли, вартість знизилась. З'явилась можливість об'єднати усі програми в одну. Потужність обладнання стала достатньою для встановлення програмного забезпечення та ведення розрахунків по всьому підприємству. В загальну програму додавали дані та приєднували користувачів.

Для роботи маленької бази достатньо невеликої віртуальної машини. Після об'єднання усіх даних підприємства може не вистачити можливостей самої сучасної машини. База може зростати на декілька гігабайт за місяць. В програмі може бути зареєстровано декілька сотень користувачів, а одночасно працювати більше сотні. Програма, яка відмінно працювала при не великому навантажені, може зовсім зупинитися в період масового вводу чи розрахунку заробітної плати.

Причиною такої роботи може бути блокування даних. Про можливі проблеми при збільшені навантаження та обсягів відомо, але є переваги. Одну програму простіше обслуговувати, та отримувати звіти по підприємству без додаткової обробки та інше. Для невеликої бази достатньо невеликої віртуальної машини. Для бази підприємства з великою кількістю одночасно

працюючих співробітників недостатньо оптимізувати обладнання, тому потрібно оптимізувати базу даних та програмну частину. Частково проблема може бути вирішена організаційно, наприклад працювати по черзі. В даній роботі це не розглядаємо. Також не розглядаємо варіанти розділу в часі ведення інформації від розрахунку.

1.2 Загальний порядок роботи програми розрахунку

Щомісячно для розрахунку заробітної плати водять початкові дані. Документи руху кадрів (прийом, звільнення, переміщення), графіки робіт, табель відпрацьованого часу, документи невиходів такі як лікарняні, відрядження, відпустки, акти виконаних робіт та інше.

На підставі введених даних розраховують спочатку відрядження, відпустки, лікарняні та інші документи. Після чого приступають до документів розрахунку заробітної плати.

Особливість розрахунку полягає в тому, що рядки в документі залежні. Розрахунок ведеться по рядкам. В першу чергу розраховуються рядки з видами розрахунку, які не залежать від інших нарахувань, тому мають нульовий пріоритет. Це види розрахунку фіксованою сумою та види розрахунку такі як пропорція відпрацьованих днів від днів по графіку робіт – оплата по часовій та по денній тарифній ставці. Після розраховують види розрахунку такі як процент від бази, наприклад премія за поточний місяць чи надбавка за стаж (див. рис. 1.1).

Пріоритети (порядок розрахунку) – задається в параметрах виду розрахунку. При розрахунку рядка документа виконується запис в регістр розрахунку. Регістри є вузьким місцем програми, тому як всі результати записується в регістри.

Після виконання всіх розрахунків формують звіти [6].

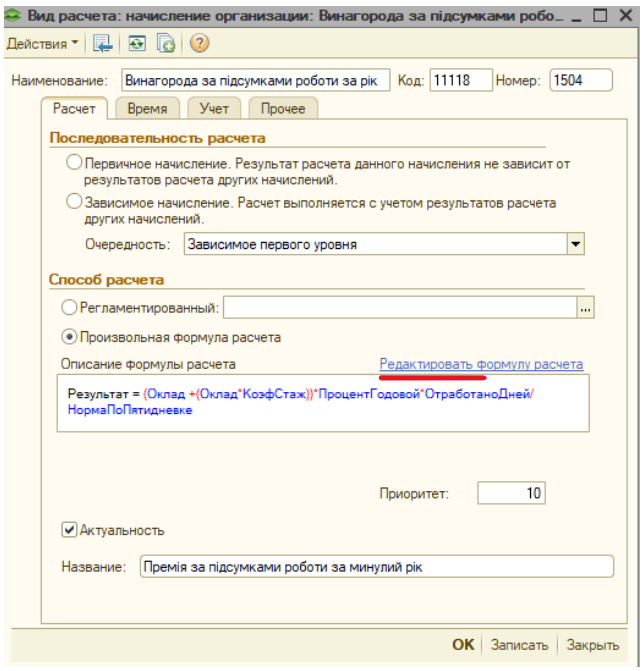


Рисунок 1.1 – Приклад налаштування виду розрахунку

Іноді бувають складні випадки, наприклад, співробітнику нараховують заробітну плату, потім співробітник за цей час бере відпустку, а під час відпустки бере лікарняний. В цьому випадку нарахування по заробітної платні будуть витіснені нарахуваннями по відпустці. А після оформлення лікарняного – частину нарахувань відпустки будуть витиснені за період дії лікарняного, в випадку оформлення лікарняного наступним місяцем частину нарахувань відпустки по періоду дії лікарняного будуть сторновані [7].

Налаштування витіснення видів нарахувань проводитися в кожному виді нарахувань на вкладці «Інші налаштування» в таблиці витіснення.

Для розуміння періоду дії виду розрахунку зробимо звіт. Звіт побудуємо на основі даних віртуальній таблиці «фактичний період дії» регістру «основні нарахування». Для візуального представлення побудуємо діаграму Ганта (див. рис. 1.2).

На діаграмі бачимо що основне нарахування таке як «оклад/тариф» було спочатку витіснена нарахуванням «оплата відпустки» в періоди з 05 по 10, а потім з 12 по 13, і ще раз з 15 по 17. Після цього внесли документ «оплата лікарняних», за допомогою якого з початку сторнирували нарахування за

період лікарняного. Було сторноване один день відпустки і два дні оплати по окладу. Після чого нараховані лікарняні.

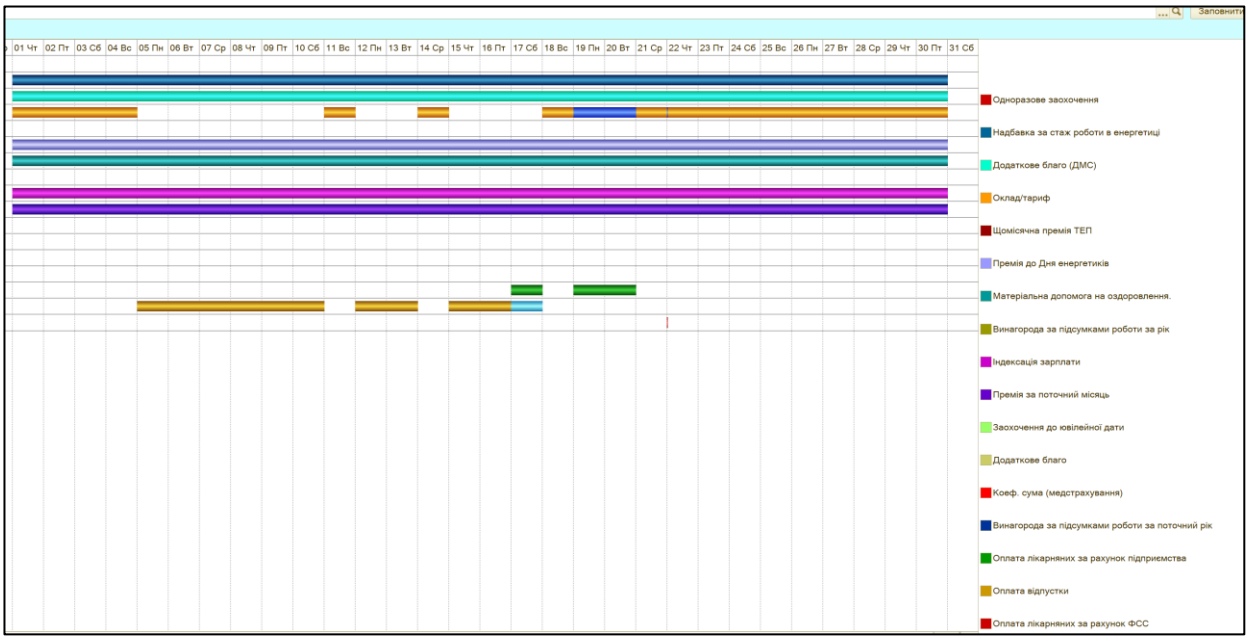


Рисунок 1.2 – Приклад звіту з діаграмою Ганта

На рисунку 1.3 представлено приклад коду модулю звіту для побудови діаграми Ганта, а на рисунку 1.4 – приклад звіту з діаграмою Ганта в конструкторі звітів. Код модулю звіту наведено в додатку Е.

```
&НаКлиенте
Процедура ЗаполнитьДанными(Команда)
    ПолучитьДанныеФПД(ДиаграммаГанта, Сотрудник);
КонецПроцедуры

&НаСервереБезКонтекста
Процедура ПолучитьДанныеФПД(Диаграмма, Сотрудник)
    Запрос = Новый Запрос;

    Запрос.Текст = "ВЫБРАТЬ
                   |    НачисленияФактическийПериодДействия.Сотрудник,
                   |    НачисленияФактическийПериодДействия.ВидРасчета,
                   |    НачисленияФактическийПериодДействия.ПериодДействияНачало,
                   |    НачисленияФактическийПериодДействия.ПериодДействияКонец,
                   |    НачисленияФактическийПериодДействия.Регистратор
                   |ИЗ
                   |    РегистрРасчета.ОсновныеНачисленияРаботниковОрганизаций.ФактическийПериодДействия(Сотрудник = &Сотрудник) КАК НачисленияФактическийПериодДействия";
    Запрос.УстановитьПараметр("Сотрудник", Сотрудник);
    РезультатЗапроса = Запрос.Выполнить();

    Диаграмма.Обновление = Ложь;
    Диаграмма.Очистить();

    Выборка = РезультатЗапроса.Выбрать();
    Пока Выборка.Следующий() Цикл
        Точка = Диаграмма.УстановитьТочку(Выборка.Сотрудник);
        Серия = Диаграмма.УстановитьСерию(Выборка.ВидРасчета);

        Значение = Диаграмма.ПолучитьЗначение(Точка, Серия);

        Значение.расшифровка =Выборка.регистратор;

        Интервал = Значение.Добавить();
        Интервал.Начало = Выборка.ПериодДействияНачало;
        Интервал.Конец = Выборка.ПериодДействияКонец;
    КонецЦикла;

    Диаграмма.Обновление = Истина;
    Диаграмма.ОтображатьПустыеЗначения=Ложь;
    Диаграмма.ПоддержкаМасштаба = ПоддержкаМасштабаДиаграммыГанта.Фиксированная;

КонецПроцедуры

&НаКлиенте
Процедура СотрудникПриИзменении(Элемент)
ПолучитьДанныеФПД(ДиаграммаГанта, Сотрудник);
КонецПроцедуры
```

Рисунок 1.3

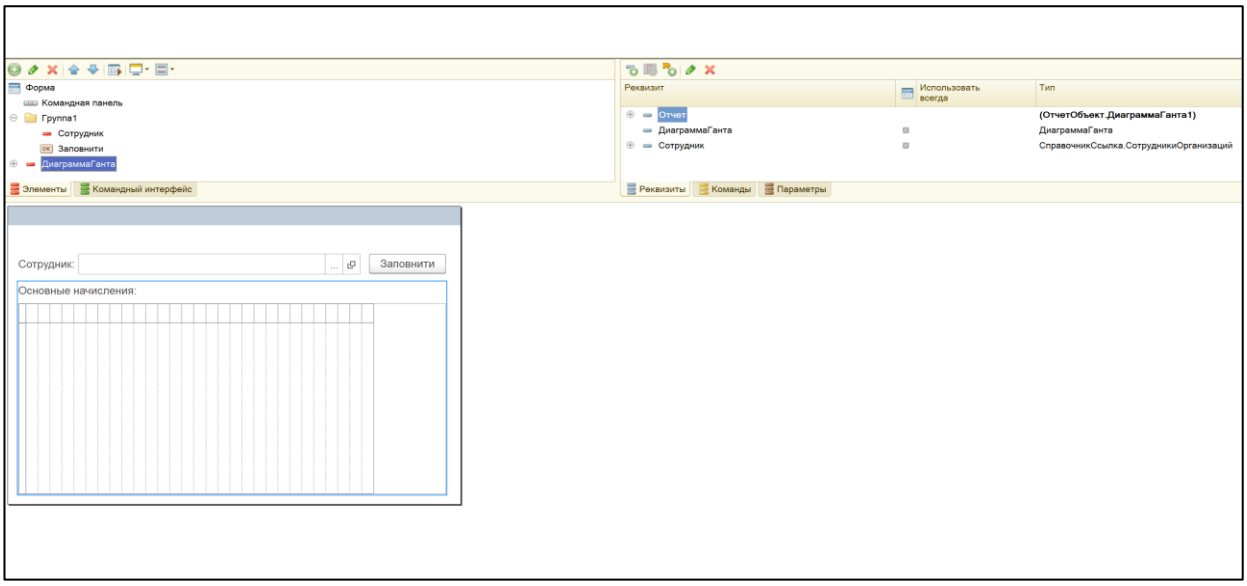


Рисунок 1.4

1.2.1 Розрахунок запланованої зарплати одним запитом

Розглянемо питання – чи є можливість порахувати регламентовану заробітну плату, наприклад, в Excel, чи необхідно використовувати складні спеціалізовані програми.

Для розрахунку фонду заробітної плати чи управлінського обліку є можливість спростувати розрахунок, не враховувати графіки робіт, фактично відпрацьований час, та не виходи на роботу, такі як відрядні чи відпустки. А лікарняні з отриманої суми відрахувати з врахуванням статистичних даних за попередній період. Цей підхід добре працює для планування розрахунків в разі планування підвищення окладів чи премій. Приклад такого розрахунку наведено в додатку Д.

Цей розрахунок виконується на SQL сервері за декілька секунд. Результат запиту одною командою додається до табличної частини документу. Але для розрахунку регламентованої заробітної плати цей метод використовувати не можливо [10].

1.2.2 Реалізуємо розрахунок днів відпрацьованих у відряджені

На деяких підприємствах відрядні не розраховують, а сплачують так, як нібито ці дні відпрацьовані співробітником без відрядження. Згідно з нормативними актами, працівнику у відрядженні потрібно виконати розрахунок за середнім та розрахувати скільки би отримав співробітник в разі коли у відряджені не був. А сплатити ту суму, яка буде більша.

Цей вид розрахунку не реалізовано в програмі, також цей вид розрахунку не вийде налаштувати без програмування. Для виконання цього алгоритму потрібно виконати два розрахунки, порівняти результати, зарахувати більший.

Розрахунок заробітної плати – складний розрахунок, на який впливає багато параметрів.

Для того щоб не дублювати розрахунок виконаний в документі нарахування заробітної плати, простіше спочатку розрахувати документи нарахування заробітної плати, потім виконати розрахунок документів для розрахунку відрядних. В такому разі буде можливість порівняти суму нарахування заробітної плати з результатом нарахувань виконаних в документі нарахування відрядних. В документ відрядних врахувати суму яка буде більша. Після чого перерахувати документи нарахування заробітної плати для перерахунку податків, внесків та утримань.

Цей алгоритм ускладнює розрахунок. Тому як усі спеціалісти по розрахунку заробітної плати знають, що потрібно спочатку розрахувати документи не виходів (відпустка, лікарняні), а потім розраховувати зарплату. При реалізації цього алгоритму є особливість для розрахунку відрядних.

Іншим суттєвим недоліком є необхідність в розрахунку заробітної плати двічі. Перший розрахунок потрібен для отримання суми для порівняння, другий – для того, щоб не нараховувалася заробітна плата за дні відрядження та розрахунку внесків та податків.

Розрахунок відрядних виконується в документі «оплата по середньому заробітку».

На рисунку 1.5 видно, що для розрахунку середньоденної заробітної плати використовуються три види розрахунку за кожен місяць.

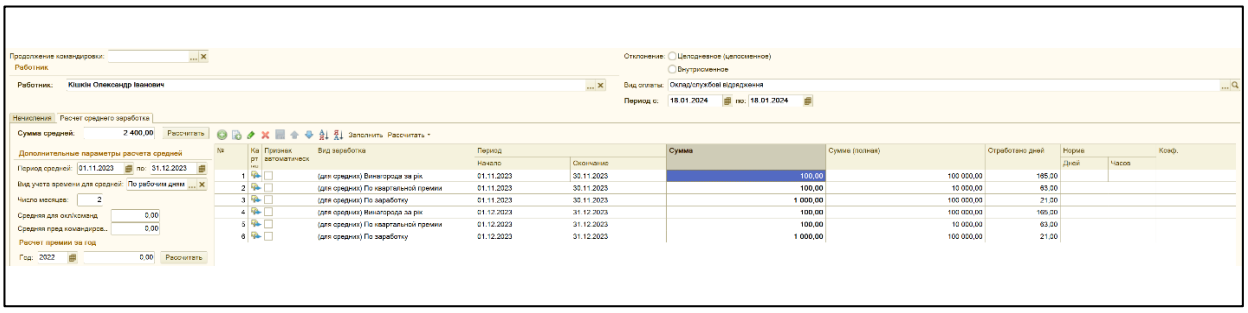


Рисунок 1.5 – Приклад розрахунку відрядних

Перший вид розрахунку «(для середніх) По заробітку». Для розрахунку потрібно скласти нарахування, які входять до бази даного виду розрахунку, за період розрахунку (місяць), та поділити на кількість відпрацьованих днів.

Другий вид розрахунку «(для середніх) По квартальної премії». Особливістю цього розрахунку є те, що база береться за квартал, а до бази беруться ті види нарахувань, базовий період в яких квартал. Відпрацьовано днів також потрібно брати за квартал. Наприклад «квартальна премія».

Особливістю третього виду нарахувань – є те, що база та відпрацьовані дні для нарахування беруться за попередній рік. В базові нарахування включаються нарахування з базовим періодом рік.

Всі ці розрахунки потрібні для розрахунку середньоденної заробітної плати, яка складається з усіх нарахувань поділена на кількість відпрацьованих днів.

В зв’язку з тим, що для розрахунку середньоденної заробітної плати необхідні дані за попередній рік, а то і два. Є особливість в програмах розрахунку заробітної плати. При переході на іншу програму виникає необхідність в перенесені фактичних нарахувань, фактично відпрацьованих днів, та днів по графікам не менш ніж за два попередніх роки. Інакше прийдеться розраховувати документи невиходів в ручну, отримуючи данні з розрахункових листиків, які зберігаються в архіві [7].

1.2.3 Введення та розрахунок оплати за роботу у вихідний день

Розглянемо додавання в програмі розрахунку заробітної плати обліку днів та автоматизацію нарахування за роботу у вихідний день. В конфігурації від постачальника це не реалізовано.

Для автоматизації обліку днів відпрацьованих у вихідний, додаємо в стан співробітника значення «Робота у вихідний з одинарною оплатою (РВО)» та «Робота у вихідний з подвійною оплатою (РВП)». Це дає можливість створювати документи відповідно до наказів на роботу у вихідний день. В наказі може бути два випадка, робітник отримує одну оплату і відгул, чи робітнику сплачують за роботу в подвійному розмірі.

Для введення наказу використаємо документ «неявки та хвороби» інша назва «відсутність на роботі в організації». Для можливості вибору нових видів відсутності додаємо в загальному модулі «процедури управління персоналом» в функцію отримання причин відсутності такі строки, як наведено на рисунку 1.6.

```
СписокСпособов.Добавить(Перечисления.СостоянияРаботникаОрганизации.РоботаУВихіднийЗОдинарною Оплатою);
СписокСпособов.Добавить(Перечисления.СостоянияРаботникаОрганизации.РоботаУВихіднийЗПодвійною Оплатою);
```

Рисунок 1.6

В довіднику «класифікатор обліку робочого часу» додаємо види часу – «Робота у вихідний день з одинарною оплатою по наказу РВО» і «Робота у вихідний день з подвійною оплатою по наказу РВП».

Це дає можливість на підставі введеного документу «неявки та хвороби» врахувати при заповнені документу «табель обліку робочого часу». Для цього вносимо зміни до модулю документу в процедуру авто заповнення (див. рис. 1.7).

```
|КОГДА Основной.Состояние = ЗНАЧЕНИЕ(Перечисление.СостоянияРаботникаОрганизации.РоботаУВихіднийЗПодвійноюОплатою)
|ТОГДА ЗНАЧЕНИЕ(Справочник.КлассификаторИспользованияРабочегоВремени.РоботаПодвійнаОплата)
|КОГДА Основной.Состояние = ЗНАЧЕНИЕ(Перечисление.СостоянияРаботникаОрганизации.РоботаУВихіднийЗОдинарноюОплатою)
|ТОГДА ЗНАЧЕНИЕ(Справочник.КлассификаторИспользованияРабочегоВремени.РоботаОдинарнаОплата)
```

Рисунок 1.7

В «планах виду розрахунку» / «основні нарахування» створюємо види розрахунку для оплати в одинарному та подвійному розміру (див. рис. 1.8).

Вносимо зміни в обробку «аналіз неявок» для автоматизації створення документів розрахунку.

В модулі форми документа в процедурі «Виконати Авто Розрахунок Реквізитів Строк Нарахування» вносимо зміни для розрахунку результату.

```
ДанныеСтроки.Результат = ДанныеСтроки.Размер * (ДанныеСтроки.ОтработаноЧасов + ДанныеСтроки.ЧасовДвойных * 2);
```

Рисунок 1.8

В модулі документу «оплата святкових і вихідних днів» створюємо процедуру «обробка заповнення» додаємо код наведений в додатку Ж. Це дасть змогу створювати та заповнювати документи на підставі кадрового документу «неявки та хвороби» – документ оплата святкових і вихідних днів.

Вносимо зміни в інші види розрахунків. Необхідно додати цей вид нарахування в базові нарахування інших видів нарахування.

Вносимо зміни в звіти для врахування нових видів розрахунків.

Таким чином ми бачимо, що для створення нового виду розрахунку виникла необхідність вносити зміни в довідники, загальні модулі, в документи та звіти [11].

2 ОПТИМІЗАЦІЯ ПРОГРАМНОЇ ЧАСТИНИ

2.1 Прискорення введення первинних документів

В зв’язку з великим обсягом робіт по щомісячному вводу даних для розрахунку заробітної плати, розроблено шаблони для заповнення даних на містах. Заповнені документи пересилаються на пошту відповідальний особі для завантаження в програму.

2.1.1 Приклад шаблону для завантаження актів виконаних робот

За місяць на підприємстві оформлюють велику кількість документів. Документи формуються на місцях виконання робіт та передаються відповідальній особі для завантаження в програму розрахунку (рис. 2.1).

Табель обліку робочого часу по персоналу
за _травень_ 2023 р.

№ Акту	Табельний №	ПІБ	Вид роботи (31 - поточний; 32 - капітальний)	Найменування об'єкту	Інв. Номер	Відпрацьований час (години)	Роз'їздні (дні)	КОД ПОТОЧНОГО ПІДРОЗДІЛУ
0404_10213203	64608		32		030122_4	30,4		
0404_10213203	64610		32		030122_4	30,4		

Рисунок 2.1 – Зовнішній вигляд файлу шаблону для завантаження

Для завантаження отримані документи складаються в папку. В програмі за натисканням кнопки вказується папка з документами і виконується завантаження.

Особливістю обробки є те, що можливо завантаження виконувати декілька разів. При завантажені строки з шаблону, зчитуєтеся номер акту, розшукується документ, в разі відсутності – створюється, та рядок з шаблону додається в документ, якщо знайдено документ, рядок замінює існуючій рядок в документі. Приклад обробки наведено в додатку А [8].

2.1.2 Приклад шаблону табелю обліку робочого часу

Двічі на місяць потрібно збирати дані по відпрацьованому часу. Для розрахунку авансу та заробітної плати. Окрім обробки для завантаження з файлу на диску, аналогічно актам виконаних робот, було реалізовано обробку для завантаження даних із шаблону методом копіювати вставити (див. рис. 2.2). Співробітник може натиснути в шаблоні кнопку «скопіювати» чи видалити необхідні строки та скопіювати, та вставити в документі. Цей метод почали використовувати в зв’язку з тим, що не всі працівники використовували рекомендований шаблон. Хтось передав дані в Excel, хтось в Word чи в PDF файлі. Є можливість скопіювати на одному робочому столі, а вставити на іншому. Без необхідності передавати файли.

Приклад обробки наведено в додатку Б.

05 Організація

Найменування підприємства (установи організації)

Табель обліку використання робочого часу

05-15 назва структурного підрозділу

за май 2023 р. Скопировать

№	ПІБ	Таб. №	Відмітки про явки та неявки за числами місяця (годин)																														
			01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
			пн	вт	ср	чт	пт	сб	вс	пн	вт	ср	чт	пт	сб	вс	пн	вт	ср	чт	пт	сб	вс	пн	вт	ср	чт	пт	сб	вс	пн	вт	ср
1		99990	1	3				3				3			3				3	8/7	8/8			8/4	8/5	8/6	8/7	8/8			8/4	8/5	8/6
2		10-02004	8/4	8/5	8/6	8/7	8/8			8/4	8/5	8/6	8/7	8/8			8/4	8/5	8/6	8/7	8/8			8/4	8/5	8/6	8/7	8/8			8/4	8/5	8/6
3		09-15062	1		2		1		1		1		3		1			1	1		2			1		1		2		3			
4		06-11055	р	в	в	в	в			р	р	р	р	р			р	р	р	р	р			р	р	р	р	р			р	р	р
5		99942	1		2		3				1			2																			
6		09-11012	р	р	р	р	р			в	в	в	в	в			вд	вд	р	р	р			р	р	р	р	р			р	р	р
7		06-02034	р-6 рн-4 шу-6	р-8 рн-6 шу-8		р-6 рн-4 шу-8	р-8 рн-6 шу-8			р-6 рн-4	р-8 рн-6		р-6 рн-4	р-8 рн-6		р-6 рн-4	р-8 рн-6		р-6 рн-4	р-8 рн-6		р-6 рн-4	р-8 рн-6		р-6 рн-4	р-8 рн-6		р-6 рн-4	р-8 рн-6		р-6 рн-4	р-8 рн-6		р-6 рн-4	р-8 рн-6	

Рисунок 2.2 – Приклад файлу шаблону для завантаження

Аналогічно зашлюзуються інші данні. При великих обсягах робіт, за обмежений час, вводити всі вхідні данні працівниками не тільки трудомістко, а також можливі помилки вводу [9].

2.2 Усунення блокування даних

В зв’язку з великими обсягами одночасного вводу та обробки даних різними працівниками, найбільш актуальною є проблема блокування даних.

Для вирішення цієї проблеми необхідно зменшити час блокування даних та знизити рівень ескалації блокування.

Для зменшення часу накладання блокування потрібно накладати блокування лише під час виконання обробки проведення документу. Усі перевірки та розрахунки виконувати до проведення документу, при проведені документів виконувати лише запис до регістрів. Також зменшити час можливо підвищенням швидкості робіт.

Для усунення зайвого блокування даних потрібно відмовитись від автоматичного блокування на рівні SQL сервера, перейти на програмні. Налаштувати SQL-сервер та сервер-додатків на максимальне розпаралелювання.

2.2.1 Налаштування операційної системи

Використовуємо 64-розрядні версії операційної системи, сервера додатків, та сервера баз даних. Не буде проблем з нестачею оперативної пам’яті.

Оновлюємо все до останньої стабільної версії.

Для того, щоб сервер не засинав, необхідно перевести режим живлення на максимальну продуктивність (рис. 2.3).

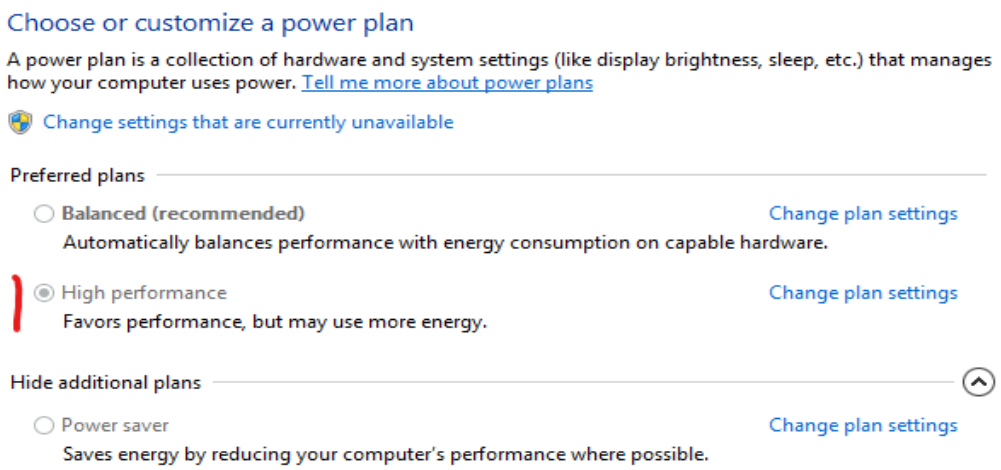


Рисунок 2.3 – Налаштування високопродуктивного режиму роботи

В панелі керування вибрати «електроживлення» / відобразити додаткові схеми / вибрати «висока продуктивність».

В налаштуваннях програм антивірусного захисту додати до виключення файли бази даних.

В налаштуваннях файрвола закрити усі порти, окрім тих, що використовуються.

2.2.2 Налаштування MS SQL сервера

Налаштування виконуються згідно до рекомендацій розробника програмного забезпечення. Розглянемо деякі з особливостей.

Сервер додатків та MS SQL-сервер краще розмістити разом. Налаштувати взаємодію через протокол доступу до загальної пам’яті «Shared Memory».

В зв’язку з тим, що сервер додатків MS SQL-сервер знаходяться на одному комп’ютері необхідно обмежити MS SQL-сервер в використанні оперативної пам’яті (рис. 2.4). Від загального обсягу віднімаємо пам’ять для операційної системи та серверу додатків. Через деякий час MS SQL-сервер займає всю доступну пам’ять.

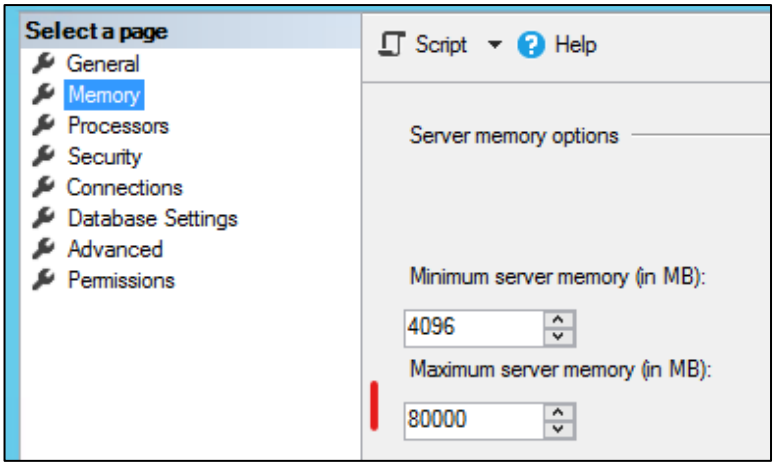
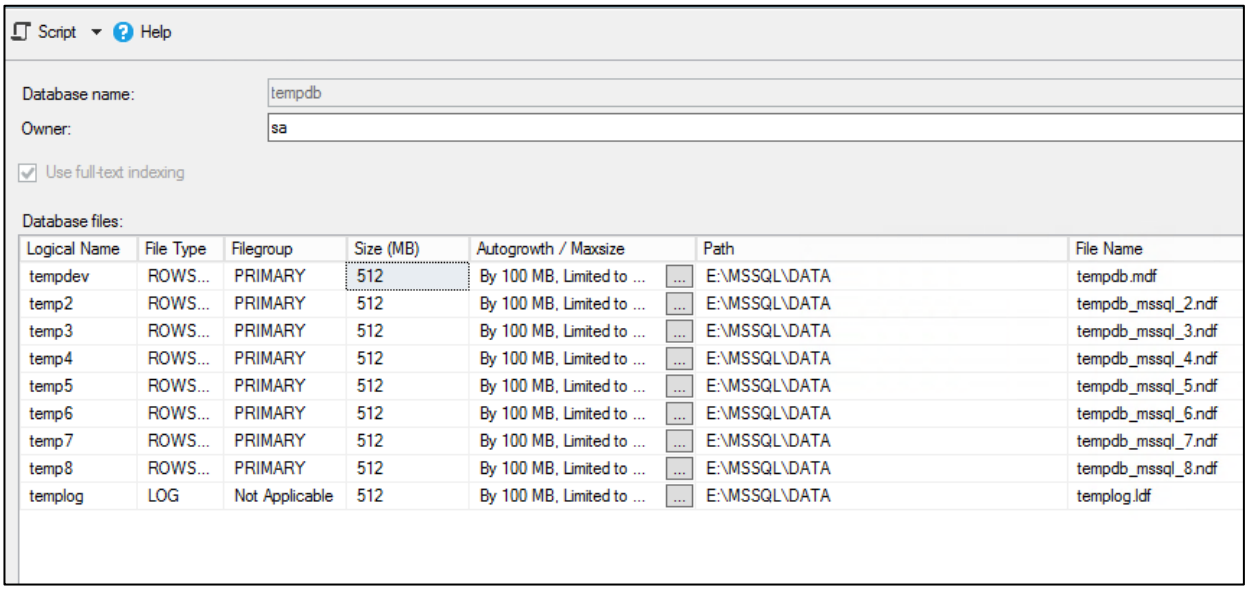


Рисунок 2.4 – Налаштування обмеження використання пам’яті MS SQL-сервером

В налаштуваннях бази встановити приріст для лог. файлів 500 Мб, і в для бази даних 1000 Мб.

Встановити операційну систему, базу даних, файли логів та базу для тимчасових даних на різні диски.

Базу для тимчасових даних розділили на різні файли (рис. 2.5).



Logical Name	File Type	Filegroup	Size (MB)	Autogrowth / Maxsize	Path	File Name
tempdev	ROWS...	PRIMARY	512	By 100 MB, Limited to ...	E:\MSSQL\DATA	tempdb.mdf
temp2	ROWS...	PRIMARY	512	By 100 MB, Limited to ...	E:\MSSQL\DATA	tempdb_mssql_2.ndf
temp3	ROWS...	PRIMARY	512	By 100 MB, Limited to ...	E:\MSSQL\DATA	tempdb_mssql_3.ndf
temp4	ROWS...	PRIMARY	512	By 100 MB, Limited to ...	E:\MSSQL\DATA	tempdb_mssql_4.ndf
temp5	ROWS...	PRIMARY	512	By 100 MB, Limited to ...	E:\MSSQL\DATA	tempdb_mssql_5.ndf
temp6	ROWS...	PRIMARY	512	By 100 MB, Limited to ...	E:\MSSQL\DATA	tempdb_mssql_6.ndf
temp7	ROWS...	PRIMARY	512	By 100 MB, Limited to ...	E:\MSSQL\DATA	tempdb_mssql_7.ndf
temp8	ROWS...	PRIMARY	512	By 100 MB, Limited to ...	E:\MSSQL\DATA	tempdb_mssql_8.ndf
templog	LOG	Not Applicable	512	By 100 MB, Limited to ...	E:\MSSQL\DATA	templog.ldf

Рисунок 2.5 – Приклад розділення бази TempDB

В налаштуваннях бази встановити режим «Ізоляція моментальних знімків» (рис. 2.6). В цьому режимі дані, використані в транзакції, копіюються до tempdb і не блокують дані в базі. Це дозволяє іншим транзакціям виконуватися без блокування попередньої не завершеної.

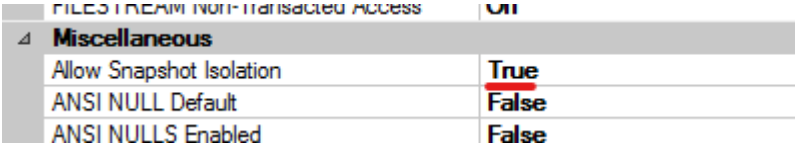


Рисунок 2.6 – Приклад включення ізоляція моментальних знімків

Налаштовуємо автоматичне резервне копіювання та виконання регламентних робіт [1].

2.2.3 Регламентні роботи на сервері баз даних

Для підтримки швидкості роботи бази потрібно виконувати регламентні роботи (рис. 2.7). Кожну ніч виконуються такі дії як перевірка бази, реорганізація індексів, відновлення статистики [2].

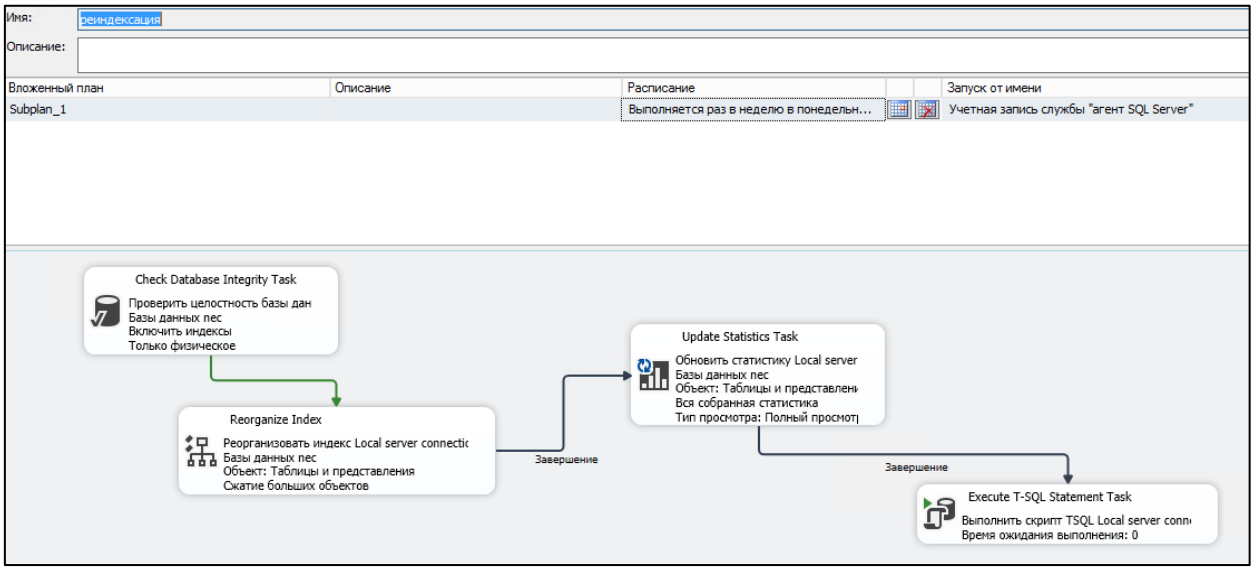


Рисунок 2.7 – Приклад налаштування регламентного завдання

При аналізі дефрагментації індексів було виявлено в великій таблиці, в яку записуються графіки робіт, індекси в таблиці швидко дефрагментуються. Тому додано регламентне завдання кожні пару годин перебудовувати індекси (рис. 2.8).

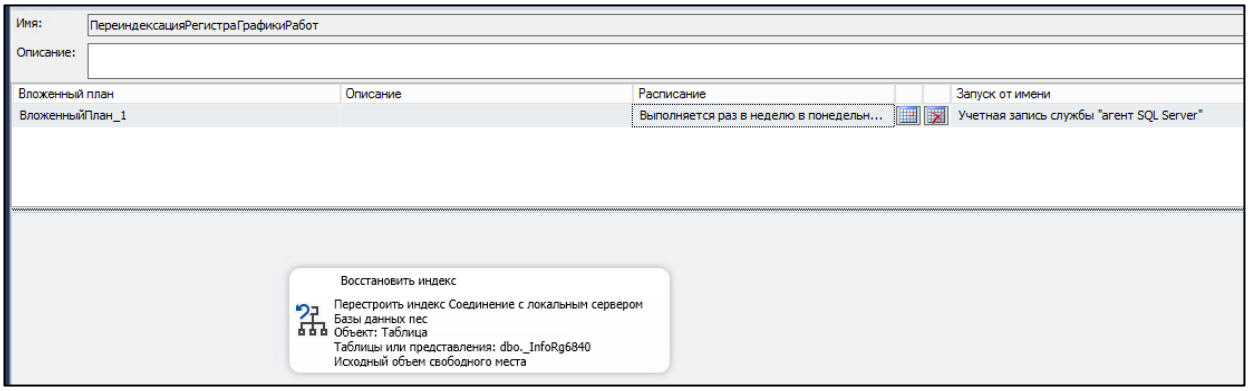


Рисунок 2.8 – Приклад регламентного завдання побудови індексів

2.2.4 Індексація полів в базі даних

Істотне збільшення швидкості пошуку в базі даних дає додавання індексів (рис. 2.9). Особливо це помітно на таблицях з мільйонами записів. Швидкість зростає в десятки, а то і в сотні разів. Відсутність індексу по полю пошуку веде до перебору усіх записів таблиці, для великих таблиць це займає багато часу і надмірному блокуванню записів. Тому спочатку будуємо індекси по усім полям, по яким виконуємо пошук, упорядкування чи відбір даних. Усі ключові поля індексуються за замовчуванням. Індексувати всі поля не варто, тому як індекс підвішує швидкість пошуку та зменшує швидкість запису. Для перевірки де ще необхідно додати індекси є інструмент, який аналізує запити.

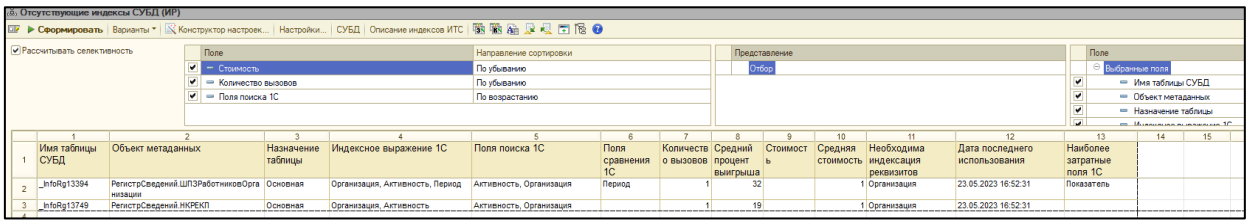


Рисунок 2.9 – Приклад обробки з рекомендацією додавання індексів

Окрім цього є можливість проаналізувати запити, які виконуються найдовше чи частіше (рис. 2.10).

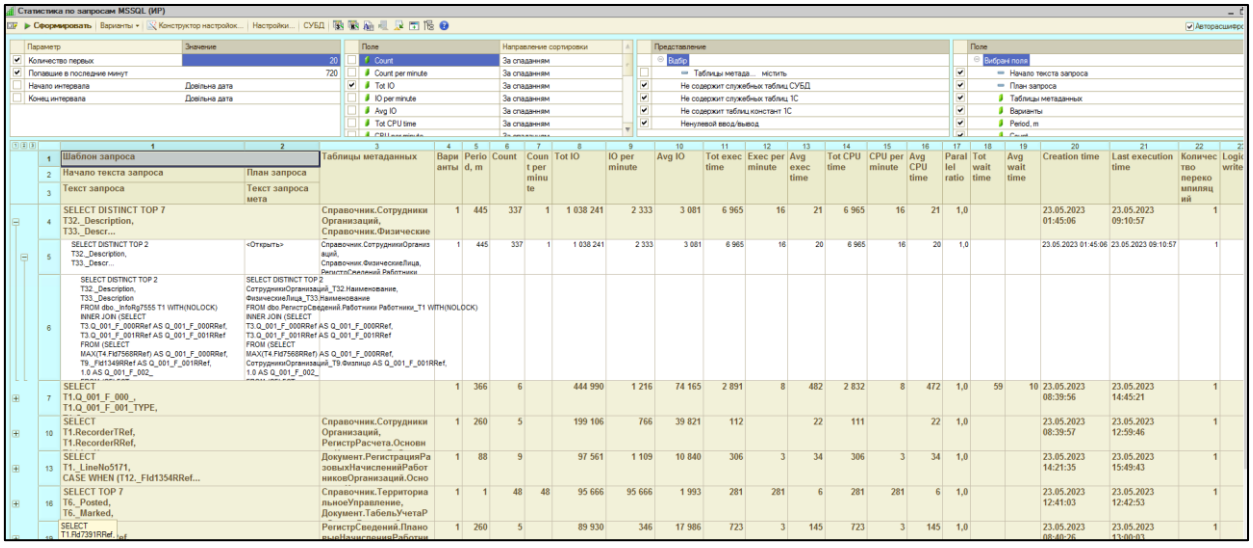


Рисунок 2.10 – Приклад обробки аналізу запитів

В профайлері при перегляді плану запиту є можливість скористуватись підказкою в разі, коли така є (рис. 2.11). Чи проаналізувати план запиту самостійно, виявити вузькі місця, і додати індекси, чи переробити запит інакше.

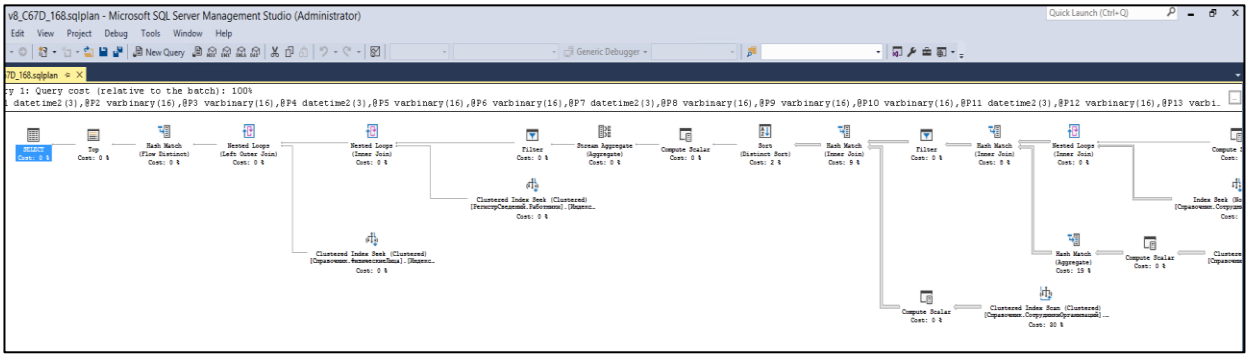


Рисунок 2.11 – Приклад плану запиту

2.3 Налаштування серверу додатків

Найважливішим налаштуванням є переведення серверу додатків з режиму блокування SQL-сервера на блокування серверу додатків (рис. 2.12).

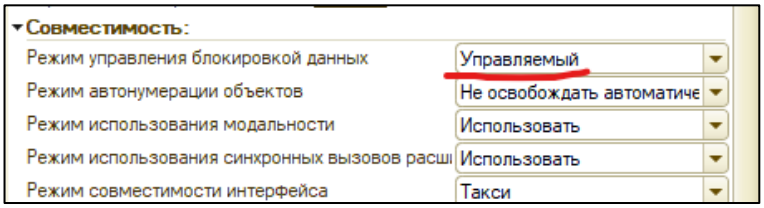


Рисунок 2.12 – Налаштування режиму блокування

2.4 Аналіз та оптимізація програмного коду

Звернемо увагу на дії, які використовуються часто та обчислення яких займає багато часу.

2.4.1 Оптимізація створення табелів обліку робочого часу

Таким є формування табелю робочого часу. За місяць формується багато документів. Час створення документу декілька хвилин. Для аналізу часу виконання окремих команд виконуємо замір продуктивності. За результатами аналізу коду виявили, що, незалежно від того новий документ чи ні, виконується обробка видалення записів з регістрів займає багато часу. Але це новий документ і в регістрі записів ще не повинно бути. Тому дописали перевірку для нових документів видалення не виконувати (рис. 2.13). Це зменшило час створення документу з декількох хвилин до менше ніж секунди. Час створення табелів по підприємству зменшився суттєво. Це можна вважати помилкою розробників, але ця помилка не впливає на час створення табелів на невеликих базах, тому цю помилку не виправляли. Час створення табелів зростає поступово з ростом бази даних.

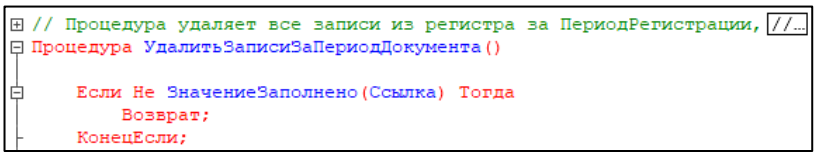


Рисунок 2.13 – Приклад коду

Окрім виправлення цієї помилки, після аналізу запитів по рекомендаціям СУБД, було побудовано індекс до регістра графіку робіт за видом часу, це суттєво пришвидшило формування табелів та вводу графіків (рис. 2.14 та 2.15).

```
/*
Отсутствуют сведения об индексе из v8_5E9D_2d0.sqlplan
Обработчик запросов считает, что реализация следующего индекса может сократить стоимость запроса на 93.2006%.
*/

/*
USE [nec]
GO
CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>]
ON [dbo].[_InfoRg6840] ([_Fld6842RRef],[_Fld6844],[_Fld6845])
INCLUDE ([_Fld6841_TYPE],[_Fld6841_RTRef],[_Fld6841_RRRef],[_Fld6843],[_Fld6847])
GO
*/
```

Рисунок 2.14 – Приклад рекомендації MS SQL серверу на побудову індексів

Модуль	Номер ст...	Строка	Кол.	Врем. (чистое)	%(Врем.) (чистое)
ОбщийМодуль.УниверсальныеМеханизмы.Модуль	786	ВыборкаЗаменяемыхФорм = Запрос.Выполнить().Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);	1	0,593894	75,47
Документ.ТабельУчетаРабочегоВремениОрганизации.Фо...	151	ЭлементыФормы.ВстроеннаяСправка.УстановитьТекст(ТекстМакетаВстроеннойСправки);	1	0,027664	3,52
Документ.ТабельУчетаРабочегоВремениОрганизации.Фо...	1 280	МесяцСтрока = РаботаСДиалогами.ДатаКакМесяцПредставление(ПериодРегистрации);	1	0,027094	3,44
ОбщийМодуль.ФормированиеПечатныхФорм.Модуль	392	ДеревоМакетов = УниверсальныеМеханизмы.ПолучитьДеревоМакетовПечати(ДокументОбъект.Ссылка, СтруктураВнутреннихПечатныхФо...	1	0,020907	2,66
Документ.ТабельУчетаРабочегоВремениОрганизации.Фо...	7 678	ПустойСотрудникОрганизации =Справочники.СотрудникиОрганизаций.ПустаяСсылка();	1	0,016477	2,09
ОбщийМодуль.УправлениеПользователями.Модуль	79	Выборка = Запрос.Выполнить().Выбрать();	3	0,010453	1,33
ОбщийМодуль.УниверсальныеМеханизмы.Модуль	217	Выборка = Запрос.Выполнить().Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);	1	0,007575	0,96
Документ.ТабельУчетаРабочегоВремениОрганизации.Фо...	7 666	Выборка = Запрос.Выполнить().Выбрать();	1	0,006696	0,85
Документ.ТабельУчетаРабочегоВремениОрганизации.Фо...	1 085	Результат = Запрос.Выполнить();	1	0,006453	0,82
ОбщийМодуль.ГлобальныйМодуль.Модуль	298	Выборка = Запрос.Выполнить().Выбрать();	1	0,006393	0,81

Рисунок 2.15 – Приклад виконання заміру продуктивності роботи програми

2.4.2 Обробка помилок при записі в регістри

Не зважаючи на всі дії, для досягнення роботи користувачів без блокування, ми допускаємо можливість цих помилок. Тому всюди, де ми записуємо в регістри, робимо обробку помилок (рис. 2.16). Замість коду програми *nНаборЗаписей.Записать();* визиваємо функцію с загальних модулів *ДляШлюза.ЗаписьНабораЗаписей(НаборОсновныеНачисления);*.

```
//НаборОсновныеНачисления.Записать();
ДляШлюза.ЗаписьНабораЗаписей(НаборОсновныеНачисления);

//НаборЗаписейРабочееВремя.Записать();
ДляШлюза.ЗаписьНабораЗаписей(НаборЗаписейРабочееВремя);
```

Рисунок 2.16 – Приклад програми

В функції, перед записом регістра, вводимо обробку помилки «*попитка*»**,** аналізуємо помилки в разі помилки, робимо паузу на одну секунду і повторюємо запис. Текст функції наведено в додатку Г.

2.4.3 Налаштування регістрів

В усіх регістрах встановлюємо режим розділення підсумків, це впливає на паралельність роботи системи [5].

Для швидкого отримання підсумків регістру на даний момент та інші моменти часу, система підтримує в актуальному стані в окремих таблицях (недоступних безпосередньо для розробника конфігурації) стан підсумків з

урахуванням всіх наявних у таблиці рухів. Записи в цих таблицях автоматично оновлюються під час запису рухів. Ці записи використовуються системою автоматично при зверненні до віртуальних таблиць регістрів для отримання відповідних результатів. Тобто, таблиця підсумків містить суму алгебри рухів (з урахуванням виду рухів) по кожній комбінації вимірювань.

При кожному записі рухів система додає чи віднімає, залежно від виду руху, значення ресурсів рухів із відповідних рядків таблиці результатів. Щоб це зробити, система повинна вважати поточне значення, збільшити чи зменшити його, і записати змінене значення. Зрозуміло, щоб ця операція пройшла коректно, потрібно заблокувати запис, що зчитується, щоб ніхто не міг змінити запис після зчитування.

Механізм поділу підсумків вводить до складу таблиці підсумків спеціальне поле, що дозволяє розпаралелювати оновлення записів підсумків. При отриманні підсумків (звернення до віртуальної таблиці) система згортає записи за комбінаціями вимірювань. Нові записи з вже існуючими комбінаціями вимірювань створюються системою тільки у випадку, якщо паралельно виконуються дві транзакції. Фактично, значення роздільника – це деяке число, що видається системою в черговій транзакції і унікальне серед усіх транзакцій, що паралельно виконуються, в даний момент часу. Система видає перше вільне число, тобто числа, видані транзакціям, що раніше завершилися, перевикористовуються. Таким чином, збільшення кількості записів підсумків залежить від кількості одночасно виконуваних транзакцій.

2.5 Оптимізація запитів до бази даних

2.5.1 Використання вкладених запитів

При використанні складних запитів краще використовувати тимчасові таблиці замість вкладених запитів. Оптимізатору СУБД простіше побудувати

коректний план запиту. Відібравши з великих таблиць потрібні записи, СУБД буде простіше з'єднувати вже відібрані рядки, але буде витрачатися час на створення тимчасових таблиць.

Зробимо два простих запита до таблиць з великою кількістю строк. Для цього запиту виберемо першу таблицю «графіки робіт по видам часу» для отримання планового часу встановленого графіком робіт. Для другої таблиці виберемо «Робочій час працівників організації». Порівнявши по співробітникам та по дням отримаємо звіт по плановому та фактично відпрацьованому часу. В першому запиті з'єднаємо таблиці зразу (рис. 2.17), в другому спочатку потрібні дані збережемо в тимчасові таблиці, а потім ці тимчасові таблиці з'єднаємо (рис. 2.18). Порівняємо результат [4].

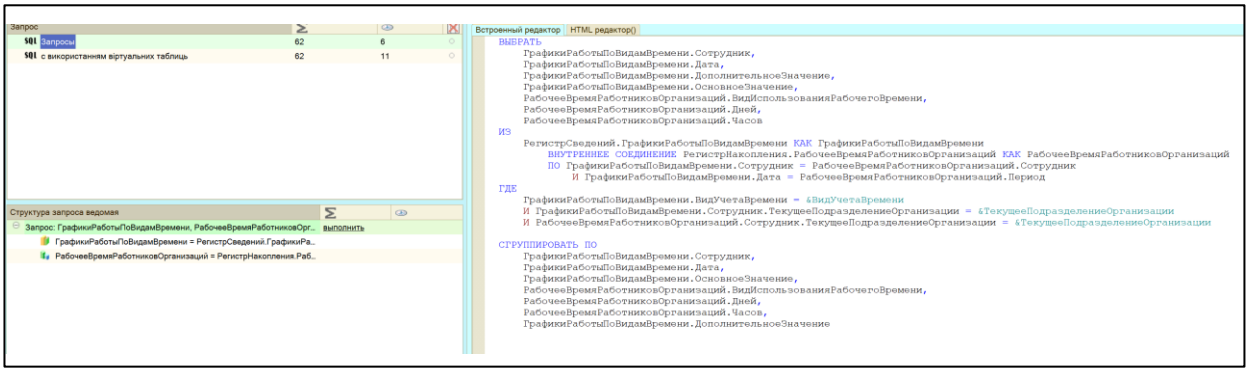


Рисунок 2.17 – Приклад першого запиту

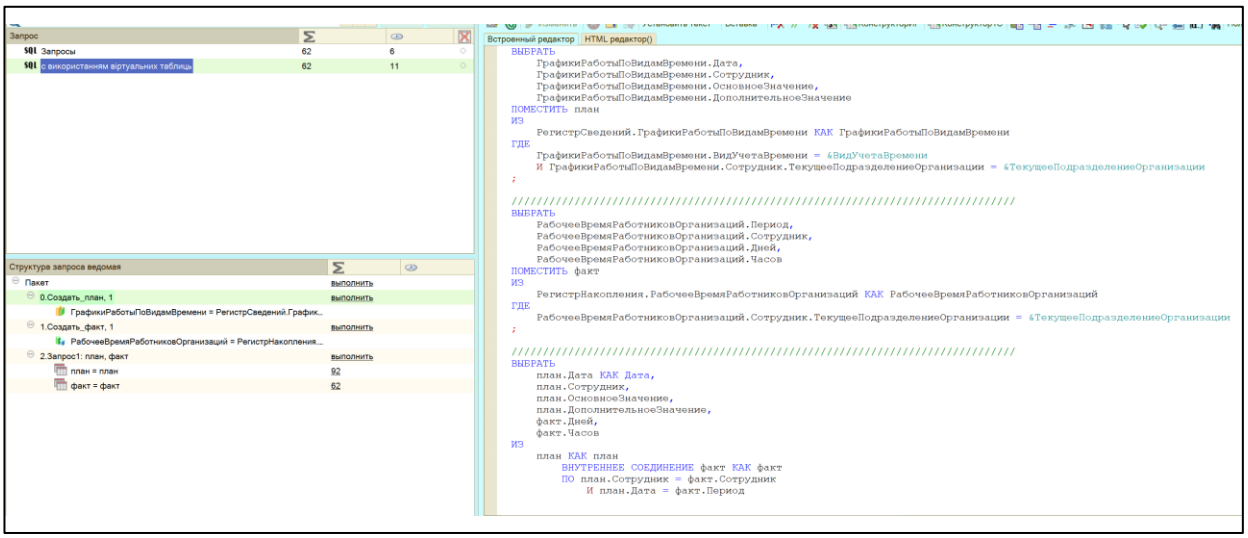


Рисунок 2.18 – Приклад другого запиту

З результатів бачимо, що перший запит було виконано вдвічі швидше ніж другий. Перший запит простіший, не потрібно витрачати час на створення тимчасових таблиць.

2.5.2 Відповідність індексів та порядку полів у SQL запиті

У SQL запитах бажано, щоб індекси відповідали порядку полів у таблиці. Це важливо для оптимізації швидкодії запитів. Якщо створюємо індекс, слід врахувати порядок полів у запиті. Наприклад, як наведено на рисунку 2.19.

```
CREATE INDEX ім'я_індексу ON назва_таблиці (поле1, поле2, поле3);
```

Рисунок 2.19

У цьому прикладі індекс буде створений на полях у вказаному порядку. Коли виконуємо запит, який використовує цей індекс, база даних може швидше знаходити відповідні записи, оскільки дані зберігаються у вказаному порядку. При виборі полів для індексів важливо враховувати частоту їх використання в запитах та їхню унікальність. Наприклад, стовпці, які часто фільтруються або по яких часто виконуються пошукові запити, можуть бути кандидатами для індексування.

2.5.3 Основні принципи роботи в блокування даних сервером додатків

Наведемо ці основні принципи:

- блокуються записи на перехресті значень полів блокування;
- блокується діапазон, а не рядок і не комірка;
- в тому випадку, коли на відбір по якомусь виміру не накладено обмеження, блокуються усі записи по виміру з обмеженням.

Наприклад, перший запит накладає відбір по першому виміру, а другий запит робить відбір по другому виміру та отримуємо очікування на блокуванні. Червоним відмічене заблоковане поле (табл. 2.1). Буде конфлікт блокування.

Таблиця 2.1 – Конфлікт блокування

відділ 1				НЗ.Отбор.Відділ.Установить(«відділ 1»);
відділ 2				НЗ.Отбор.Посада.Установить(«посада 2»);
	посада 1	посада 2	посада 3	

Жовтим відмічена область блокування (табл. 2.2). Очікування на блокуванні буде лише в тому випадку, коли другий запит буде звертатися до заблокованого поля [7].

Таблиця 2.2 – Очікування на блокування

відділ 1				НЗ.Отбор.Відділ.Установить(«відділ 1»); НЗ.Отбор.Посада.Установить(«посада 2»);
відділ 2				
	посада 1	посада 2	посада 3	

2.6 Використання для збереження даних універсального механізму чи окремі таблиці

В усіх сучасних програмах використовується універсальний механізм збереження даних. Цей механізм робить програму універсальною. Без

внесення змін до програм та баз даних є можливість створювати нові реквізити для довідників (рис. 2.20).

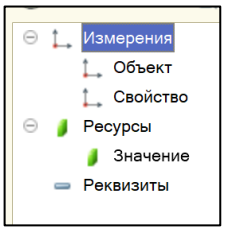


Рисунок 2.20 – Структура регістру значення властивостей об'єктів

В цьому методі додавання реквізитів є недолік. На великих обсягах запити виконуються повільніше. Це пов’язано з тим, що реквізит об’єкт має складний тип. В разі додавання різних об’єктів з великою кількість рядків, швидкість відпрацювання запитів стає довше. Кожний об’єкт – це окрема таблиця. Запиту доводиться сканувати усі таблиці, для пошуку потрібних записів, і пошук не індексований.

Виконаємо тестування, зробимо запити, порівняємо результати.

Перший запит зробимо к індексованій таблиці без з’єднань з іншими таблицями (рис. 2.21).

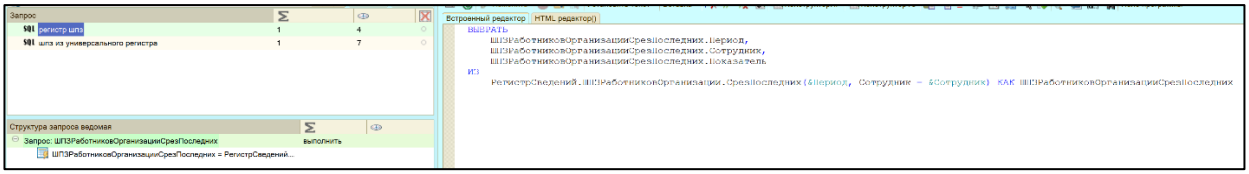


Рисунок 2.21 – Скріншот першого запиту.

Другий запит виконуємо до універсальному регістру (рис. 2.22). В параметрі «Об’єкт» може бути одна чи декілька з 431 таблиць бази даних. При виконані запиту виконуємо з’єднання таблиць (рис. 2.23).

В результаті виконання запитів отримуємо час виконання до універсальної таблиці в декілька разів більший, ніж до окремої проіндексованої таблиці [3].

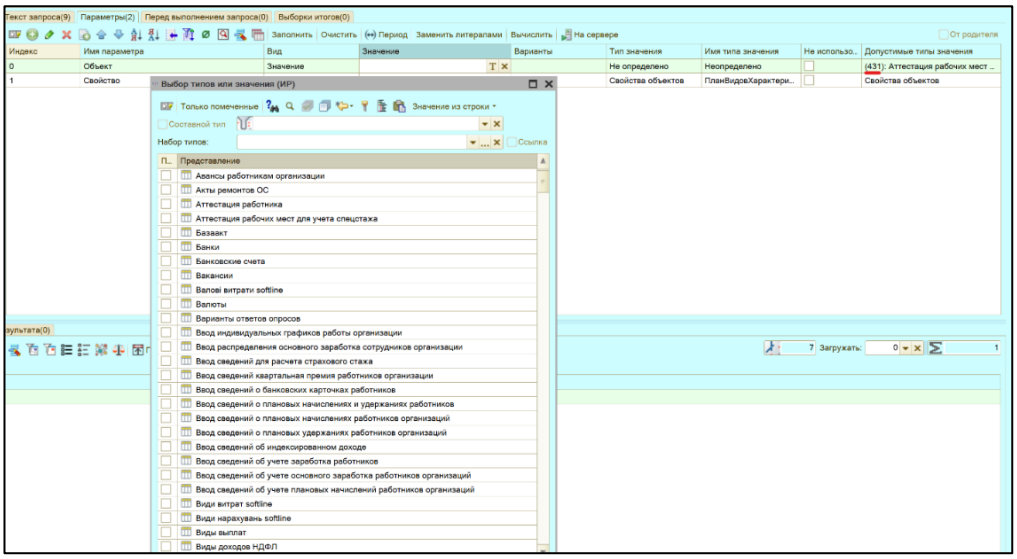


Рисунок 2.22 – Скриншот вибору параметру запиту

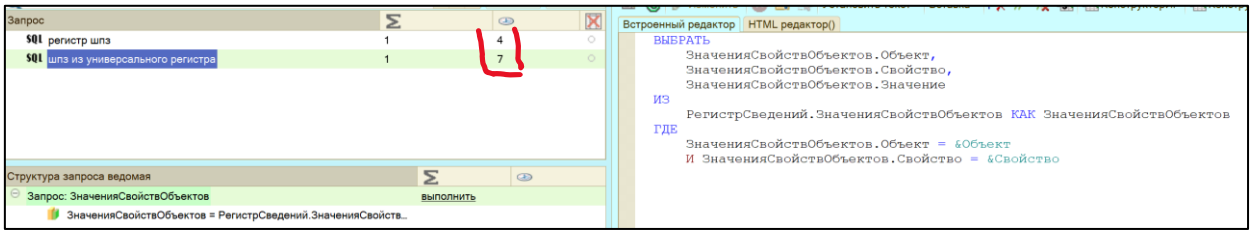


Рисунок 2.23 – Скриншот другого запиту

Це не є критичним для окремого запиту, але в разі виконання запиту в циклі час виконання обробки зросте в декілька разів.

2.7 Оптимізація алгоритмів

Для суттєвого прискорення розрахунків необхідно зменшити кількість серверних викликів, запитів до бази даних. Зазвичай, розрахунки виконуються по рядкам. Кожний параметр до розрахунку отримують окремим запитом. На запит витрачається багато часу. В разі об’єднання запитів, час виконання одного запиту зростає не суттєво. Для документів з великою кількістю рядків можна отримати усі необхідні дані одним запитом, вивантажити результат запиту в таблицю значень, в разі необхідності виконати пост обробку, і

завантажити в табличну частину документу, чи використовувати отримані значення в процесі розрахунку.

Наведемо приклад розрахунку в документі «запланована зарплата» приклад коду в додатку Д. В цьому коді ми спочатку одним запитом отримуємо всі необхідні данні та розрахунки, а потім завантажуємо отримані данні в табличну частину документу. Розрахунок триває декілька секунд. При традиційному розрахунку по рядкам – час розрахунку був би суттєво більше.

3 ТЕСТУВАННЯ ТА АНАЛІЗ

3.1 Аналіз роботи програми з різною кількістю процесорних потоків

З метою тестування роботи програми для імітації роботи користувачів використали програму, яка створює завдання по одночасному запису в декілька потоків.

Завдання виконували на віртуальній машині з різною кількістю процесорних потоків. В результаті тестів видно, що максимальна продуктивність роботи залежить від кількості потоків.

Максимальна продуктивність настає тоді, коли кількість одночасно виконаних робіт дорівнює кількості потоків. В разі перевищення одночасних завдань над кількістю потоків, продуктивність не зростає та може погіршитись (рис. 3.1).

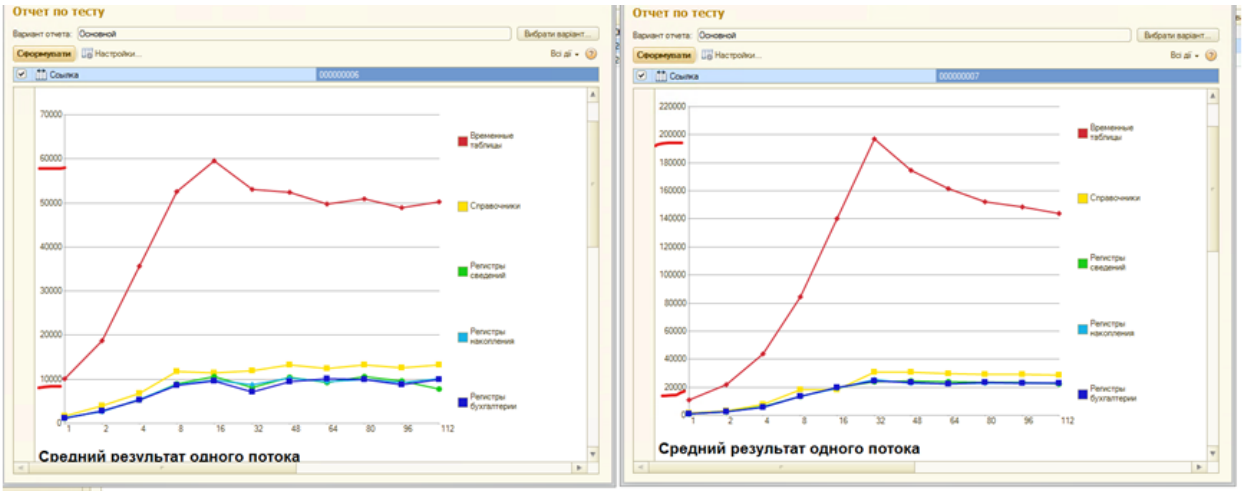


Рисунок 3.1 – Графіки роботи програми з різною кількістю потоків

Після цих тестів видно що на сервері краще збільшити кількість потоків до тої кількості, скільки одночасно повинно виконуватися завдань.

3.2 Приклад паралельного розрахунку документу

Є випадки, коли необхідно розрахувати документ одним користувачем по співробітникам усього підприємства, приблизно 10 000 співробітників, з складним видом розрахунку. Такий розрахунок займає кілька годин. В цьому випадку програма не може задіяти всі ресурси, частина обладнання простоює. Це пов’язано з тим, що розрахунок виконується в один потік. Для усунення цього недоліку є можливість розраховувати документ паралельно.

Для організації багатопоточного обчислення необхідно:

- розділити данні на незалежні блоки;
- кожен потік працює зі своїм блоком даних (в цьому випадку не повинно виникати взаємного блокування та очікування на блокуванні);
- кількість потоків не повинна перевищувати можливостей обчислювальної системи.

Документ розраховується по рядкам та рядки є незалежними. Так, в разі коли на комп’ютері 24 ядра ми маємо можливість розділити рядки документу на 24 частини, і кожну з частину порахувати в окремому потоці. В цьому разі документ розрахується в рази швидше. Приклад такого розрахунку – компенсації відпустки, розраховується після розрахунку заробітної плати, тому не впливає на роботу інших користувачів. Розрахунок виконуємо на сервері додатків, для організації багато поточності використаємо фонові завдання. Текст програми наведено в додатку В.

3.2.1 Тестові обчислення та аналіз результатів

Проведемо розрахунки з різною кількість потоків. Результати обчислень зведемо в таблицю. Звернемо увагу на такі показники, як кількість потоків, час обчислення в секундах, навантаження на процесор. Обчислимо кількість

розрахованих строк (документів) за секунду. На сервері встановлено процесор на 24 потоки. Тому виконаємо обчислення з одного потоку до 36 (табл. 3.1) та побудуємо відповідні графіки залежностей (див. рис. 3.2 та 3.3).

Таблиця 3.1 – Зведена таблиця результатів

Кількість потоків	Кількість розрахованих строк за секунду	Приріст відносно однопоточного розрахунку	Середнє завантаження процесору в %
1	0,26	1,00	5
2	0,62	2,38	10
4	1,35	5,19	20
6	1,99	7,65	25
8	2,93	11,27	37
10	3,40	13,08	44
12	3,87	14,88	50
14	4,21	16,19	60
16	4,40	16,92	65
18	4,55	17,50	70
20	4,66	17,92	77
22	4,88	18,77	85
24	5,05	19,42	90
26	5,15	19,81	92
28	5,20	20,00	93
30	5,27	20,25	95
32	5,35	20,58	97
34	5,42	20,85	98
36	5,48	21,08	100

Рисунок 3.2

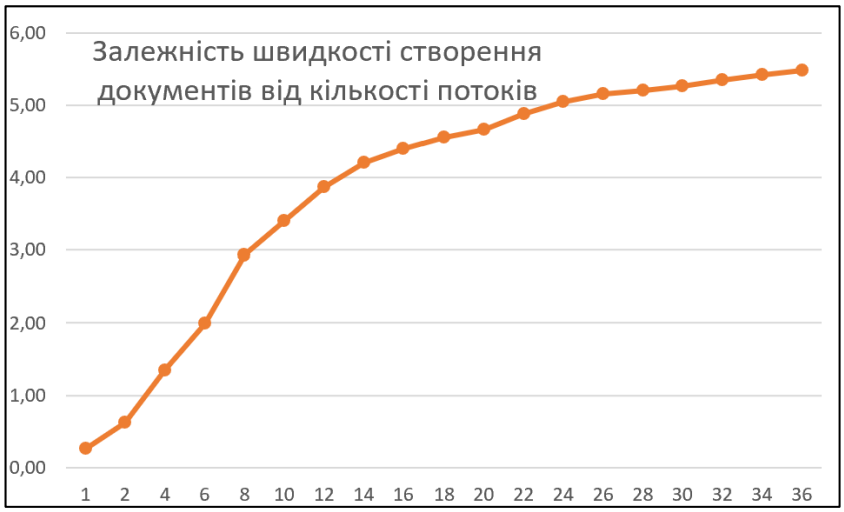


Рисунок 3.3

Розглянемо скриншоти диспетчера завдань при обчисленні з різною кількістю потоків (див. рис. 3.4–3.6).

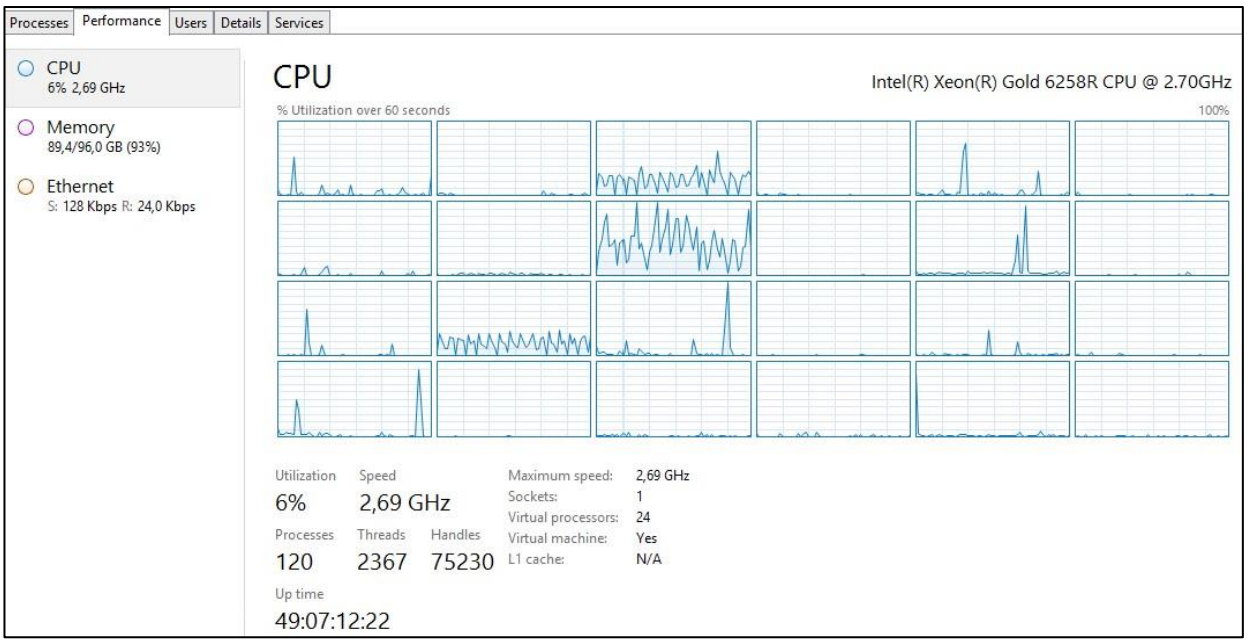


Рисунок 3.4 – Скріншот диспетчера завдань при однопотоковому обчисленні

З скріншоту видно, що обчислення в один потік не може завантажити сервер. Частково завантажені два ядра процесору. Використовуються сервером додатків та SQL сервером.

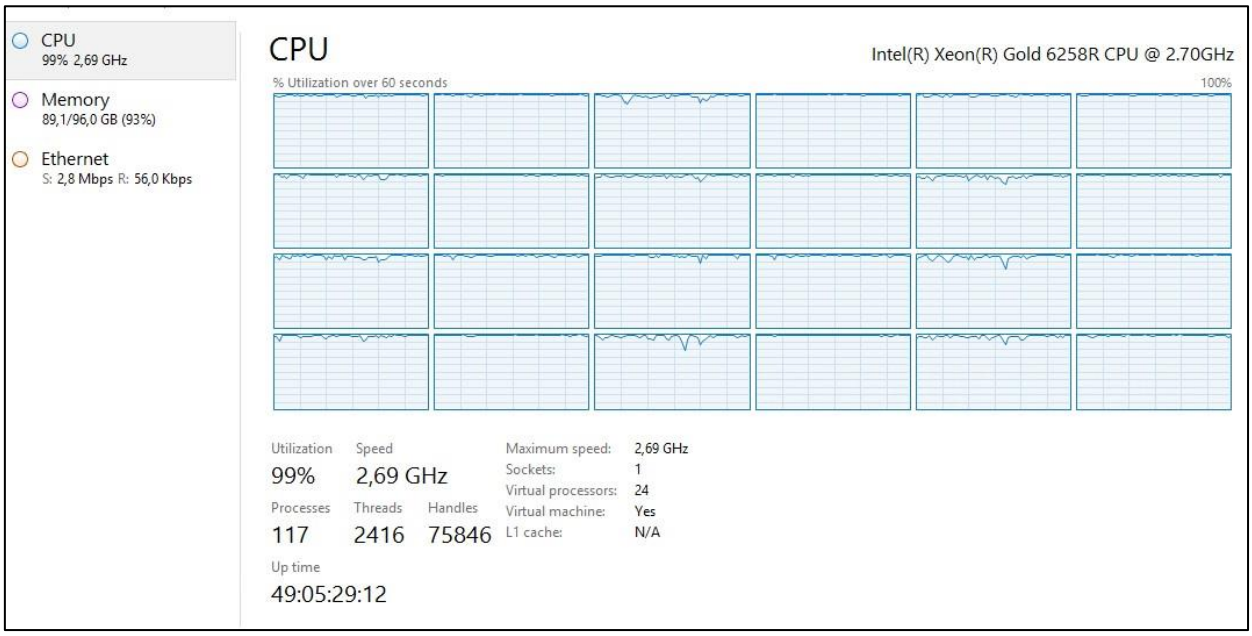


Рисунок 3.5 – Скриншот диспетчера завдань при обчисленні в 36 потоків

При обчислені в 36 потоків на сервері з процесором з 24 ядра завантаження процесору 100 відсотків. Сервер перезавантажено тривалий час. Таке використання серверу не бажано. Сервер не буде мати можливості відповідати на запити інших завдань.

Рисунок 3.6 – Скриншот диспетчера завдань при обчисленні в 12 потоків

Для практичного використання бажано використовувати кількість потоків рівною половині від кількості потоків процесору. Середнє завантаження процесору дорівнює 50 відсотків, а прискорення обчислення отримуємо в 15 разів.

Є можливість для досягнення кращого результату обчислювати з тією кількістю потоків скільки є на процесорі. В тесті використовувався 24 поточний процесор – обчислення в 24 потоки дало приріст швидкості в 20 разів, при навантажені процесору на 90 відсотків.

При використані більшої кількості потоків ми не отримуємо приросту в швидкості обчислені. Тому як процесор повністю завантажено. При необхідності пришвидшити обчислення необхідно збільшувати кількість віртуальних процесорів. На теперішній час зазвичай використовують віртуальні машини, збільшити кількість віртуальних процесорів на віртуальній машині не є проблемою. Змінювати обладнання для цього не має потреби.

3.3 Інтеграція з іншими програмами

Не зважаючи на те, що вся робота була присвячена тому, щоб виконувати всі роботи в одній програмі, в разі коли ми не можемо досягти потрібної швидкості робіт, може виникнути необхідність поділити виконання робіт на частини.

Наприклад, вести кадровий облік в одній програмі а розрахунок заробітної плати в іншій, налаштувавши інтеграцію між програмами для усунення ручного вводу.

Це дає можливість розподілити навантаження на декілька серверів, зменшить кількість помилок блокування, підвищує швидкість, та залишає можливість отримувати звіти по підприємству. Суттєвий недолік в тому, що витрачаєтеся час на синхронізацію баз.

3.4 Організаційні питання

Не завжди є можливість замінити обладнання чи оптимізувати програми, але завжди є можливість правильно організувати роботу з програмою. При розрахунку зарплати важливо виконувати роботу в правильної послідовності. Спочатку вводяться первині документи, потім виконуються розрахунки, після чого отримуємо звіти.

Суттєво зменшити затримки на блокування можливо розділивши у часі введення документів в робочий час, а розрахунок після роботи. У цього підходу є недолік, він полягає в тому, що результат ми отримуємо на наступний день. Знаходимо помилки, виправляємо, а результати знов на наступний день.

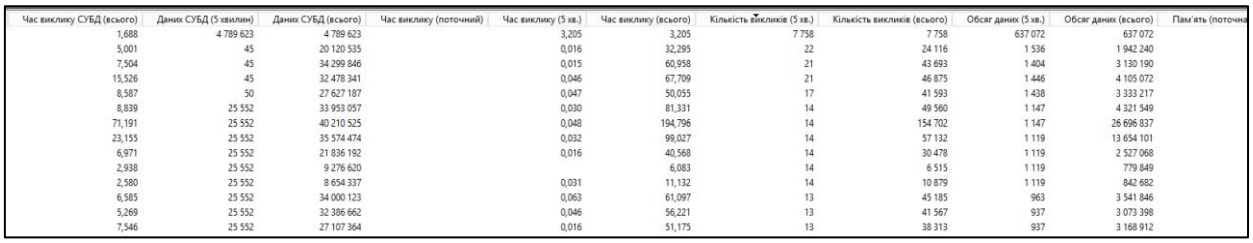
Приклад хибної організації виконання роботи. Працівниця відділу розрахунку заробітної плати вирішила розрахувати заробітну плату усіх працівників підприємства в одному документі. Після виконання розрахунку, з'ясувалося, що виправили помилки в табелі обліку робочого часу, потрібно документ розрахунку перерахувати. Так повторювалось декілька разів. Тому, замість того, щоб розраховувати усе в одному документі, краще створювати документи розрахунку на підставі кадрових документів. Розраховувати в одному документі не більш 300 осіб.

3.5 На що звертати увагу при зниженні швидкості роботи

При виконані аналізу швидкості роботи серверу, в першу чергу звертаємо увагу на консоль кластера сервера (див. рис. 3.7).

В випадку великих значень в стовпчику час виклику (поточний) – потрібно аналізувати код програми. Виконати замір виконання обробки. Виявивши операцію, яка виконується довго.

У випадку великих значень в стовпчику час виклику СУБД (поточний) – не налаштована СУБД.



Час виклику СУБД (всього)	Даних СУБД (5 хвилин)	Даних СУБД (всього)	Час виклику (поточний)	Час виклику (5 хв.)	Час виклику (всього)	Кількість викликів (5 хв.)	Кількість викликів (всього)	Обсяг даних (5 хв.)	Обсяг даних (всього)	Пам'ять (поточна
1,688	4 789 623	4 789 623		3,205	3,205	7 758	7 758	637 072	637 072	
5,001	45	20 120 535		0,016	32,295	22	24 116	1 536	1 942 240	
7,504	45	34 299 846		0,015	60,958	21	43 693	1 404	3 130 190	
15,526	45	32 478 341		0,046	67,709	21	46 875	1 446	4 105 072	
8,587	50	27 627 187		0,047	50,055	17	41 593	1 438	3 333 217	
8,839	25 552	33 953 057		0,030	81,331	14	49 560	1 147	4 321 549	
71,191	25 552	40 210 525		0,048	194,796	14	154 702	1 147	26 696 837	
23,155	25 552	35 574 474		0,032	99,027	14	57 132	1 119	13 654 101	
6,971	25 552	21 836 192		0,016	40,568	14	30 478	1 119	2 527 068	
2,938	25 552	9 276 620			6,083	14	6 515	1 119	779 849	
2,580	25 552	8 654 337		0,031	11,132	14	10 879	1 119	842 682	
6,585	25 552	34 000 123		0,063	61,097	13	45 185	963	3 541 846	
5,269	25 552	32 386 662		0,046	56,221	13	41 567	937	3 073 398	
7,546	25 552	27 107 364		0,016	51,175	13	38 313	937	3 168 912	

Рисунок 3.7 – Приклад скриншоту консолі кластеру серверу

Звертаємо увагу на використання пам’яті – скільки користувач споживає пам’яті. Нормою можна вважати мегабайти. В тому разі, коли використовуються гігабайти, скоріш за все, не вірно написані запити до бази даних чи витік пам’яті. Потрібно враховувати, що багато пам’яті використовується також і при формуванні складних звітів за великий період по всьому підприємству.

Велике використання пам’яті заважає праці інших користувачів – такі сеанси на сервері своєчасно потрібно закривати. Також є можливість в налаштуваннях серверу поставити галку напроти примусово завершувати проблемні процеси.

В разі отримання помилки взаємного блокування, потрібно звернути увагу на стовпчик «заблоковано СУБД» та стовпчик «захоплено СУБД» той сеанс, в якому «захоплено СУБД» буде більше – припинити.

В базах з великою кількістю даних документ може записуватись та проводитись швидко, а затримки отримувати при побудові інтерфейсу, в якому використовується пошук по усім полям динамічного списку. Для уникнення затримок краще відключити пошук по всім полям. Для пошуку потрібної інформації зробити інший механізм, наприклад ввести в інтерфейсі поле пошуку.

На великих базах автоматичний запуск регламентного завдання «повнотекстовий пошук» краще відключити і запускати в не робочий час.

Для того, щоб відстежувати, від якого користувача виконується фонове завдання на сервері додатків, краще для кожного фонового завдання створити окремого користувача і давати назви користувачу по імені фонового завдання.

В цьому випадку буде ясно, яке фонове завдання працює в консолі кластера сервера.

ВИСНОВКИ

На цьому прикладі ми побачили, що на маленьких базах чи при низькому навантажені програма може працювати чудово. При зростанні бази та при великому навантажені, не коректний код програми, чи відсутність індексів в базі може привести до суттєвого зниження швидкості роботи програми. Подібні проблеми можуть виникати в різних програмах.

В цій роботі розглянуті загальні тенденції по усуненню «ручних» робіт для вводу даних, оптимізації роботи програми, підвищення швидкості, зменшення рівня ескалації на блокуванні. Після виконання цих процедур, підвищуються швидкість та збільшення можливостей одночасної роботи користувачів.

За результатами роботи, виявлено, що збільшення швидкості роботи дискової підсистеми у два рази пришвидшує роботу програми в два рази. А вірно розставивши індекси, чи оптимізувавши код програми, ми отримуємо приріст продуктивності в рази. Чим більші розміри бази, тим суттєвіше впливає оптимізація.

Кількість процесорних бажано щоб була не менше ніж планується одночасно працюючих користувачів.

Без постійного нагляду і виконання своєчасних регламентних робот, швидкість може суттєво знижатися, місце на дисках закінчуватися, архіви не створюватися. Це підвищує ризик втрати даних та зменшення продуктивності.

ПЕРЕЛІК ПОСИЛАНЬ

1. Берко А. Ю., Верес О. М., Пасічник В. В. Системи баз даних та знань. Книга 1. Організація баз даних та знань. Львів : «Магнолія-2006», 2008. 456 с.

2. Берко А. Ю., Верес О. М., Пасічник В. В. Системи баз даних та знань. Книга 2. Системи баз даних та знань. Львів : «Магнолія-2006», 2013. 680 с.

3. Гайна Г. А. Основи проектування баз даних : навч. посібник. К.: КНУБА, 2005. 204 с.

4. Мулеса О. Ю. Основи мови запитів SQL. Ужгород, 2015. 48 с.

5. Влаштування та використання режиму поділу підсумків регістрів. URL: https://stimul.kiev.ua/materialy.htm?a=ustroystvo_i_ispolzovanie_rezhima_razdeleniya_itogov_registrov (дата звернення: 03.09.2023).

6. Інформаційна система ITC. URL: https://its.bas-soft.eu/section/dev (дата звернення: 03.09.2023).

7. Business Automation Software. Посібник адміністратора. URL: https://its.bas-soft.eu/db/basadmua (дата звернення: 18.09.2023).

8. Універсальна технологія обміну – конфігурація "Конвертація даних 2.1". Методики роботи. URL: https://its.bas-soft.eu/db/metoddevu/browse/13/-1/9172/9215/9220 (дата звернення: 03.09.2023).

9. Розширювана мова розмітки XML. URL: https://uk.wikipedia.org/wiki/XML (дата звернення: 18.09.2023).

10. Навчальні ресурси з SQL. URL: https://learn.microsoft.com/en-us/sql/sql-server/educational-sql-resources?view=sql-server-ver16 (дата звернення: 18.09.2023).

11. Про затвердження Порядку обчислення середньої заробітної плати. URL: https://zakon.rada.gov.ua/laws/show/100-95-%D0%BF#Text (дата звернення: 05.09.2023).

ДОДАТОК А

Приклад завантаження документів з файлів із папки

```
Процедура ЗагрузитьИзЕксель(Кнопка)
ПерваяСтрока =8;
КолонкаНомерДокумента =1;
КолонкаТабельныйНомер =2;
КолонкаВидВытрат =4;
КолонкаКодПодразделения =5;
КолонкаКодПодразделенияИнвентарногоНомера = 9;

КолонкаИнвентарныйНомер = 6;
КолонкаКоличествоДнейЧасов =7;
КолонкаРазъездной =8;

ДиалогВыбора = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
ДиалогВыбора.Заголовок = "Выберите файл ";

Если ДиалогВыбора.Выбрать() Тогда
ИмяФайла = ДиалогВыбора.ПолноеИмяФайла;
КонецЕсли;

//подключаемся к эксель
Попытка
Excel = Новый COMОбъект("Excel.Application");
Excel.WorkBooks.Open(ИмяФайла);
Состояние("Обработка файла Microsoft Excel...");
Исключение
Сообщить("Ошибка при открытии файла с помощью Excel! Загрузка не будет произведена!");
Сообщить(ОписаниеОшибки());
Возврат;
КонецПопытки;

Попытка
//Открываем необходимый лист
Excel.Sheets(1).Select(); // лист 1, по умолчанию
Исключение
//Закрываем Excel
Excel.ActiveWorkbook.Close();
Excel = 0;
Сообщить("Файл "+Строка(ИмяФайла)+" не соответствует необходимому формату! Первый лист не найден!");
ОтменитьТранзакцию();
Возврат;
КонецПопытки;

//Получим количество строк и колонок.
//В разных версиях Excel получаются по-разному, поэтому сначала определим версию Excel
Версия = Лев(Excel.Version,Найти(Excel.Version,".")-1);
Если Версия = "8" тогда
ФайлСтрок = Excel.Cells.CurrentRegion.Rows.Count;
ФайлКолонок = Макс(Excel.Cells.CurrentRegion.Columns.Count, 13);
Иначе
ФайлСтрок = Excel.Cells(1,1).SpecialCells(11).Row;
ФайлКолонок = Excel.Cells(1,1).SpecialCells(11).Column;
Конецесли;
```

```
Для НС = ПерваяСтрока по ФайлСтрок Цикл // НС указываем с какой строки начинать обработку

Состояние("Файл "+Строка(ИмяФайла)+": Обрабатывается первый лист "+Строка(Формат(?(ФайлСтрок=0,0,((100*НС)/ФайлСтрок)),"ЧЦ=3; ЧДЦ=0"))+" %");

ОбработкаПрерыванияПользователя(); //указав данный оператор, цикл можно прервать в любой момент нажатие ctrl+break

Попытка

если ЗначениеЗаполнено(Excel.Cells(НС, КолонкаНомерДокумента).Text) тогда
//проверяем наличие документа
ссылка=Документы.АктыРемонтовОС.НайтиПоНомеру(Excel.Cells(НС, КолонкаНомерДокумента).Text);
если  ссылка.Пустая() тогда
//для нового документа заполняем шапку
ссылка=Документы.АктыРемонтовОС.СоздатьДокумент();
ссылка.Дата = ОбщегоНазначения.ПолучитьРабочуюДату();//ТекущаяДата();
ссылка.МесяцНачисления =    НачалоМесяца(ТекущаяДата());
ссылка.ПериодРегистрации =  НачалоМесяца(ТекущаяДата());
ссылка.Номер = Excel.Cells(НС, КолонкаНомерДокумента).Text;
ссылка.ИнвентарныйНомер  = Excel.Cells(НС, КолонкаИнвентарныйНомер).Text;
ссылка.ВидВитрат=Справочники.ВидиВитратSoftline.НайтиПоКоду(Прав(Excel.Cells(НС,
КолонкаВидВытрат).Text,2));
ссылка.ШифрРабот= Справочники.ШифрРаботSoftline.НайтиПоКоду("110");
ссылка.КодПодразделенияИнвНомера  =  Справочники.ПодразделенияSoftline.НайтиПоКоду(Excel.Cells(НС,
КолонкаКодПодразделенияИнвентарногоНомера).Text);//Справочники.ПодразделенияSoftline.НайтиПоНаим
енованию(Excel.Cells(НС, КолонкаКодПодразделения).Text);
ссылка.Организация                                                                                    =
Справочники.Организации.НайтиПоНаименованию(ПолучитьОсновнаяОрганизация());//Справочники.Орган
изации.НайтиПоКоду("05");
ссылка.Ответственный = ПараметрыСеанса.ТекущийПользователь;
если ЗначениеЗаполнено(ПараметрыСеанса.ТерриториальноеУправление) тогда
ссылка.ТерриториальноеУправление = ПараметрыСеанса.ТерриториальноеУправление[0];
КонецЕсли;
ссылка.Записать();
КонецЕсли;
ссылка = ссылка.ссылка.ПолучитьОбъект();
//заполняем табличную часть
НоваяСтрока                    =                    ссылка.Работники.Найти(НайтиСотрудника(Excel.Cells(НС,
КолонкаТабельныйНомер).Text),"");
если не ЗначениеЗаполнено(НоваяСтрока) тогда
НоваяСтрока = ссылка.Работники.Добавить();
КонецЕсли;

НоваяСтрока.Сотрудник          = НайтиСотрудника(Excel.Cells(НС, КолонкаТабельныйНомер).Text);
НоваяСтрока.ТабНомер  = НайтиСотрудника(Excel.Cells(НС, КолонкаТабельныйНомер).Text).код;
НоваяСтрока.КоличествоДнейЧасов       = Excel.Cells(НС, КолонкаКоличествоДнейЧасов).Text;

НоваяСтрока.ПодразделениеBSoftline                =
ПолучитьКодПодразделенияСофтлайн(НоваяСтрока.Сотрудник, ссылка);
НоваяСтрока.ШПЗ                                   =                ДляШлюза.ПолучитьШПЗ(ссылка.ПериодРегистрации,
НоваяСтрока.Сотрудник);
НоваяСтрока.Инвалидность       = ПолучитьИнвалидность(НоваяСтрока.Сотрудник);
НоваяСтрока.ДниРазъездн        = (Excel.Cells(НС, КолонкаРазъездной).Text);
НоваяСтрока.ТерриториальноеУправление = НоваяСтрока.Сотрудник.ТерриториальноеУправление;

ссылка.Записать();

КонецЕсли;

Исключение
Сообщить("Докумен " + Excel.Cells(НС, КолонкаНомерДокумента).Text +" табельный номер " +
Excel.Cells(НС, КолонкаТабельныйНомер).Text + " не найден");
```

```
Продолжить;
КонецПопытки;

КонецЦикла;

Excel.DisplayAlerts = 0;
Excel.Quit();
//Excel.DisplayAlerts = 1;

КонецПроцедуры
```

ДОДАТОК Б

Приклад заповнення документу методом скопіювати-вставити

```
Процедура ЗапонитьОчередьСБуфера()
	ДанныеБуфера = ПолучитьТекстИзБуфераОбмена();

	Если СокрЛП(ДанныеБуфера) = "" Тогда
		Сообщить("Буфер пуст!!!");
		Возврат;
	КонецЕсли;
	НайденыйСимвол = Найти(ДанныеБуфера,""+Символы.Таб+""+Символы.Таб);

	МассивСтрок = ОбщегоНазначения.РазложитьСтрокуВМассивПодстрок(ДанныеБуфера,
Символы.ПС);

	Для Каждого СтрокаБуфера из МассивСтрок Цикл
		Если СокрЛП(СтрокаБуфера) = "" Тогда
			Продолжить;
		 КонецЕсли;

		 МассивПодСтрок =
ОбщегоНазначения.РазложитьСтрокуВМассивПодстрок(СтрокаБуфера, Символы.Таб);

		если
ЗначениеЗаполнено(ОтработанноеВремя.Найти((НайтиСотрудника(МассивПодСтрок[0])), "Сотрудник"))
тогда	//ищем строку по сотруднику

	ОтработанноеВремя.Удалить(ОтработанноеВремя.Найти(НайтиСотрудника(МассивПодСтрок[0]),
"Сотрудник")); //удаляем найденую строку по сотруднику
		КонецЕсли;

	ЗаполняемаяТаблица=ЭтаФорма.ЭлементыФормы.ОтработанноеВремя;
	ЗаполняемаяТаблица.ДобавитьСтроку();
	НовСтрока = ЗаполняемаяТаблица.ТекущаяСтрока;

НовСтрока.Сотрудник=НайтиСотрудника(МассивПодСтрок[0]);
НовСтрока.Назначение=НовСтрока.Сотрудник;

		Сч = 0;
```

```
                Для Каждого ЗначениеКолонки Из  МассивПодСтрок Цикл
                        если Сч = 0 тогда
                                Сч = Сч+1;
                                Продолжить КонецЕсли;
                                Попытка

        ОбработкаВводаДанныхВЯчейку(ЭтаФорма.ЭлементыФормы.ОтработанноеВремя.Колонки["День"
+Сч].ЭлементУправления, МассивПодСтрок[Сч], , истина, Сч);

                                Исключение
                                КонецПопытки;
                                Сч = Сч+1;
                КонецЦикла;
        КонецЦикла;
КонецПроцедуры
```

ДОДАТОК В

Приклад розрахунку компенсації відпустки в декілька потоків

```
Перем КоличествоПотоков;

Процедура Инициализировать(Объект, ИмяТабличнойЧасти, ТабличноеПоле) Экспорт;
ВремяНачалаВыполнения=ТекущаяДата();
//Объект.Записать(РежимЗаписиДокумента.Запись);

РазмерПорции=0;
КодГраницыТекущегоПотока=0;
КодГраницыПредыдущегоПотока=0;
ЭтоПоследнийПоток=Ложь;

КоличествоСтрокВТаблице=Объект.ИспользованиеЕжегодногоОтпуска.Количество();

КоличествоПотоков=24;

Если ВвестиЧисло(КоличествоПотоков, "Введите число") Тогда
//Сообщить(«Введенное значение: «+КоличествоПотоков);
Иначе
Сообщить("Значение не было введено, по умолчанию 24");
КоличествоПотоков=24;
КонецЕсли;
РазмерПорции=Цел(КоличествоСтрокВТаблице/КоличествоПотоков);

для НомераПотока=1 по КоличествоПотоков цикл
        //определяем порции данных для обработки
        КодГраницыПредыдущегоПотока        =РазмерПорции* (НомераПотока-1)+1;
        КодГраницыТекущегоПотока=РазмерПорции* НомераПотока;
        если (НомераПотока =КоличествоПотоков) тогда
                //если последний поток - обрабатываются все оставшиеся данные
                //т.к. число потоков не кратно количеству сторок
                КодГраницыТекущегоПотока =  КоличествоСтрокВТаблице;
                ЭтоПоследнийПоток=Истина;
        КонецЕсли;

        НаборПараметров = Новый Массив;
        НаборПараметров.Добавить(Объект.Ссылка);
```

```
        //НаборПараметров.Добавить(РазмерПорции);
        НаборПараметров.Добавить(КодГраницыПредыдущегоПотока);
        НаборПараметров.Добавить(КодГраницыТекущегоПотока);
        НаборПараметров.Добавить(ЭтоПоследнийПоток);

        //запуск фоновых заданий
Задание
=ФоновыеЗадания.Выполнить("ОбработчикиФоновыхЗаданий.СозданиеДокументовНачисленияОтпуска",
                НаборПараметров, Новый УникальныйИдентификатор, "Создание документов начисления
отпуска");
        //масив где будут хранится фоновые задания
        МассивЗаданий = Новый Массив;
        МассивЗаданий.Добавить(Задание);

КонецЦикла;
        //проверяем резусльтат выполнения фонового задания
        если МассивЗаданий.Количество()>0 тогда
                Попытка
                        ОбработчикиФоновыхЗаданий.ОжидатьЗавершения(МассивЗаданий);
                Исключение
                КонецПопытки;
        КонецЕсли;

//      Сообщить("Задание Выполнено");

Сообщить("Время выполниения " + Окр((ТекущаяДата()-ВремяНачалаВыполнения)/60,2) +" мин.") ;
КонецПроцедуры
```

ДОДАТОК Г

Приклад запису набора записів в регістри

```
Процедура ЗаписьНабораЗаписей(пНаборЗаписей, пПараметрыЗаписи = Неопределено,
пПаузаСколькоСекунд = 1, пКоличествоИтераций = 10 ) Экспорт

//Записать(<Замещать>, <ТолькоЗапись>, <ЗаписьФактическогоПериодаДействия>, <ЗаписьПерерасчетов>)
        лЗаписали = Ложь;
        Для инд = 1 по пКоличествоИтераций Цикл

            Попытка
                Если пПараметрыЗаписи = Неопределено Тогда

                    пНаборЗаписей.Записать();
                    лЗаписали = Истина;
                    Прервать;

                Иначе

                    Замещать                    = Истина;
                    ТолькоЗапись                = Ложь;
                    ЗаписьФактическогоПериодаДействия = Истина;
                    ЗаписьПерерасчетов          = Истина;

                    Если пПараметрыЗаписи.Свойство("Замещать") Тогда
                        Замещать = пПараметрыЗаписи.Замещать;
                    КонецЕсли;

                    Если пПараметрыЗаписи.Свойство("ТолькоЗапись") Тогда
                        ТолькоЗапись = пПараметрыЗаписи.ТолькоЗапись;
                    КонецЕсли;

                    Если
пПараметрыЗаписи.Свойство("ЗаписьФактическогоПериодаДействия") Тогда
                        ЗаписьФактическогоПериодаДействия =
пПараметрыЗаписи.ЗаписьФактическогоПериодаДействия;
                    КонецЕсли;

                    Если пПараметрыЗаписи.Свойство("ЗаписьПерерасчетов") Тогда
```

```
                                    ЗаписьПерерасчетов = пПараметрыЗаписи.ЗаписьПерерасчетов;
                                КонецЕсли;

                                пНаборЗаписей.Записать(Замещать, ТолькоЗапись,
ЗаписьФактическогоПериодаДействия, ЗаписьПерерасчетов );
                                лЗаписали = Истина;
                                Прервать;

                            КонецЕсли;

                        Исключение
                            Пауза(пПаузаСколькоСекунд);
                        КонецПопытки;

                КонецЦикла;

                Если Не лЗаписали Тогда
                    ВызватьИсключение "Не смогли записать необходимый регистр. Обработка документа
завершена неудачей";
                КонецЕсли;

КонецПроцедуры
```

ДОДАТОК Д

Розрахунок планової заробітної плати одним запитом

```
&НаСервере
Процедура ЗаполнитьНаСервере()
запрос= новый Запрос;
запрос.Текст= "ВЫБРАТЬ ПЕРВЫЕ 9
	|	a.Порог КАК Порог1,
	|	a.Ставка КАК СтавкаПром
	|ПОМЕСТИТЬ b
	|ИЗ
	|	РегистрСведений.ШкалаВыслугиЛет.СрезПоследних КАК a
	|ГДЕ
	|	a.ВидСтажа = &ВидСтажаПром
	|
	|УПОРЯДОЧИТЬ ПО
	|	Порог1
	|;
	|
	|////////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ ПЕРВЫЕ 9
	|	q.Порог,
	|	q.Ставка КАК СтавкаПром
	|ПОМЕСТИТЬ a
	|ИЗ
	|	РегистрСведений.ШкалаВыслугиЛет.СрезПоследних КАК q
	|ГДЕ
	|	q.ВидСтажа = &ВидСтажаПром
	|
	|УПОРЯДОЧИТЬ ПО
	|	q.Порог
	|;
	|
	|////////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ ПЕРВЫЕ 9
	|	a.Порог КАК a,
	|	КОЛИЧЕСТВО(РАЗЛИЧНЫЕ b.Порог1) КАК b,
	|	a.СтавкаПром
	|ПОМЕСТИТЬ c
	|ИЗ
	|	a КАК a
	|		ВНУТРЕННЕЕ СОЕДИНЕНИЕ b КАК b
	|		ПО a.Порог >= b.Порог1
	|
	|СГРУППИРОВАТЬ ПО
	|	a.Порог,
	|	a.СтавкаПром
	|
	|УПОРЯДОЧИТЬ ПО
	|	a.Порог
	|;
	|
	|////////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|	d.b КАК ПорядковыйНомер,
	|	d.a КАК Начало,
	|	c.a КАК Конец,
```

```
|           c.СтавкаПром
|ПОМЕСТИТЬ ШкалаПром
|ИЗ
|           c КАК d
|                       ВНУТРЕННЕЕ СОЕДИНЕНИЕ c КАК c
|                       ПО (d.b = c.b - 1)
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ ПЕРВЫЕ 9
|           a.Порог КАК Порог1,
|           a.Ставка КАК СтавкаНеПром
|ПОМЕСТИТЬ bn
|ИЗ
|           РегистрСведений.ШкалаВыслугиЛет.СрезПоследних КАК a
|ГДЕ
|           a.ВидСтажа = &ВидСтажаНеПром
|
|УПОРЯДОЧИТЬ ПО
|           Порог1
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ ПЕРВЫЕ 9
|           q.Порог,
|           q.Ставка КАК СтавкаНеПром
|ПОМЕСТИТЬ an
|ИЗ
|           РегистрСведений.ШкалаВыслугиЛет.СрезПоследних КАК q
|ГДЕ
|           q.ВидСтажа = &ВидСтажаНеПром
|
|УПОРЯДОЧИТЬ ПО
|           q.Порог
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ ПЕРВЫЕ 9
|           an.Порог КАК an,
|           КОЛИЧЕСТВО(РАЗЛИЧНЫЕ bn.Порог1) КАК bn,
|           an.СтавкаНеПром
|ПОМЕСТИТЬ cn
|ИЗ
|           an КАК an
|                       ВНУТРЕННЕЕ СОЕДИНЕНИЕ bn КАК bn
|                       ПО an.Порог >= bn.Порог1
|
|СГРУППИРОВАТЬ ПО
|           an.Порог,
|           an.СтавкаНеПром
|
|УПОРЯДОЧИТЬ ПО
|           an.Порог
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|           cn.bn КАК ПорядковыйНомер,
|           cn.an КАК Начало,
|           dn.an КАК Конец,
|           dn.СтавкаНеПром
|ПОМЕСТИТЬ ШкалаНеПром
|ИЗ
```

```
	|		cn КАК cn
	|			ВНУТРЕННЕЕ СОЕДИНЕНИЕ cn КАК dn
	|			ПО (cn.bn = dn.bn - 1)
	|;
	|
	|////////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|		ШкалаНеПром.ПорядковыйНомер,
	|		ШкалаНеПром.Начало,
	|		ШкалаНеПром.Конец,
	|		ШкалаНеПром.СтавкаНеПром,
	|		ШкалаПром.СтавкаПром
	|ПОМЕСТИТЬ Шкала
	|ИЗ
	|		ШкалаПром КАК ШкалаПром
	|			ВНУТРЕННЕЕ СОЕДИНЕНИЕ ШкалаНеПром КАК ШкалаНеПром
	|			ПО ШкалаПром.ПорядковыйНомер = ШкалаНеПром.ПорядковыйНомер
	|;
	|
	|////////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|		ШПЗРаботниковОрганизацииСрезПоследних.Период,
	|		ШПЗРаботниковОрганизацииСрезПоследних.Регистратор,
	|		ШПЗРаботниковОрганизацииСрезПоследних.НомерСтроки,
	|		ШПЗРаботниковОрганизацииСрезПоследних.Активность,
	|		ШПЗРаботниковОрганизацииСрезПоследних.Организация,
	|		ШПЗРаботниковОрганизацииСрезПоследних.Сотрудник,
	|		ШПЗРаботниковОрганизацииСрезПоследних.Показатель КАК Показатель,
	|		ШПЗРаботниковОрганизацииСрезПоследних.Действие
	|ПОМЕСТИТЬ ШПЗ
	|ИЗ
	|		РегистрСведений.ШПЗРаботниковОрганизации.СрезПоследних(&Период, ) КАК
ШПЗРаботниковОрганизацииСрезПоследних
	|;
	|
	|////////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|		РаботникиОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
	|		РаботникиОрганизацийСрезПоследних.Сотрудник.Код КАК ТабельныйНомер,
	|		РаботникиОрганизацийСрезПоследних.Сотрудник.ТекущееПодразделениеОрганизации
КАК Подразделение,
	|		РаботникиОрганизацийСрезПоследних.Сотрудник.ТекущаяДолжностьОрганизации КАК
Должность,
	|		ШтатноеРасписаниеОрганизацийСрезПоследних.МинимальнаяТарифнаяСтавка /
&Коэфициент КАК РасчетныйКоэфициентМин,
	|		ШтатноеРасписаниеОрганизацийСрезПоследних.МаксимальнаяТарифнаяСтавка /
&Коэфициент КАК РасчетныйКоэфициентМакс,
	|		ШтатноеРасписаниеОрганизацийСрезПоследних.МинимальнаяТарифнаяСтавка КАК
ВилкаМин,
	|		ШтатноеРасписаниеОрганизацийСрезПоследних.МаксимальнаяТарифнаяСтавка КАК
ВилкаМакс,
	|		(ШтатноеРасписаниеОрганизацийСрезПоследних.МаксимальнаяТарифнаяСтавка +
ШтатноеРасписаниеОрганизацийСрезПоследних.МинимальнаяТарифнаяСтавка) / 2 КАК ВилкаСредняя,
	|		ШПЗ.Показатель КАК ШПЗ
	|ПОМЕСТИТЬ Работники
	|ИЗ
	|		РегистрСведений.РаботникиОрганизаций.СрезПоследних(&Период, ) КАК
РаботникиОрганизацийСрезПоследних
	|			ЛЕВОЕ СОЕДИНЕНИЕ
РегистрСведений.ШтатноеРасписаниеОрганизаций.СрезПоследних(&Период, ) КАК
ШтатноеРасписаниеОрганизацийСрезПоследних
```

```
        |                    ПО
РаботникиОрганизацийСрезПоследних.Сотрудник.ТекущаяДолжностьОрганизации =
ШтатноеРасписаниеОрганизацийСрезПоследних.Должность
        |                        И
РаботникиОрганизацийСрезПоследних.Сотрудник.ТекущееПодразделениеОрганизации =
ШтатноеРасписаниеОрганизацийСрезПоследних.ПодразделениеОрганизации
        |                    ЛЕВОЕ СОЕДИНЕНИЕ ШПЗ КАК ШПЗ
        |                    ПО РаботникиОрганизацийСрезПоследних.Сотрудник = ШПЗ.Сотрудник
        |ГДЕ
        |    РаботникиОрганизацийСрезПоследних.ПричинаИзмененияСостояния <>
ЗНАЧЕНИЕ(перечисление.ПричиныИзмененияСостояния.Увольнение)
        |    И РаботникиОрганизацийСрезПоследних.Организация = &Организация
        |
        |ИНДЕКСИРОВАТЬ ПО
        |    Сотрудник
        |;
        |
        |////////////////////////////////////////////////////////////////////////////
        |ВЫБРАТЬ
        |    ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
        |    ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
        |ПОМЕСТИТЬ Оклад
        |ИЗ
        |    РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
        |ГДЕ
        |    (ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
ЗНАЧЕНИЕ(ПланВидовРасчета.ОсновныеНачисленияОрганизаций.ОкладПоДням)
        |                    ИЛИ
ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
ЗНАЧЕНИЕ(ПланВидовРасчета.ОсновныеНачисленияОрганизаций.ОкладПоЧасам))
        |
        |ИНДЕКСИРОВАТЬ ПО
        |    Сотрудник
        |;
        |
        |////////////////////////////////////////////////////////////////////////////
        |ВЫБРАТЬ
        |    ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
        |    ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
        |ПОМЕСТИТЬ Премия
        |ИЗ
        |    РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
        |ГДЕ
        |    ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
ЗНАЧЕНИЕ(ПланВидовРасчета.ОсновныеНачисленияОрганизаций.ПремияТек)
        |
        |ИНДЕКСИРОВАТЬ ПО
        |    Сотрудник
        |;
        |
        |////////////////////////////////////////////////////////////////////////////
        |ВЫБРАТЬ
        |    КвартальнаяПремияРаботниковОрганизацииСрезПоследних.Сотрудник КАК Сотрудник,
        |    КвартальнаяПремияРаботниковОрганизацииСрезПоследних.Показатель
        |ПОМЕСТИТЬ КвартальнаяПремия
        |ИЗ
        |    РегистрСведений.КвартальнаяПремияРаботниковОрганизации.СрезПоследних(&Период, )
КАК КвартальнаяПремияРаботниковОрганизацииСрезПоследних
        |
```

```
	|ИНДЕКСИРОВАТЬ ПО
	|	Сотрудник
	|;
	|
	|////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
	|	ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
	|ПОМЕСТИТЬ ОВД
	|ИЗ
	|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
	|ГДЕ
	|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаОВД
	|
	|ИНДЕКСИРОВАТЬ ПО
	|	Сотрудник
	|;
	|
	|////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
	|	ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
	|ПОМЕСТИТЬ Класность
	|ИЗ
	|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
	|ГДЕ
	|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаКласность
	|
	|ИНДЕКСИРОВАТЬ ПО
	|	Сотрудник
	|;
	|
	|////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
	|	ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
	|ПОМЕСТИТЬ Вредность
	|ИЗ
	|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
	|ГДЕ
	|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаВредность
	|
	|ИНДЕКСИРОВАТЬ ПО
	|	Сотрудник
	|;
	|
	|////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|
	ПроцентОплатыЗаРаботуСВреднымиУсловиямиТрудаСрезПоследних.СотрудникиОрганизаций,
	|
	ПроцентОплатыЗаРаботуСВреднымиУсловиямиТрудаСрезПоследних.ПроцентОплатыЗаРаботуСВр
едными УсловиямиТруда
	|ПОМЕСТИТЬ Вредность2
```

```
|ИЗ
|
РегистрСведений.ПроцентОплатыЗаРаботуСВреднымиУсловиямиТруда.СрезПоследних(&Период, )
КАК ПроцентОплатыЗаРаботуСВреднымиУсловиямиТрудаСрезПоследних
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
|	ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
|ПОМЕСТИТЬ ВДП
|ИЗ
|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
|ГДЕ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаВДП
|
|ИНДЕКСИРОВАТЬ ПО
|	Сотрудник
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
|	ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
|ПОМЕСТИТЬ Совмещение
|ИЗ
|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
|ГДЕ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаСовмещение
|
|ИНДЕКСИРОВАТЬ ПО
|	Сотрудник
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
|
СУММА(ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1,
0)) КАК Показатель1
|ПОМЕСТИТЬ РЗО_ЗОР
|ИЗ
|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
|ГДЕ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета
В(&ВидРасчетаРЗО_ЗОР)
|
|СГРУППИРОВАТЬ ПО
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник
|
|ИНДЕКСИРОВАТЬ ПО
|	Сотрудник
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
```

```
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
|
СУММА(ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1,
0)) КАК Показатель1
|ПОМЕСТИТЬ ВО
|ИЗ
|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
|ГДЕ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета
В(&ВидРасчетаВО)
|
|СГРУППИРОВАТЬ ПО
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник
|
|ИНДЕКСИРОВАТЬ ПО
|	Сотрудник
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
|
СУММА(ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1,
0)) КАК Показатель1
|ПОМЕСТИТЬ КБ
|ИЗ
|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
|ГДЕ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета
В(&ВидРасчетаКБ)
|
|СГРУППИРОВАТЬ ПО
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник
|
|ИНДЕКСИРОВАТЬ ПО
|	Сотрудник
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
|
СУММА(ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1,
0)) КАК Показатель1
|ПОМЕСТИТЬ КТ
|ИЗ
|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
|ГДЕ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета
В(&ВидРасчетаКТ)
|
|СГРУППИРОВАТЬ ПО
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник
|
|ИНДЕКСИРОВАТЬ ПО
|	Сотрудник
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
```

```
        |        ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
        |        ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
        |ПОМЕСТИТЬ Секретность
        |ИЗ
        |        РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
        |ГДЕ
        |        ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаСекретность
        |
        |ИНДЕКСИРОВАТЬ ПО
        |        Сотрудник
        |;
        |
        |////////////////////////////////////////////////////////////////////////////////
        |ВЫБРАТЬ
        |        ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
        |        ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
        |ПОМЕСТИТЬ Звание
        |ИЗ
        |        РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
        |ГДЕ
        |        ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаЗвание
        |
        |ИНДЕКСИРОВАТЬ ПО
        |        Сотрудник
        |;
        |
        |////////////////////////////////////////////////////////////////////////////////
        |ВЫБРАТЬ
        |        ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
        |        ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
        |ПОМЕСТИТЬ ДоплатаЗаРоздРобот
        |ИЗ
        |        РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
        |ГДЕ
        |        ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаДоплатаЗаРоздРобот
        |
        |ИНДЕКСИРОВАТЬ ПО
        |        Сотрудник
        |;
        |
        |////////////////////////////////////////////////////////////////////////////////
        |ВЫБРАТЬ
        |        ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
        |        ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
        |ПОМЕСТИТЬ Ненормированый
        |ИЗ
        |        РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
        |ГДЕ
        |        ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаНенормированый
        |
        |ИНДЕКСИРОВАТЬ ПО
```

```
|	Сотрудник
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
|	ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
|ПОМЕСТИТЬ Персональная
|ИЗ
|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
|ГДЕ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаПерсональная
|
|ИНДЕКСИРОВАТЬ ПО
|	Сотрудник
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ШкалаВыслугиЛетСрезПоследних.Период,
|	ШкалаВыслугиЛетСрезПоследних.ВидСтажа,
|	ШкалаВыслугиЛетСрезПоследних.НомерПорога,
|	ШкалаВыслугиЛетСрезПоследних.УдалитьСпособУчетаСтажа,
|	ШкалаВыслугиЛетСрезПоследних.Ставка,
|	ШкалаВыслугиЛетСрезПоследних.Порог
|ПОМЕСТИТЬ ставки
|ИЗ
|	РегистрСведений.ШкалаВыслугиЛет.СрезПоследних(&Период, ) КАК
ШкалаВыслугиЛетСрезПоследних
|ГДЕ
|	ШкалаВыслугиЛетСрезПоследних.ВидСтажа В(&ВидСтажа)
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Сотрудник КАК Сотрудник,
|	ЕСТЬNULL(ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.Показатель1, 0)
КАК Показатель1
|ПОМЕСТИТЬ НаучнаяСтепень
|ИЗ
|	РегистрСведений.ПлановыеНачисленияРаботниковОрганизаций.СрезПоследних(&Период,
) КАК ПлановыеНачисленияРаботниковОрганизацийСрезПоследних
|ГДЕ
|	ПлановыеНачисленияРаботниковОрганизацийСрезПоследних.ВидРасчета =
&ВидРасчетаНаучнаяСтепень
|
|ИНДЕКСИРОВАТЬ ПО
|	Сотрудник
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	РаботникиОрганизацийСрезПоследних.Сотрудник,
|	РаботникиОрганизацийСрезПоследних.ГрафикРаботы
|ПОМЕСТИТЬ Сотрудники
|ИЗ
|	РегистрСведений.РаботникиОрганизаций.СрезПоследних(&Месяц, ) КАК
РаботникиОрганизацийСрезПоследних
|ГДЕ
```

```
|	РаботникиОрганизацийСрезПоследних.ГрафикРаботы.ВидГрафика =
ЗНАЧЕНИЕ(Перечисление.ВидыРабочихГрафиков.Сменный)
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ГрафикиРаботыПоВидамВремени.ГрафикРаботы,
|	СУММА(ГрафикиРаботыПоВидамВремени.ОсновноеЗначение) КАК ОсновноеЗначение
|ПОМЕСТИТЬ Графики
|ИЗ
|	РегистрСведений.ГрафикиРаботыПоВидамВремени КАК ГрафикиРаботыПоВидамВремени
|ГДЕ
|	ГрафикиРаботыПоВидамВремени.Месяц = &Месяц
|	И ГрафикиРаботыПоВидамВремени.План = ИСТИНА
|	И ГрафикиРаботыПоВидамВремени.ВидУчетаВремени =
ЗНАЧЕНИЕ(Перечисление.ВидыУчетаВремени.ПоНочнымЧасам)
|
|СГРУППИРОВАТЬ ПО
|	ГрафикиРаботыПоВидамВремени.ГрафикРаботы
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ГрафикиРаботыПоВидамВремени.ГрафикРаботы,
|	СУММА(ГрафикиРаботыПоВидамВремени.ОсновноеЗначение) КАК ОсновноеЗначение
|ПОМЕСТИТЬ НормаЧасов
|ИЗ
|	РегистрСведений.ГрафикиРаботыПоВидамВремени КАК ГрафикиРаботыПоВидамВремени
|ГДЕ
|	ГрафикиРаботыПоВидамВремени.Месяц = &Месяц
|	И ГрафикиРаботыПоВидамВремени.План = ИСТИНА
|	И ГрафикиРаботыПоВидамВремени.ВидУчетаВремени =
ЗНАЧЕНИЕ(Перечисление.ВидыУчетаВремени.ПоЧасам)
|
|СГРУППИРОВАТЬ ПО
|	ГрафикиРаботыПоВидамВремени.ГрафикРаботы
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	Сотрудники.Сотрудник,
|	СУММА(Графики.ОсновноеЗначение) КАК ОсновноеЗначение,
|	СУММА(НормаЧасов.ОсновноеЗначение) КАК Норма
|ПОМЕСТИТЬ СотрудникНочные
|ИЗ
|	Сотрудники КАК Сотрудники
|		ВНУТРЕННЕЕ СОЕДИНЕНИЕ Графики КАК Графики
|		ПО Сотрудники.ГрафикРаботы = Графики.ГрафикРаботы
|		ВНУТРЕННЕЕ СОЕДИНЕНИЕ НормаЧасов КАК НормаЧасов
|		ПО Сотрудники.ГрафикРаботы = НормаЧасов.ГрафикРаботы
|ГДЕ
|	Графики.ОсновноеЗначение <> 0
|
|СГРУППИРОВАТЬ ПО
|	Сотрудники.Сотрудник
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	РаботникиОрганизацийСрезПоследних.Сотрудник,
|	СУММА(ГрафикиРаботыПоВидамВремени.ОсновноеЗначение) КАК ПраздничныхЧасов
|ПОМЕСТИТЬ Праздничные
|ИЗ
```

```
	|	РегистрСведений.РегламентированныйПроизводственныйКалендарь КАК
РегламентированныйПроизводственныйКалендарь
	|		ВНУТРЕННЕЕ СОЕДИНЕНИЕ
РегистрСведений.ГрафикиРаботыПоВидамВремени КАК ГрафикиРаботыПоВидамВремени
	|			ВНУТРЕННЕЕ СОЕДИНЕНИЕ
РегистрСведений.РаботникиОрганизаций.СрезПоследних КАК РаботникиОрганизацийСрезПоследних
	|			ПО (РаботникиОрганизацийСрезПоследних.ГрафикРаботы =
ГрафикиРаботыПоВидамВремени.ГрафикРаботы)
	|		ПО РегламентированныйПроизводственныйКалендарь.ДатаКалендаря =
ГрафикиРаботыПоВидамВремени.Дата
	|ГДЕ
	|	ГрафикиРаботыПоВидамВремени.Месяц = &Месяц
	|	И ГрафикиРаботыПоВидамВремени.ВидУчетаВремени =
ЗНАЧЕНИЕ(Перечисление.ВидыУчетаВремени.ПоЧасам)
	|	И ГрафикиРаботыПоВидамВремени.План = ИСТИНА
	|	И РегламентированныйПроизводственныйКалендарь.ВидДня =
ЗНАЧЕНИЕ(Перечисление.ВидыДнейПроизводственногоКалендаря.Праздник)
	|	И ГрафикиРаботыПоВидамВремени.ОсновноеЗначение <> 0
	|
	|СГРУППИРОВАТЬ ПО
	|	ГрафикиРаботыПоВидамВремени.ГрафикРаботы,
	|	РегламентированныйПроизводственныйКалендарь.Пятидневка,
	|	РегламентированныйПроизводственныйКалендарь.ВидДня,
	|	РаботникиОрганизацийСрезПоследних.Сотрудник
	|;
	|
	|////////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|	НачальныеСведенияОСтажеРаботниковОрганизации.ФизЛицо КАК ФизЛицо,
	|	НачальныеСведенияОСтажеРаботниковОрганизации.ВидСтажа КАК ВидСтажа,
	|	НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа КАК ДатаСтажа,
	|	ВЫБОР
	|		КОГДА &Период <=
НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа
	|			ТОГДА ВЫРАЗИТЬ(0 КАК ЧИСЛО(5, 0))
	|		КОГДА ГОД(&Период) =
ГОД(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|			ТОГДА ВЫРАЗИТЬ(0 КАК ЧИСЛО(5, 0))
	|		КОГДА МЕСЯЦ(&Период) >
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|				ИЛИ МЕСЯЦ(&Период) =
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|					И ДЕНЬ(&Период) >=
ДЕНЬ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|			ТОГДА ГОД(&Период) -
ГОД(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|		ИНАЧЕ ГОД(&Период) -
ГОД(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа) - 1
	|	КОНЕЦ КАК ЛетСтажа,
	|	ВЫБОР
	|		КОГДА &Период <=
НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа
	|			ТОГДА ВЫРАЗИТЬ(0 КАК ЧИСЛО(5, 0))
	|		КОГДА ГОД(&Период) =
ГОД(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|				И МЕСЯЦ(&Период) =
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|			ТОГДА ВЫРАЗИТЬ(0 КАК ЧИСЛО(5, 0))
	|		КОГДА МЕСЯЦ(&Период) =
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|				И ДЕНЬ(&Период) >=
ДЕНЬ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|			ТОГДА ВЫРАЗИТЬ(0 КАК ЧИСЛО(5, 0))
```

```
	|			КОГДА МЕСЯЦ(&Период) =
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|				ТОГДА ВЫРАЗИТЬ(11 КАК ЧИСЛО(5, 0))
	|			КОГДА МЕСЯЦ(&Период) >
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|					И ДЕНЬ(&Период) >=
ДЕНЬ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|				ТОГДА МЕСЯЦ(&Период) -
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|			КОГДА МЕСЯЦ(&Период) >
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|				ТОГДА МЕСЯЦ(&Период) -
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа) - 1
	|			КОГДА ДЕНЬ(&Период) >=
ДЕНЬ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|				ТОГДА МЕСЯЦ(&Период) -
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа) + 12
	|			ИНАЧЕ МЕСЯЦ(&Период) -
МЕСЯЦ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа) + 11
	|		КОНЕЦ КАК МесяцевСтажа,
	|		ВЫБОР
	|			КОГДА &Период <=
НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа
	|				ТОГДА ВЫРАЗИТЬ(0 КАК ЧИСЛО(5, 0))
	|			КОГДА ДЕНЬ(&Период) =
ДЕНЬ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|				ТОГДА ВЫРАЗИТЬ(0 КАК ЧИСЛО(5, 0))
	|			КОГДА ДЕНЬ(&Период) >
ДЕНЬ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|				ТОГДА ДЕНЬ(&Период) -
ДЕНЬ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|			ИНАЧЕ ДЕНЬ(КОНЕЦПЕРИОДА(ДОБАВИТЬКДАТЕ(&Период, МЕСЯЦ, -1),
МЕСЯЦ)) + ДЕНЬ(&Период) - ДЕНЬ(НачальныеСведенияОСтажеРаботниковОрганизации.ДатаСтажа)
	|		КОНЕЦ КАК ДнейСтажа
	|ПОМЕСТИТЬ Стаж
	|ИЗ
	|
	РегистрСведений.НачальныеСведенияОСтажеРаботниковОрганизации.СрезПоследних(&Период, )
КАК НачальныеСведенияОСтажеРаботниковОрганизации
	|ГДЕ
	|		НачальныеСведенияОСтажеРаботниковОрганизации.ВидСтажа В(&ВидСтажа)
	|;
	|
	|////////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|		Стаж.ФизЛицо,
	|		Стаж.ЛетСтажа,
	|		Шкала.Начало,
	|		Шкала.Конец,
	|		Шкала.СтавкаПром,
	|		Шкала.СтавкаНеПром
	|ПОМЕСТИТЬ СтажПроценты
	|ИЗ
	|		Стаж КАК Стаж,
	|		Шкала КАК Шкала
	|ГДЕ
	|		Стаж.ЛетСтажа >= Шкала.Начало
	|		И Стаж.ЛетСтажа < Шкала.Конец
	|;
	|
	|////////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|		Работники.Сотрудник,
```

```
|	Работники.ТабельныйНомер КАК ТабельныйНомер,
|	Работники.Подразделение,
|	Работники.Должность,
|	Работники.ШПЗ,
|	Работники.РасчетныйКоэфициентМин,
|	Работники.РасчетныйКоэфициентМакс,
|	Оклад.Показатель1 / &Коэфициент КАК ПримененыйРасчетныйКоэфицент,
|	Работники.ВилкаМин,
|	Работники.ВилкаМакс,
|	Работники.ВилкаСредняя,
|	ЕСТЬNULL(Оклад.Показатель1, 0) КАК Оклад,
|	ВЫБОР
|		КОГДА Премия.Показатель1 ЕСТЬ NULL
|			ТОГДА ЕСТЬNULL(КвартальнаяПремия.Показатель, 0)
|		ИНАЧЕ ЕСТЬNULL(Премия.Показатель1, 0)
|	КОНЕЦ КАК ПремияПроцент,
|	ВЫБОР
|		КОГДА Премия.Показатель1 ЕСТЬ NULL
|			ТОГДА ЕСТЬNULL(Оклад.Показатель1 *
(ЕСТЬNULL(КвартальнаяПремия.Показатель, 0) + ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) +
ЕСТЬNULL(Ненормированый.Показатель1, 0)) / 100, 0) + ЕСТЬNULL(Совмещение.Показатель1, 0) +
(ЕСТЬNULL(Оклад.Показатель1 * ЕСТЬNULL(Вредность.Показатель1,
Вредность2.ПроцентОплатыЗаРаботуСВреднымиУсловиямиТруда) / 100, 0) +
ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) + ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 /
СотрудникНочные.Норма, 0)) * КвартальнаяПремия.Показатель / 100
|		ИНАЧЕ ЕСТЬNULL(Оклад.Показатель1 * (ЕСТЬNULL(Премия.Показатель1, 0) +
ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) + ЕСТЬNULL(Ненормированый.Показатель1, 0)) / 100, 0) +
ЕСТЬNULL(Совмещение.Показатель1, 0) + (ЕСТЬNULL(Оклад.Показатель1 *
ЕСТЬNULL(Вредность.Показатель1, Вредность2.ПроцентОплатыЗаРаботуСВреднымиУсловиямиТруда) /
100, 0) + ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) + ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 /
СотрудникНочные.Норма, 0)) * Премия.Показатель1 / 100
|	КОНЕЦ КАК Премия,
|	ЕСТЬNULL(Класность.Показатель1, 0) КАК КлассностьПроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * Класность.Показатель1 / 100, 0) КАК Класность,
|	ЕСТЬNULL(Вредность.Показатель1,
Вредность2.ПроцентОплатыЗаРаботуСВреднымиУсловиямиТруда) КАК ВредныеУсловияТрудаПроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * ЕСТЬNULL(Вредность.Показатель1,
Вредность2.ПроцентОплатыЗаРаботуСВреднымиУсловиямиТруда) / 100, 0) КАК ВредныеУсловияТруда,
|	ЕСТЬNULL(Ненормированый.Показатель1, 0) КАК НенормированыйПроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * Ненормированый.Показатель1 / 100, 0) КАК
Ненормированый,
|	ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) КАК РЗО_ЗОРПроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * РЗО_ЗОР.Показатель1 / 100, 0) КАК РЗО_ЗОР,
|	ЕСТЬNULL(ВО.Показатель1, 0) КАК ВОПроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * ВО.Показатель1 / 100, 0) КАК ВО,
|	ЕСТЬNULL(КБ.Показатель1, 0) КАК КБПроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * КБ.Показатель1 / 100, 0) КАК КБ,
|	ЕСТЬNULL(КТ.Показатель1, 0) КАК КТПроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * КТ.Показатель1 / 100, 0) КАК КТ,
|	ЕСТЬNULL(Секретность.Показатель1, 0) КАК СекретностьПроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * Секретность.Показатель1 / 100, 0) КАК Секретность,
|	ЕСТЬNULL(ВДП.Показатель1, 0) КАК ВДППроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * ВДП.Показатель1 / 100, 0) КАК ВДП,
|	ЕСТЬNULL(Персональная.Показатель1, 0) КАК ПерсональнаяПроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * Персональная.Показатель1 / 100, 0) КАК Персональная,
|	ЕСТЬNULL(Звание.Показатель1, 0) КАК ЗваниеПроцент,
|	ЕСТЬNULL(Оклад.Показатель1 * Звание.Показатель1 / 100, 0) КАК Звание,
|	ЕСТЬNULL(Совмещение.Показатель1 / Оклад.Показатель1 * 100, 0) КАК
СовмещениеПроцент,
|	ЕСТЬNULL(Совмещение.Показатель1, 0) КАК Совмещение,
|	ЕСТЬNULL(НаучнаяСтепень.Показатель1, 0) КАК НаучнаяСтепеньПроцент,
```

```
	|	ЕСТЬNULL(Оклад.Показатель1 * НаучнаяСтепень.Показатель1 / 100, 0) КАК
НаучнаяСтепень,
	|	СтажПроценты.ЛетСтажа,
	|	ВЫБОР
	|		КОГДА
Работники.Сотрудник.ТекущееПодразделениеОрганизации.ВидыПодразделенийОрганизации =
ЗНАЧЕНИЕ(справочник.ВидыПодразделенийОрганизации.Непроизводственые)
	|			ТОГДА СтажПроценты.СтавкаНеПром * 100
	|		ИНАЧЕ СтажПроценты.СтавкаПром * 100
	|	КОНЕЦ КАК СтажПроцент,
	|	ВЫБОР
	|		КОГДА
Работники.Сотрудник.ТекущееПодразделениеОрганизации.ВидыПодразделенийОрганизации =
ЗНАЧЕНИЕ(справочник.ВидыПодразделенийОрганизации.Непроизводственые)
	|			ТОГДА Оклад.Показатель1 * СтажПроценты.СтавкаНеПром
	|		ИНАЧЕ Оклад.Показатель1 * СтажПроценты.СтавкаПром
	|	КОНЕЦ КАК Стаж,
	|	ВЫБОР
	|		КОГДА
Работники.Сотрудник.ТекущееПодразделениеОрганизации.ВидыПодразделенийОрганизации =
ЗНАЧЕНИЕ(справочник.ВидыПодразделенийОрганизации.Непроизводственые)
	|			ТОГДА ВЫБОР
	|					КОГДА Премия.Показатель1 ЕСТЬ NULL
	|						ТОГДА ЕСТЬNULL(Оклад.Показатель1, 0) +
(ЕСТЬNULL(Оклад.Показатель1 * (ЕСТЬNULL(КвартальнаяПремия.Показатель, 0) +
ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) + ЕСТЬNULL(Ненормированый.Показатель1, 0)) / 100, 0) +
ЕСТЬNULL(КвартальнаяПремия.Показатель, 0) * ЕСТЬNULL(Совмещение.Показатель1, 0) / 100 +
(ЕСТЬNULL(Оклад.Показатель1 * ЕСТЬNULL(Вредность.Показатель1,
Вредность2.ПроцентОплатыЗаРаботуСВреднымиУсловиямиТруда) / 100, 0) +
ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) + ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 /
СотрудникНочные.Норма, 0)) * Премия.Показатель1 / 100) + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Класность.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Вредность.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Ненормированый.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(ВДП.Показатель1, 0) / 100 + ЕСТЬNULL(Совмещение.Показатель1, 0) +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Звание.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(НаучнаяСтепень.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Секретность.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Персональная.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(СтажПроценты.СтавкаНеПром, 0) +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(КБ.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(ВО.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(КТ.Показатель1, 0) / 100 + ЕСТЬNULL(ОВД.Показатель1,
0) + ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 / СотрудникНочные.Норма, 0) +
ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) + ЕСТЬNULL(Оклад.Показатель1 * ДоплатаЗаРоздРобот.Показатель1 / 100, 0)
	|					ИНАЧЕ ЕСТЬNULL(Оклад.Показатель1, 0) +
(ЕСТЬNULL(Оклад.Показатель1 * (ЕСТЬNULL(Премия.Показатель1, 0) +
ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) + ЕСТЬNULL(Ненормированый.Показатель1, 0)) / 100, 0) +
ЕСТЬNULL(Премия.Показатель1, 0) * ЕСТЬNULL(Совмещение.Показатель1, 0) / 100 +
(ЕСТЬNULL(Оклад.Показатель1 * ЕСТЬNULL(Вредность.Показатель1,
Вредность2.ПроцентОплатыЗаРаботуСВреднымиУсловиямиТруда) / 100, 0) +
ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) + ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 /
СотрудникНочные.Норма, 0)) * Премия.Показатель1 / 100) + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Класность.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Вредность.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Ненормированый.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(ВДП.Показатель1, 0) / 100 + ЕСТЬNULL(Совмещение.Показатель1, 0) +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Звание.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(НаучнаяСтепень.Показатель1, 0) / 100 +
```

```
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Секретность.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Персональная.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(СтажПроценты.СтавкаНеПром, 0) +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(КБ.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(ВО.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(КТ.Показатель1, 0) / 100 + ЕСТЬNULL(ОВД.Показатель1,
0) + ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 / СотрудникНочные.Норма, 0) +
ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) + ЕСТЬNULL(Оклад.Показатель1 * ДоплатаЗаРоздРобот.Показатель1 / 100, 0)
        |                                   КОНЕЦ
        |                ИНАЧЕ ВЫБОР
        |                                   КОГДА Премия.Показатель1 ЕСТЬ NULL
        |                                       ТОГДА ЕСТЬNULL(Оклад.Показатель1, 0) +
(ЕСТЬNULL(Оклад.Показатель1 * (ЕСТЬNULL(КвартальнаяПремия.Показатель, 0) +
ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) + ЕСТЬNULL(Ненормированый.Показатель1, 0)) / 100, 0) +
ЕСТЬNULL(КвартальнаяПремия.Показатель, 0) * ЕСТЬNULL(Совмещение.Показатель1, 0) / 100 +
(ЕСТЬNULL(Оклад.Показатель1 * ЕСТЬNULL(Вредность.Показатель1,
Вредность2.ПроцентОплатыЗаРаботуСВреднымиУсловиямиТруда) / 100, 0) +
ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) + ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 /
СотрудникНочные.Норма, 0)) * Премия.Показатель1 / 100) + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Класность.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Вредность.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Ненормированый.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(ВДП.Показатель1, 0) / 100 + ЕСТЬNULL(Совмещение.Показатель1, 0) +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Звание.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(НаучнаяСтепень.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Секретность.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Персональная.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(СтажПроценты.СтавкаПром, 0) +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(КБ.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(ВО.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(КТ.Показатель1, 0) / 100 + ЕСТЬNULL(ОВД.Показатель1,
0) + ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 / СотрудникНочные.Норма, 0) +
ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) + ЕСТЬNULL(Оклад.Показатель1 * ДоплатаЗаРоздРобот.Показатель1 / 100, 0)
        |                                   ИНАЧЕ ЕСТЬNULL(Оклад.Показатель1, 0) +
(ЕСТЬNULL(Оклад.Показатель1 * (ЕСТЬNULL(Премия.Показатель1, 0) +
ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) + ЕСТЬNULL(Ненормированый.Показатель1, 0)) / 100, 0) +
ЕСТЬNULL(Премия.Показатель1, 0) * ЕСТЬNULL(Совмещение.Показатель1, 0) / 100 +
(ЕСТЬNULL(Оклад.Показатель1 * ЕСТЬNULL(Вредность.Показатель1,
Вредность2.ПроцентОплатыЗаРаботуСВреднымиУсловиямиТруда) / 100, 0) +
ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) + ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 /
СотрудникНочные.Норма, 0)) * Премия.Показатель1 / 100) + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Класность.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Вредность.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(Ненормированый.Показатель1, 0) / 100 + ЕСТЬNULL(Оклад.Показатель1, 0) *
ЕСТЬNULL(ВДП.Показатель1, 0) / 100 + ЕСТЬNULL(Совмещение.Показатель1, 0) +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Звание.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(НаучнаяСтепень.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Секретность.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(РЗО_ЗОР.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(Персональная.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(СтажПроценты.СтавкаПром, 0) +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(КБ.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(ВО.Показатель1, 0) / 100 +
ЕСТЬNULL(Оклад.Показатель1, 0) * ЕСТЬNULL(КТ.Показатель1, 0) / 100 + ЕСТЬNULL(ОВД.Показатель1,
0) + ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 / СотрудникНочные.Норма, 0) +
ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) + ЕСТЬNULL(Оклад.Показатель1 * ДоплатаЗаРоздРобот.Показатель1 / 100, 0)
```

```
|                                КОНЕЦ
|            КОНЕЦ КАК Итого,
|
|        Работники.Сотрудник.ТекущееПодразделениеОрганизации.ВидыПодразделенийОрганизации,
|            ЕСТЬNULL(ОВД.Показатель1, 0) КАК ОВД,
|            ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение, 0) КАК НочныхЧасов,
|            ЕСТЬNULL(СотрудникНочные.ОсновноеЗначение * 40 / 100 * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) КАК Ночных,
|            Работники.Должность.КодКП КАК КодДолжности,
|            ЕСТЬNULL(Праздничные.ПраздничныхЧасов, 0) КАК ПраздничныхЧасов,
|            ЕСТЬNULL(Праздничные.ПраздничныхЧасов * Оклад.Показатель1 /
СотрудникНочные.Норма, 0) КАК Праздничных,
|            ЕСТЬNULL(ДоплатаЗаРоздРобот.Показатель1, 0) КАК ЗаРоздРоботПроцент,
|            ЕСТЬNULL(Оклад.Показатель1 * ДоплатаЗаРоздРобот.Показатель1 / 100, 0) КАК
ЗаРоздРобот
|        ИЗ
|            Работники КАК Работники
|                ЛЕВОЕ СОЕДИНЕНИЕ Класность КАК Класность
|                ПО Работники.Сотрудник = Класность.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ ОВД КАК ОВД
|                ПО Работники.Сотрудник = ОВД.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ Премия КАК Премия
|                ПО Работники.Сотрудник = Премия.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ Оклад КАК Оклад
|                ПО Работники.Сотрудник = Оклад.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ Вредность КАК Вредность
|                ПО Работники.Сотрудник = Вредность.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ РЗО_ЗОР КАК РЗО_ЗОР
|                ПО Работники.Сотрудник = РЗО_ЗОР.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ ВО КАК ВО
|                ПО Работники.Сотрудник = ВО.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ Секретность КАК Секретность
|                ПО Работники.Сотрудник = Секретность.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ Персональная КАК Персональная
|                ПО Работники.Сотрудник = Персональная.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ ВДП КАК ВДП
|                ПО Работники.Сотрудник = ВДП.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ Совмещение КАК Совмещение
|                ПО Работники.Сотрудник = Совмещение.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ Звание КАК Звание
|                ПО Работники.Сотрудник = Звание.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ Ненормированый КАК Ненормированый
|                ПО Работники.Сотрудник = Ненормированый.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ НаучнаяСтепень КАК НаучнаяСтепень
|                ПО Работники.Сотрудник = НаучнаяСтепень.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ СтажПроценты КАК СтажПроценты
|                ПО Работники.Сотрудник.Физлицо = СтажПроценты.ФизЛицо
|                ЛЕВОЕ СОЕДИНЕНИЕ КБ КАК КБ
|                ПО Работники.Сотрудник = КБ.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ КТ КАК КТ
|                ПО Работники.Сотрудник = КТ.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ Вредность2 КАК Вредность2
|                ПО Работники.Сотрудник = Вредность2.СотрудникиОрганизаций
|                ЛЕВОЕ СОЕДИНЕНИЕ СотрудникНочные КАК СотрудникНочные
|                ПО Работники.Сотрудник = СотрудникНочные.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ Праздничные КАК Праздничные
|                ПО Работники.Сотрудник = Праздничные.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ ДоплатаЗаРоздРобот КАК ДоплатаЗаРоздРобот
|                ПО Работники.Сотрудник = ДоплатаЗаРоздРобот.Сотрудник
|                ЛЕВОЕ СОЕДИНЕНИЕ КвартальнаяПремия КАК КвартальнаяПремия
|                ПО Работники.Сотрудник = КвартальнаяПремия.Сотрудник
|
|        УПОРЯДОЧИТЬ ПО
```

```
|           ТабельныйНомер";

запрос.УстановитьПараметр("Категория","категория");
запрос.УстановитьПараметр("Период",Объект.Дата);
запрос.УстановитьПараметр("Коэфициент",Объект.Коэфициент);
запрос.УстановитьПараметр("ВидРасчетаОВД",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.Най
тиПоНаименованию("Надбавка за ОВД"));
запрос.УстановитьПараметр("ВидРасчетаВДП",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.Най
тиПоНаименованию("Надбавка за  ВДП та проф. майстерність"));
запрос.УстановитьПараметр("ВидРасчетаХарактер",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций
.НайтиПоНаименованию("Допл. за раз. короткосрочный характер работ"));
запрос.УстановитьПараметр("ВидРасчетаВО",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.Найт
иПоНаименованию("Доплата за ведення військового обліку"));
запрос.УстановитьПараметр("ВидРасчетаЗОР",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.Най
тиПоНаименованию("Доплата за збільшений обсяг робіт"));
запрос.УстановитьПараметр("ВидРасчетаКБ",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.Найт
иПоНаименованию("Доплата за керивництво бригадою"));
запрос.УстановитьПараметр("ВидРасчетаРЗО",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.Най
тиПоНаименованию("Доплата за розш.зони обслугов."));
запрос.УстановитьПараметр("ВидРасчетаСовмещение",ПланыВидовРасчета.ОсновныеНачисленияОрганиза
ций.НайтиПоНаименованию("Доплата за суміщ. професій"));
запрос.УстановитьПараметр("ВидРасчетаВредность",ПланыВидовРасчета.ОсновныеНачисленияОрганизаци
й.НайтиПоНаименованию("Доплата за шкідливість"));
запрос.УстановитьПараметр("ВидРасчетаПерсональная",ПланыВидовРасчета.ОсновныеНачисленияОрганиз
аций.НайтиПоНаименованию("Доплата персональна"));
запрос.УстановитьПараметр("ВидРасчетаЗвание",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.Н
айтиПоНаименованию("Надбавка за звання"));
запрос.УстановитьПараметр("ВидРасчетаКласность",ПланыВидовРасчета.ОсновныеНачисленияОрганизаци
й.НайтиПоНаименованию("Надбавка за класність"));
запрос.УстановитьПараметр("ВидРасчетаНаучнаяДеятельность",ПланыВидовРасчета.ОсновныеНачисления
Организаций.НайтиПоНаименованию("Надбавка за наукову діяльність"));
запрос.УстановитьПараметр("ВидРасчетаНаучнаяСтепень",ПланыВидовРасчета.ОсновныеНачисленияОрга
низаций.НайтиПоНаименованию("Надбавка за наукову ступінь"));
запрос.УстановитьПараметр("ВидРасчетаСекретность",ПланыВидовРасчета.ОсновныеНачисленияОрганиза
ций.НайтиПоНаименованию("Надбавка за секретність"));
запрос.УстановитьПараметр("ВидРасчетаНенормированый",ПланыВидовРасчета.ОсновныеНачисленияОрга
низаций.НайтиПоНаименованию("Доплата за ненормований робочий день"));
запрос.УстановитьПараметр("ВидРасчетаКТ",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.Найт
иПоНаименованию("Доплата з конкурсних торгів"));
запрос.УстановитьПараметр("ВидРасчетаДоплатаЗаРоздРобот",ПланыВидовРасчета.ОсновныеНачисленияО
рганизаций.ДоплатаЗаРоздРобот);//
запрос.УстановитьПараметр("ВидСтажаПром",Справочники.ВидыСтажа.НайтиПоНаименованию("для
выслуги лет (26)"));
запрос.УстановитьПараметр("ВидСтажаНеПром",Справочники.ВидыСтажа.НайтиПоНаименованию("Стаж
для непром"));
запрос.УстановитьПараметр("Месяц",НачалоМесяца(Объект.Дата));
запрос.УстановитьПараметр("Организация",Объект.Организация);
//запрос.УстановитьПараметр("ВидРасчетаВДП",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.На
йтиПоНаименованию(""));
//запрос.УстановитьПараметр("ВидРасчетаВДП",ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.На
йтиПоНаименованию(""));

ВидСтажа=Новый Массив;
ВидСтажа.Добавить(Справочники.ВидыСтажа.НайтиПоНаименованию("для выслуги лет (26)"));
ВидСтажа.Добавить(Справочники.ВидыСтажа.НайтиПоНаименованию("Стаж для непром"));
Запрос.УстановитьПараметр("ВидСтажа", ВидСтажа);

ВидРасчетаРЗО_ЗОР=Новый Массив;
ВидРасчетаРЗО_ЗОР.Добавить(ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.НайтиПоНаименова
нию("Доплата за збільшений обсяг робіт"));
ВидРасчетаРЗО_ЗОР.Добавить(ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.НайтиПоНаименова
нию("Доплата за розш.зони обслугов."));
```

```
//ВидРасчетаРЗО_ЗОР_ВО.Добавить(ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.НайтиПоНаименованию("Доплата за ведення військового обліку"));
Запрос.УстановитьПараметр("ВидРасчетаРЗО_ЗОР", ВидРасчетаРЗО_ЗОР);

//выборка = запрос.Выполнить().Выбрать();
Объект.ТабличнаяЧасть1.Загрузить(Запрос.Выполнить().Выгрузить());

КонецПроцедуры
```

ДОДАТОК Е

Код модулю звіту для демонстрації витиснення видів розрахунку

```
&НаКлиенте
Процедура ЗаполнитьДанными(Команда)
	ПолучитьДанныеФПД(ДиаграммаГанта, Сотрудник);
КонецПроцедуры

&НаСервереБезКонтекста
Процедура ПолучитьДанныеФПД(Диаграмма, Сотрудник)
   Запрос = Новый Запрос;

	Запрос.Текст = "ВЫБРАТЬ
		|	НачисленияФактическийПериодДействия.Сотрудник,
		|	НачисленияФактическийПериодДействия.ВидРасчета,
		|	НачисленияФактическийПериодДействия.ПериодДействияНачало,
		|	НачисленияФактическийПериодДействия.ПериодДействияКонец,
		|	НачисленияФактическийПериодДействия.Регистратор
		|ИЗ
		|
	РегистрРасчета.ОсновныеНачисленияРаботниковОрганизаций.ФактическийПериодДействия(Сотру
дник = &Сотрудник) КАК НачисленияФактическийПериодДействия";
	Запрос.УстановитьПараметр("Сотрудник", Сотрудник);
	РезультатЗапроса = Запрос.Выполнить();

	Диаграмма.Обновление = Ложь;
	Диаграмма.Очистить();

	Выборка = РезультатЗапроса.Выбрать();
	Пока Выборка.Следующий() Цикл
		Точка = Диаграмма.УстановитьТочку(Выборка.Сотрудник);
		Серия = Диаграмма.УстановитьСерию(Выборка.ВидРасчета);

		Значение = Диаграмма.ПолучитьЗначение(Точка, Серия);

		Значение.расшифровка =Выборка.регистратор;

		Интервал = Значение.Добавить();
		Интервал.Начало = Выборка.ПериодДействияНачало;
		Интервал.Конец = Выборка.ПериодДействияКонец;
	КонецЦикла;

	Диаграмма.Обновление = Истина;
	Диаграмма.ОтображатьПустыеЗначения=Ложь;
	Диаграмма.ПоддержкаМасштаба = ПоддержкаМасштабаДиаграммыГанта.Фиксированная;

КонецПроцедуры

&НаКлиенте
Процедура СотрудникПриИзменении(Элемент)
ПолучитьДанныеФПД(ДиаграммаГанта, Сотрудник);
КонецПроцедуры
```

ДОДАТОК Ж

Код модулю документу «оплата святкових і вихідних днів» процедура «обробка заповнення»

```
Процедура ОбработкаЗаполнения(ДанныеЗаполнения, СтандартнаяОбработка)

	Если ТипЗнч(ДанныеЗаполнения) = Тип("ДокументСсылка.ОтсутствиеНаРаботеОрганизаций") 
Тогда
		// Заповнення шапки
		Комментарий = ДанныеЗаполнения.Комментарий + " " + ДанныеЗаполнения.Номер;
		КраткийСоставДокумента = ДанныеЗаполнения.КраткийСоставДокумента;
		Организация = ДанныеЗаполнения.Организация;
		Ответственный = ДанныеЗаполнения.Ответственный;
		ПериодРегистрации = ДанныеЗаполнения.ПериодРегистрации;
		ТерриториальноеУправление = ДанныеЗаполнения.ТерриториальноеУправление;

		Для Каждого ТекСтрокаРаботникиОрганизации Из 
ДанныеЗаполнения.РаботникиОрганизации Цикл
			если ТекСтрокаРаботникиОрганизации.ПричинаОтсутствия = 
Перечисления.СостоянияРаботникаОрганизации.РоботаУВихіднийЗОдинарноюОплатою тогда
			НоваяСтрока = Начисления.Добавить();
			НоваяСтрока.ДатаВыхода = ТекСтрокаРаботникиОрганизации.ДатаНачала;
			НоваяСтрока.Сотрудник = ТекСтрокаРаботникиОрганизации.Сотрудник;
			НоваяСтрока.Физлицо = ТекСтрокаРаботникиОрганизации.ФизЛицо;
			НоваяСтрока.ПодразделениеОрганизации = 
ТекСтрокаРаботникиОрганизации.Сотрудник.ТекущееПодразделениеОрганизации;
			НоваяСтрока.ВидРасчета = 
ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.ОплатаВыходныхОдинарныйРВО;
			если ЗначениеЗаполнено(ТекСтрокаРаботникиОрганизации.Години) тогда
			НоваяСтрока.ОтработаноЧасов = ТекСтрокаРаботникиОрганизации.Години;
			иначе
			НоваяСтрока.ОтработаноЧасов = 8.25;
			КонецЕсли;
			НоваяСтрока.Размер = 
ПроцедурыУправленияПерсоналом.ЧасоваяТарифнаяСтавкаРаботникаОрг(НоваяСтрока.Сотрудник, 
НоваяСтрока.ДатаВыхода, КонецМесяца(ПериодРегистрации),мВалютаРегламентированногоУчета);
			НоваяСтрока.Результат = НоваяСтрока.Размер * НоваяСтрока.ОтработаноЧасов;
			КонецЕсли;
			если ТекСтрокаРаботникиОрганизации.ПричинаОтсутствия = 
Перечисления.СостоянияРаботникаОрганизации.РоботаУВихіднийЗПодвійноюОплатою тогда
			НоваяСтрока = Начисления.Добавить();
			НоваяСтрока.ДатаВыхода = ТекСтрокаРаботникиОрганизации.ДатаНачала;
			НоваяСтрока.Сотрудник = ТекСтрокаРаботникиОрганизации.Сотрудник;
			НоваяСтрока.Физлицо = ТекСтрокаРаботникиОрганизации.ФизЛицо;
			НоваяСтрока.ПодразделениеОрганизации = 
ТекСтрокаРаботникиОрганизации.Сотрудник.ТекущееПодразделениеОрганизации;
			НоваяСтрока.ВидРасчета = 
ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.ОплатаВыходныхДвойнойРВП;
			если ЗначениеЗаполнено(ТекСтрокаРаботникиОрганизации.Години) тогда
			НоваяСтрока.ОтработаноЧасов = ТекСтрокаРаботникиОрганизации.Години;
			иначе
			НоваяСтрока.ОтработаноЧасов = 8.25;
			КонецЕсли;
			НоваяСтрока.Размер = 
ПроцедурыУправленияПерсоналом.ЧасоваяТарифнаяСтавкаРаботникаОрг(НоваяСтрока.Сотрудник, 
НоваяСтрока.ДатаВыхода, КонецМесяца(ПериодРегистрации),мВалютаРегламентированногоУчета);
```

```
                        НоваяСтрока.Результат = НоваяСтрока.Размер * НоваяСтрока.ОтработаноЧасов;
                        //-----------------------------------------------
                        НоваяСтрока = Начисления.Добавить();
                        НоваяСтрока.ДатаВыхода = ТекСтрокаРаботникиОрганизации.ДатаНачала;
                        НоваяСтрока.Сотрудник = ТекСтрокаРаботникиОрганизации.Сотрудник;
                        НоваяСтрока.Физлицо = ТекСтрокаРаботникиОрганизации.ФизЛицо;
                        НоваяСтрока.ПодразделениеОрганизации =
ТекСтрокаРаботникиОрганизации.Сотрудник.ТекущееПодразделениеОрганизации;
                        НоваяСтрока.ВидРасчета =
ПланыВидовРасчета.ОсновныеНачисленияОрганизаций.ДоплатаВыходныхДвойнойРВП;
                        если ЗначениеЗаполнено(ТекСтрокаРаботникиОрганизации.Години) тогда
                        НоваяСтрока.ОтработаноЧасов = ТекСтрокаРаботникиОрганизации.Години;
                        иначе
                        НоваяСтрока.ОтработаноЧасов = 8.25;
                        КонецЕсли;
                        НоваяСтрока.Размер =
ПроцедурыУправленияПерсоналом.ЧасоваяТарифнаяСтавкаРаботникаОрг(НоваяСтрока.Сотрудник,
НоваяСтрока.ДатаВыхода, КонецМесяца(ПериодРегистрации),мВалютаРегламентированногоУчета);
                        НоваяСтрока.Результат = НоваяСтрока.Размер * НоваяСтрока.ОтработаноЧасов;
                        КонецЕсли;

                КонецЦикла;
        КонецЕсли;

КонецПроцедуры
```