



КЕРІВНИЦТВО ПРОГРАМІСТА

Версія 7.11.027

Сторінок 61

2018

ЗМІСТ

Інструментарій програміста в системі	3
1 Вбудована алгоритмічна мова	3
1.1 Вступ.....	3
1.2 Редактор програм	3
1.3 Елементи програми	5
1.3.1 Поля.....	7
1.3.2 Функції.....	8
1.3.3 Бібліотека	9
1.3.4 Довідники	10
1.3.5 Реєстри користувача.....	11
1.3.6 Вставки	12
2 Засоби створення вихідних форм	12
2.1 Загальні відомості для створення і коригування, перевірки звітів.....	12
2.1.1 Виклик меню звітів	13
2.1.2 Імпорт-експорт шаблонів	15
2.1.3 Генерація звітів	15
2.1.4 Перегляд і друк звітів.	16
2.1.5 Експорт в звітах.....	17
2.1.6 Архів звітів.....	17
2.1.7 Шаблони	18
2.2 Дизайнер звітів FastReport	22
2.2.1 Етапи побудови звіту.....	22
2.3 Експорт зовнішньої звітності.	28
2.3.1 Загальні відомості.....	28
2.3.2 Перелік доступних констант, функцій і процедур.....	29
2.3.3 Приклад використання функцій експорту зовнішньої звітності в звіті Fast Report	32
2.4 Імпорт файлів MS Office Word (*.doc) в FastReport	35
2.5 Розробка звітів MS Office (Excel, Word)	36
2.5.1 Приклад програми Excel звіту	38
2.5.2 Приклад програми Word звіту	40
3 Робота з COM об'єктами системи IS-pro	46
3.1 COM об'єкт для виконання SQL запитів.....	47
3.2 Запуск ресурсних модулів з програми BASIC.....	49
3.3 Проблеми при розробці звітів	50
4. Використання в програмі користувача розрахункових функцій головної книги	52
4.1 Програма отримання оборотів і сальдо по рахунку в програмах користувача	52
4.2 Приклади використання розрахункових функцій в програмах користувача.....	53
4.3 Опис функцій головної книги.....	57

Інструментарій програміста в системі

В системі для створення нестандартних проводок (проводок, які формуються за переліком умов), автоматичне формування текстових документів або коментарів (тексти наказів, договорів), розрахунку законодавчо невизначених процесів (наприклад, обробка банківських документів з програми клієнт–банк) та інших подій використовується програми користувача. Для створення програм використовується вбудована алгоритмічна мова. Для формування звітів необхідних для роботи користувача, крім представлених в системі, існують засоби створення і формування вихідних форм і засоби доступу до об'єктів даних системи.

1 Вбудована алгоритмічна мова

1.1 Вступ

Одна з найбільш важливих функцій системного комплексу **IS-pro** – можливість завдання алгоритмічних процедур. За допомогою вбудованих або програмованих користувачем процедур, що застосовуються в усіх системах комплексу, відбудеться налаштування проводок по рахунках бухгалтерського обліку, алгоритмів розрахунку заробітної плати, податків, амортизації основних засобів, програмування і обчислення параметрів в звітах, настройка користувацького фільтра, а також вирішуються інші завдання. Для написання процедур в комплекс вбудований **Редактор програм**, забезпечений довідником змінних, функцій і конструкцій мови (**Елементи програми**), а також функцією синтаксичного перевірки написаної процедури.

1.2 Редактор програм

Редактор програм викликається в тих місцях підсистем **IS-pro**, де потрібно налаштування алгоритму програмним способом. Наприклад, в підсистемі **Головна книга** редактор програм викликається з розділу **Проведення** модуля **Налаштування** для додавання або зміни алгоритму формування проводки.

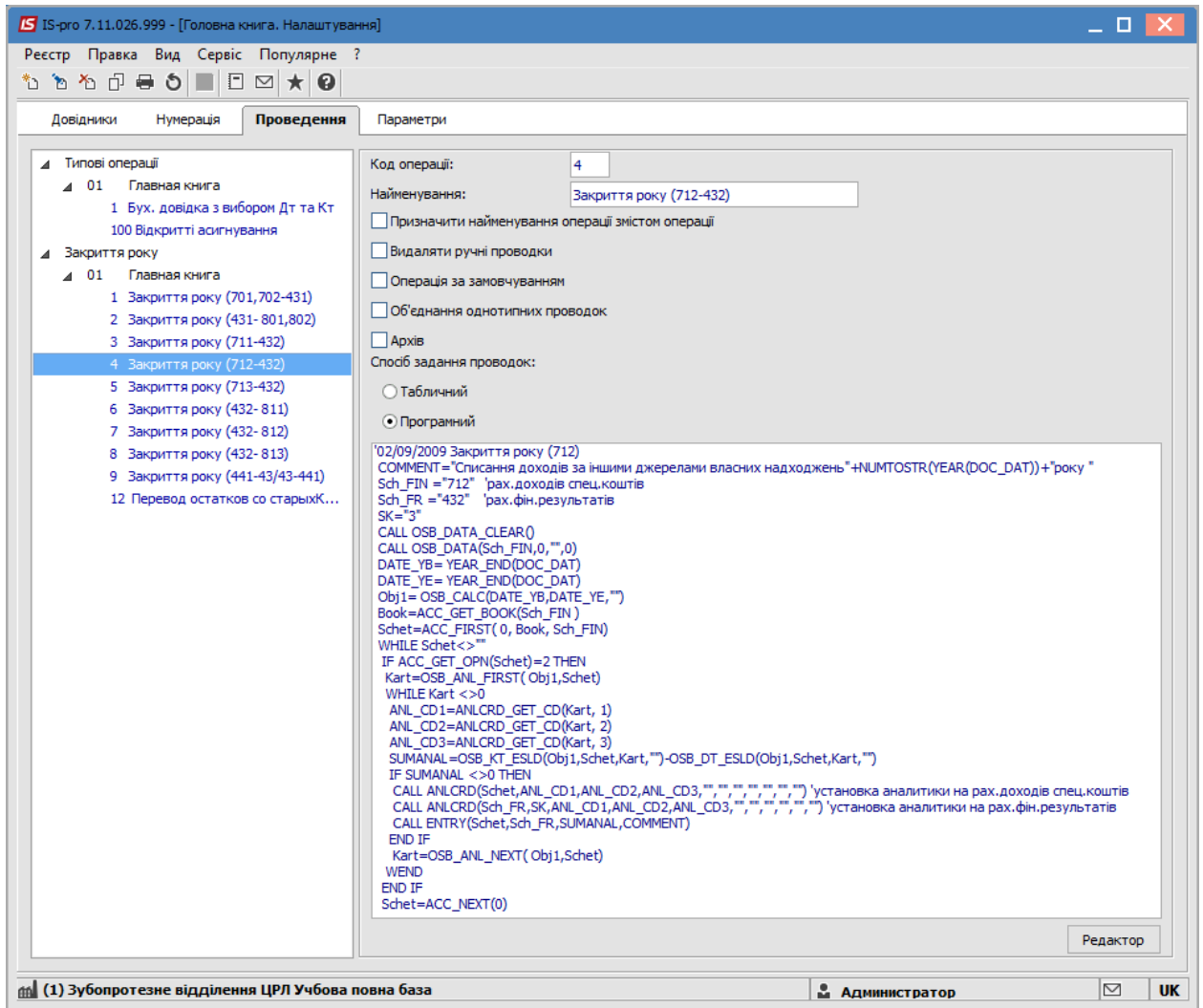


Рис. 1 – Модуль Налаштування підсистеми Головна книга

Щоб змінити задану проводку, курсор встановлюється на потрібний пункт лівій панелі реєстру і потім по клавіші **Tab** курсор переводиться на праву панель. Далі вибирається спосіб завдання проводки **Програмний** і по клавіші **F4** викликається **Редактор програм**.

Редактор програм це текстовий редактор, що підтримує стандартні функції редагування тексту – виділення фрагмента, видалення фрагмента, копіювання в буфер обміну, вставка з буфера. Виділення фрагмента тексту виконується по комбінації клавіш **Shift** і управління курсором. Знімається виділення натисканням будь-якої клавіші. Команди вставити, видалити, вирізати і копіювати виконуються через пункт меню **Правка**. Вставка проводиться з буфера обміну або зі спеціального довідника **Елементи програми**. Пункт меню **Правка / Знайти** дозволяє знайти в тексті програми заданий рядок. Пункт меню **Правка / Знайти далі** обробить решту тексту і знайде, де дана рядок зустрічається в наступний раз. Пункт меню **Правка / Зберегти блок в файл** збереже виділений фрагмент програми в існуючий текстовий файл, перезаписавши його вміст. В текст програми вставляється вміст текстового файлу, вставка виконується через пункт меню **Правка / Вставити блок з файлу**.

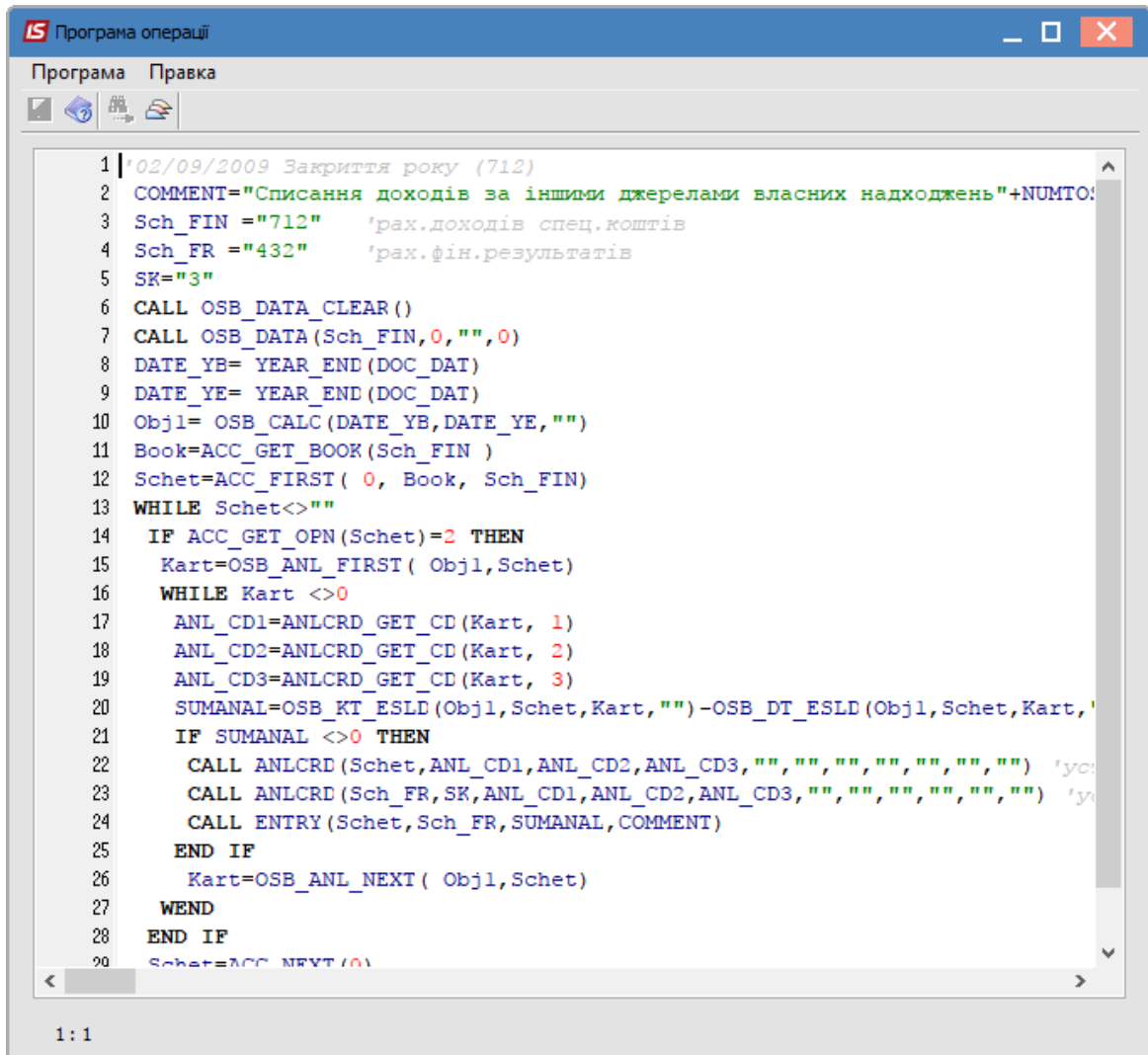
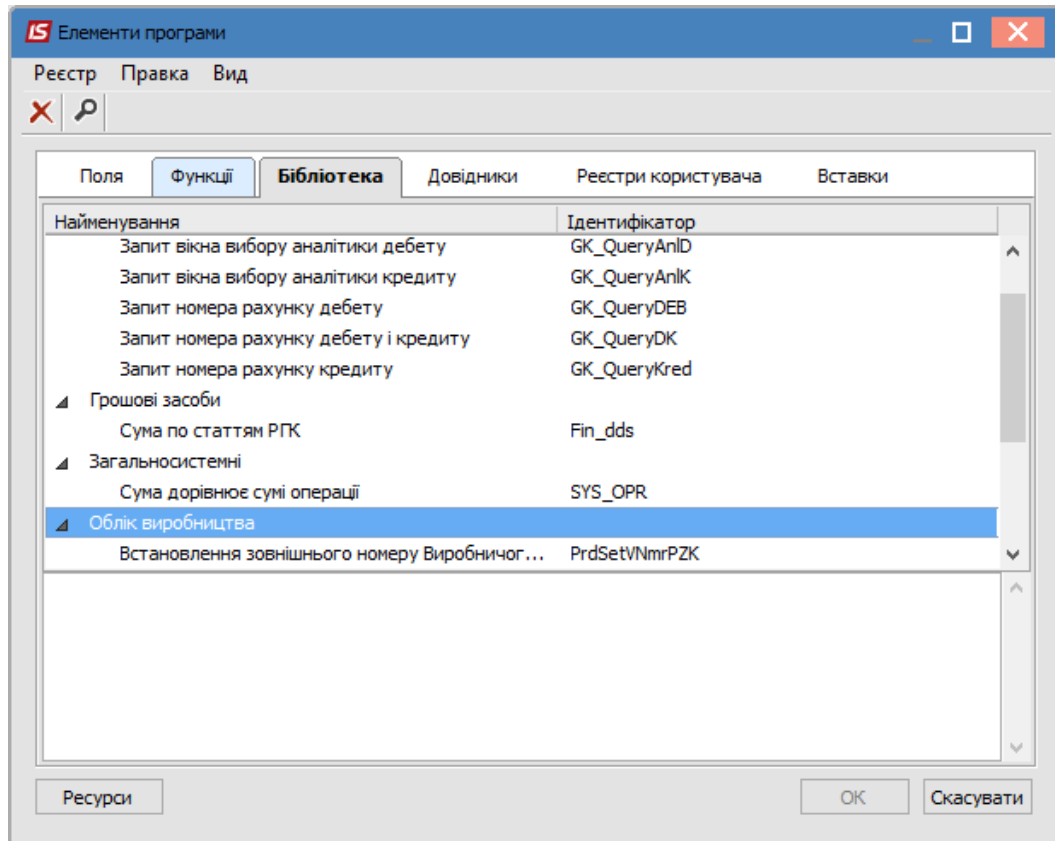


Рис. 2 – Редактор програм

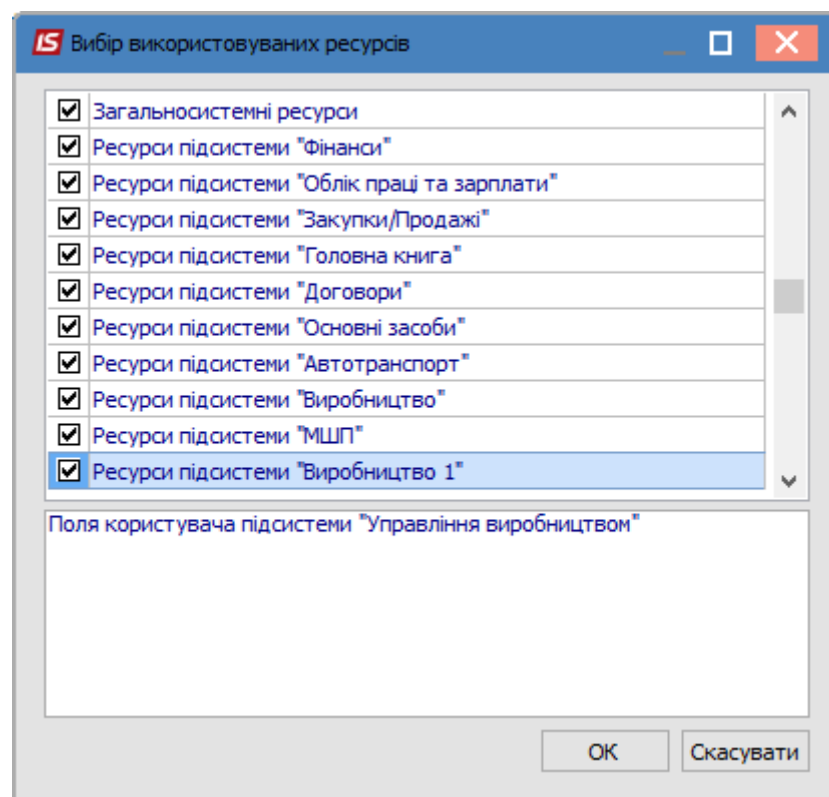
Перевірка правильності написання програми, виявлення помилок виконується по пункту меню **Програма / Перевірка**, або по комбінації клавіш **Ctrl + F9**.

1.3 Елементи програми

При написанні програм зручно використовувати вбудований довідник **Елементи програми**, що містить закладки **Поля**, **Функції**, **Бібліотека**, **Довідники**, **Реєстри користувача**, **Вставки**. Довідник елементів програм викликається з пункту меню **Правка / Викликати довідник**, або по клавіші **F3**. При завантаженні модуля завантажуються дані, що відносяться до даної підсистеми. У кожній підсистемі відображаються загальні дані і дані підсистеми. По кнопці **Ресурси** проводиться під'єднання полів і функцій інших підсистем.



а)



б)

Рис. 3 – Вікно Елементи програми (а) і підключення ресурсів інших підсистем (б)

1.3.1 Поля

Розділ довідника **Поля** містить список параметрів, доступних для налаштування даного алгоритму. Для зручності перегляду всі параметри розділені на кілька груп. Набір груп і параметрів, що входять в ці групи, залежить від підсистеми і призначення настановленого алгоритму. Так, наприклад, група параметрів **Вбудовані поля** пропонується для настройки проводок в різних підсистемах. При налаштуванні різних форм звітності пропонуються групи **Системні поля**, **Стандартні поля**, **Реквізити партії**, **Реквізити фірм**, **Користувальницькі поля звіту**, **Аліасні поля звіту**, **Поля користувача** і ін. При цьому набір параметрів першої групи постійний, в другу групу включені параметри, які характерні тільки для даного виду звіту. В розділ **Користувальницькі поля** звіту можна самостійно ввести новий параметр. У цьому випадку буде потрібно визначити алгоритм розрахунку даного параметра.

Для кожного поля в довіднику показані **Найменування**, **Ідентифікатор** і **Тип**. Для написання алгоритмів використовується **Ідентифікатор**, текстове **Найменування** тільки пояснює призначення даного параметра. **Тип** поля може бути: NUM - числове, CHAR - символне, DATE - дата, TIME - час. Необхідно стежити за тим, щоб у виразах брали участь параметри одного типу. При необхідності використовуються функції перетворення типів. Формат поля DATE – дата «МІСЯЦЬ / ДЕНЬ / РІК».

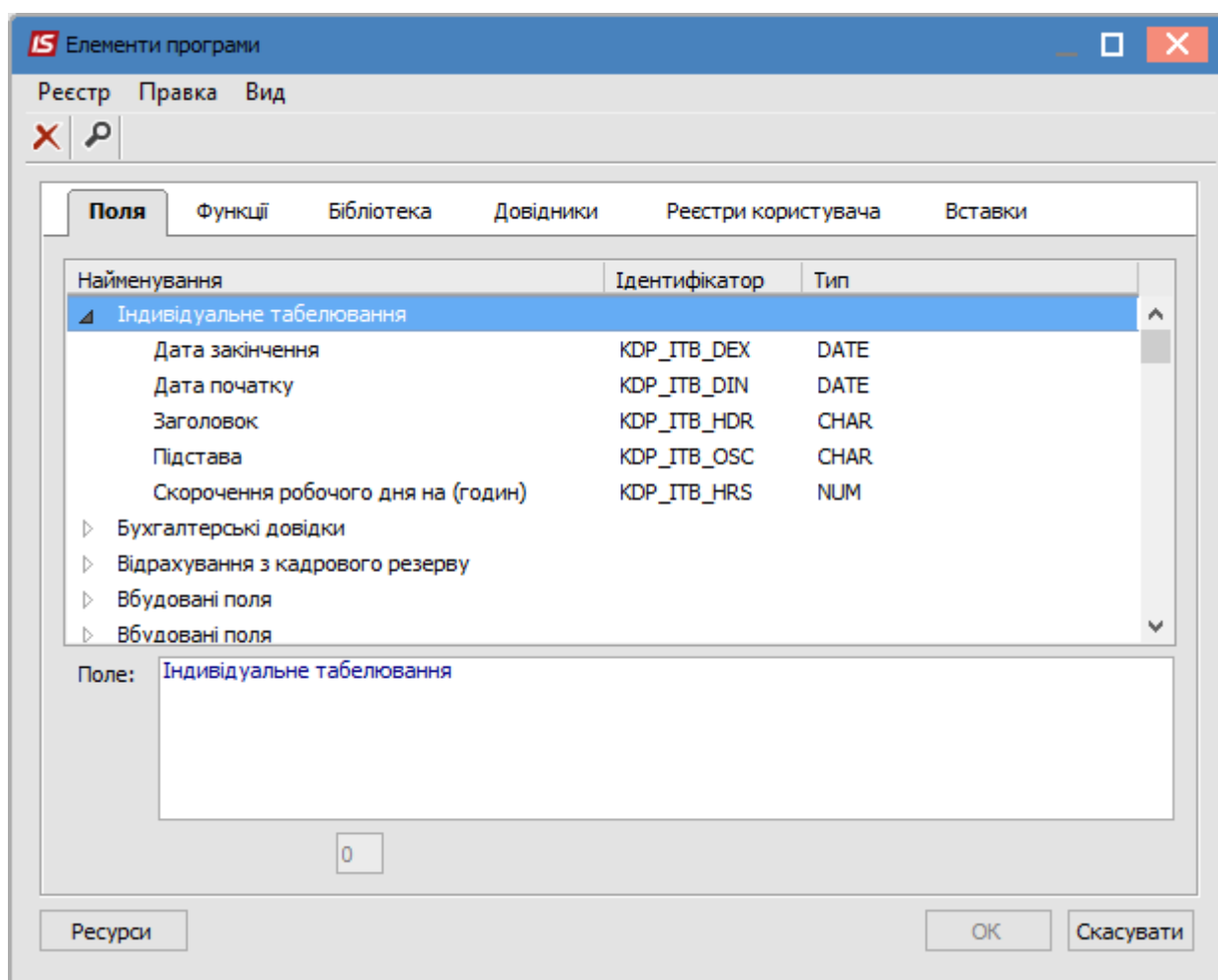


Рис. 4 – Поля

Довідник елементів програм містить досить велику кількість полів, тому важлива функція пошуку полів, що викликається по клавіші **F7**. Ви можете шукати за назвою, за ідентифікатором з урахуванням або без урахування регістру. Для пошуку необхідно встати на початок списку, так як пошук здійснюється один раз від поточного поля вниз по списку. Здійснивши пошук, курсор встановлюється на наступне поле після знайденого.

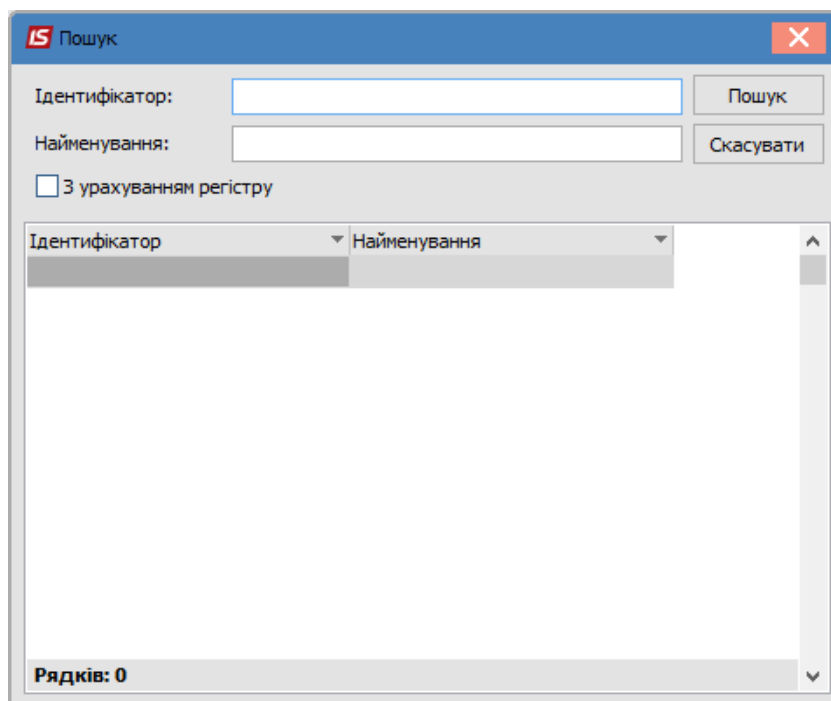


Рис. 5 – Діалогове вікно Пошук поля

Можливо введення частини ідентифікатора, а відсутні символи замінюються зірочкою. Наприклад, для пошуку поля **Сума проводки**, вибирається спосіб пошуку **По найменуванню** та вказується в поле **Найменування**: Сума *. При цьому команда пошуку знайде перше поле, що містить зазначений набір - Сума проводки. Пошук всіх полів, що задовольняють вказаній вимозі, проводиться по комбінації клавіш **Alt + F7**. Таким чином, знаходяться послідовно поля **Сума операції**, **Сума комісії**. Перелік полів наводиться в документі **перелік полів системи** (файл poly7xxxx.xls), де 7xxxx- індекс версії станом на яку створено файл.

1.3.2 Функції

Довідник функцій **IS-pro** знаходиться на закладці **Функції** довідника елементів програм.

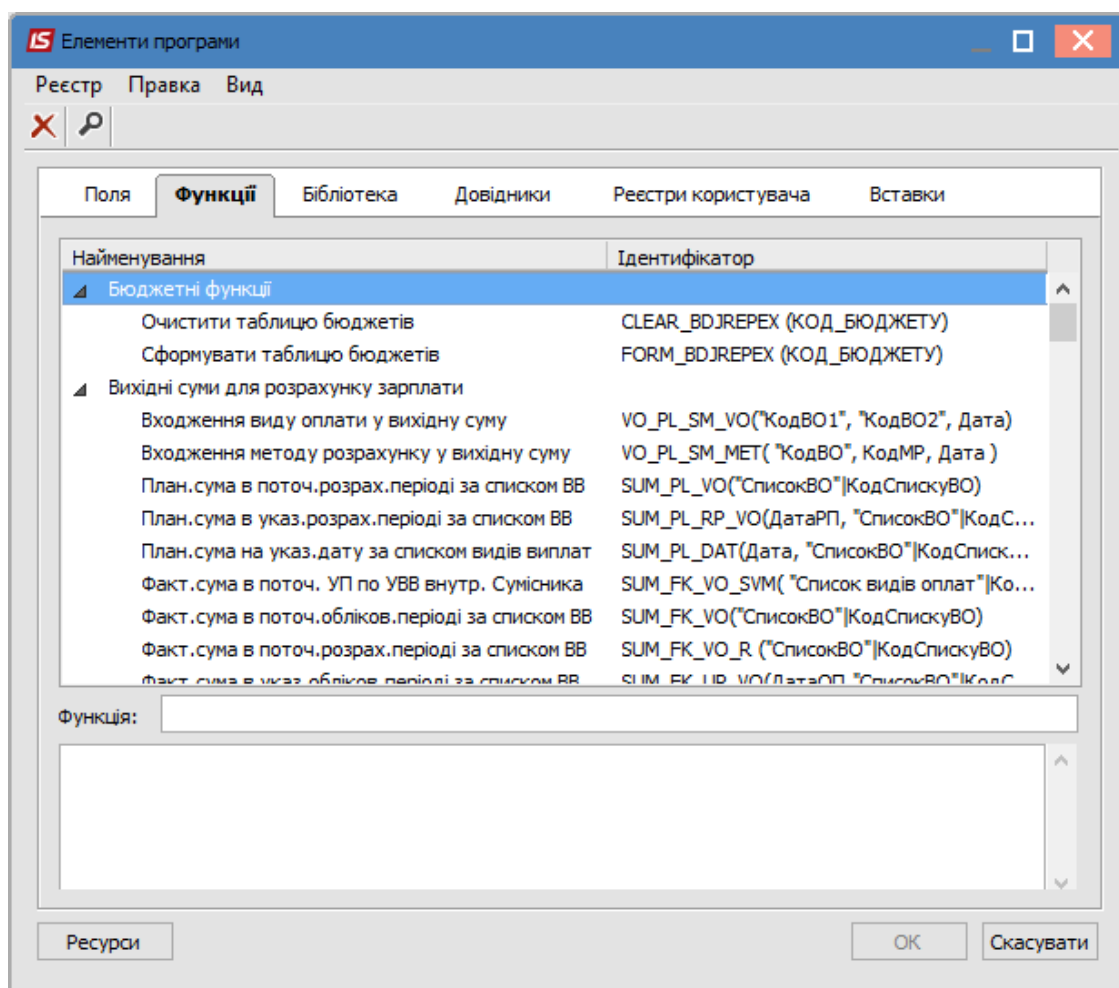


Рис. 6 – Довідник функцій

Загальносистемні функції доступні у всіх підсистемах. Функції для полів користувача доступні у всіх підсистемах.

Комплекс містить вбудовані функції, які застосовуються для перетворення дат і валют, генерації проводок, розрахунку оборотів, балансу і аналізу по аналітичним картками і так далі. У нижній частині розділу знаходиться підказка по функції, на якій встановлений курсор - формат функції і короткий коментар. Перелік функцій наводиться в документі **перелік функцій системи** (файл funkc7xxxx.xls), де 7xxxx- індекс версії станом на яку створено файл.

1.3.3 Бібліотека

На закладці **Бібліотека** розташовані найменування підсистем і ідентифікатори функцій, які можна використовувати при написанні програм користувача.

Шаблон обраної функції (текст програми) відображається в нижній частині вікна і може бути вставлений в призначену для користувача програму натисканням на клавішу **Enter** або натисканням кнопки **ОК**.

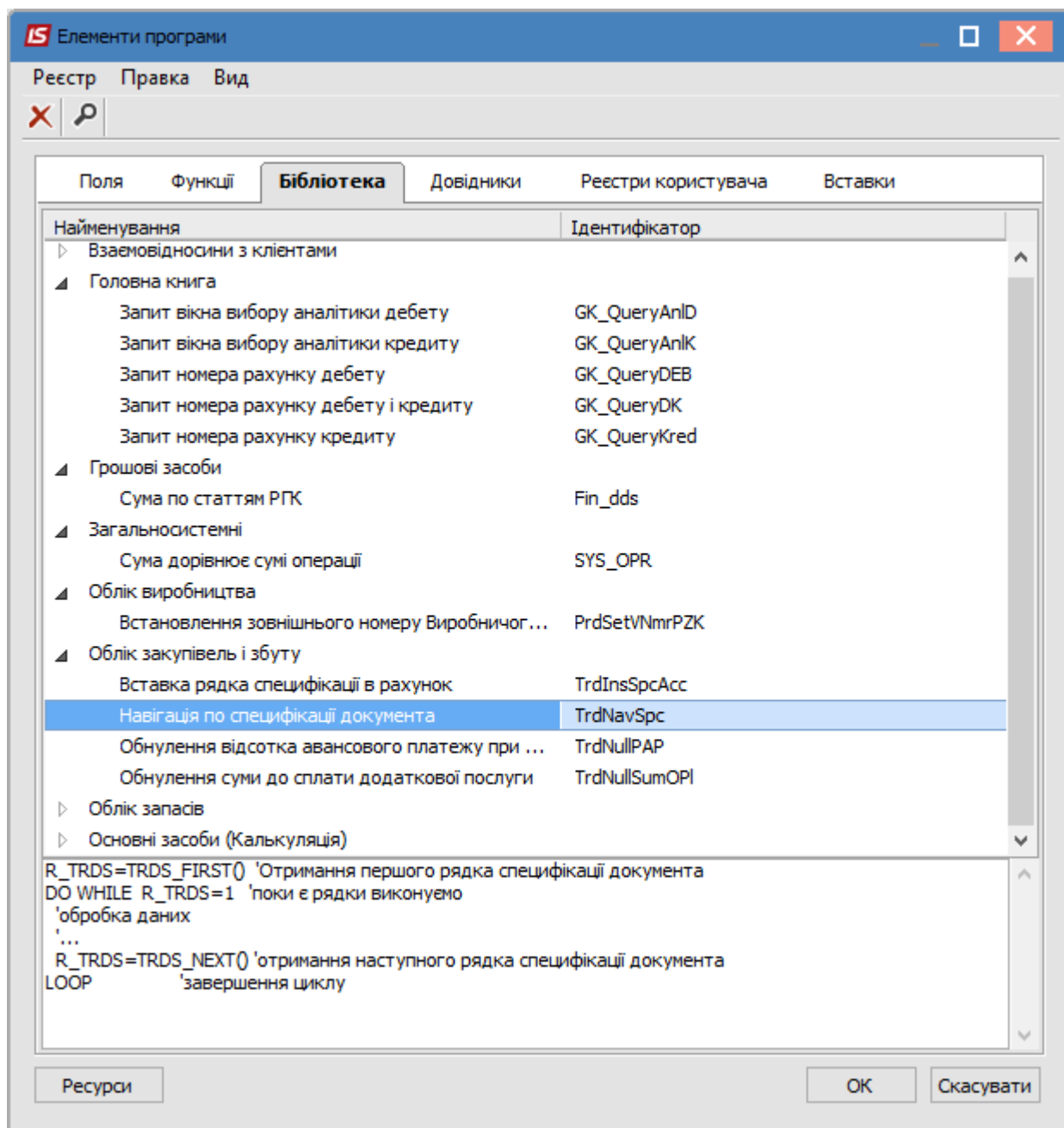


Рис. 7 – Бібліотека

1.3.4 Довідники

На даній закладці розташовуються Системні, Аналітичні і Користувальницькі довідники системи.

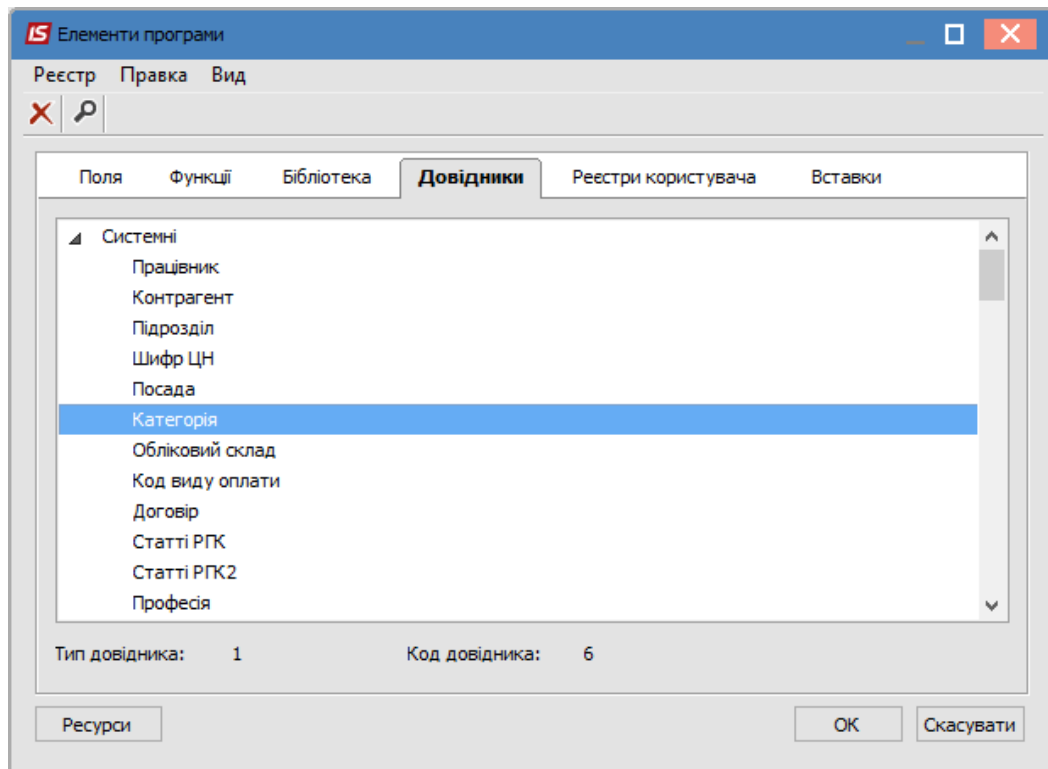


Рис. 8 – Довідники

1.3.5 Реєстри користувача

На даній закладці відобразяться **Реєстри користувача** системи.

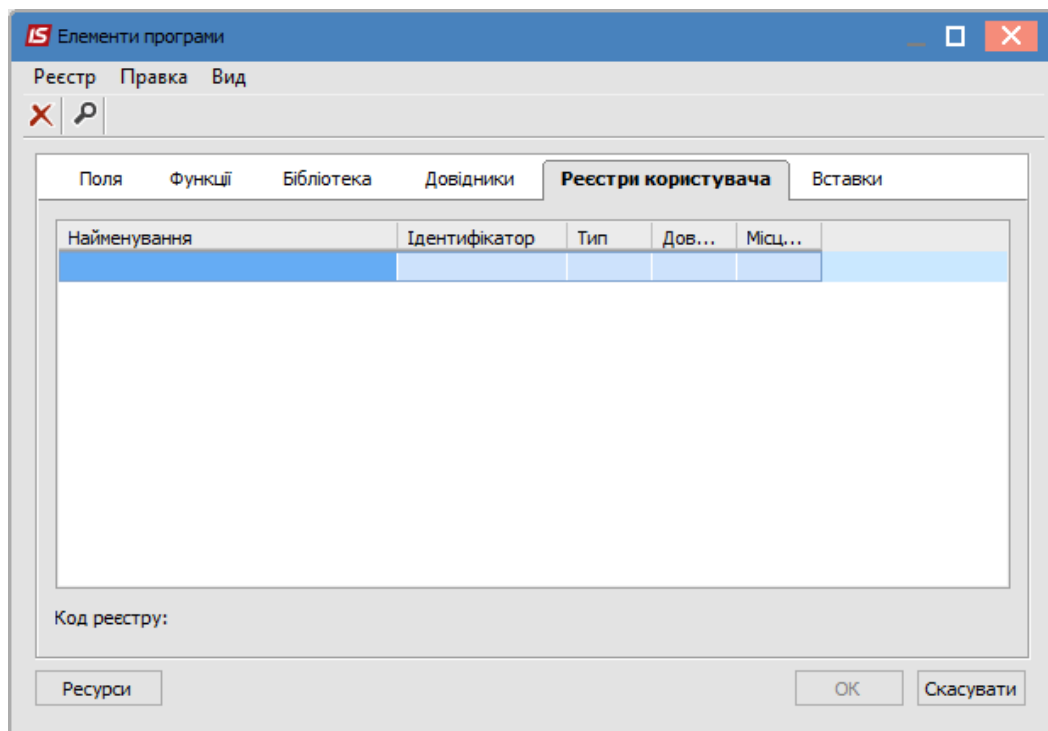


Рис. 9 – Реєстри користувача

1.3.6 Вставки

На даній закладці є підключення своїх текстів бібліотек до програм типових операцій, калькуляцій та розрахункових таблиць аналогічно `#include file_name` з C ++. Файли з текстами бібліотек для підключення до програм зберігаються в серверній частини системи **IS-pro**, в каталозі **USER_DATA \ INCLUDE**. Файли мають розширення **VBA**.

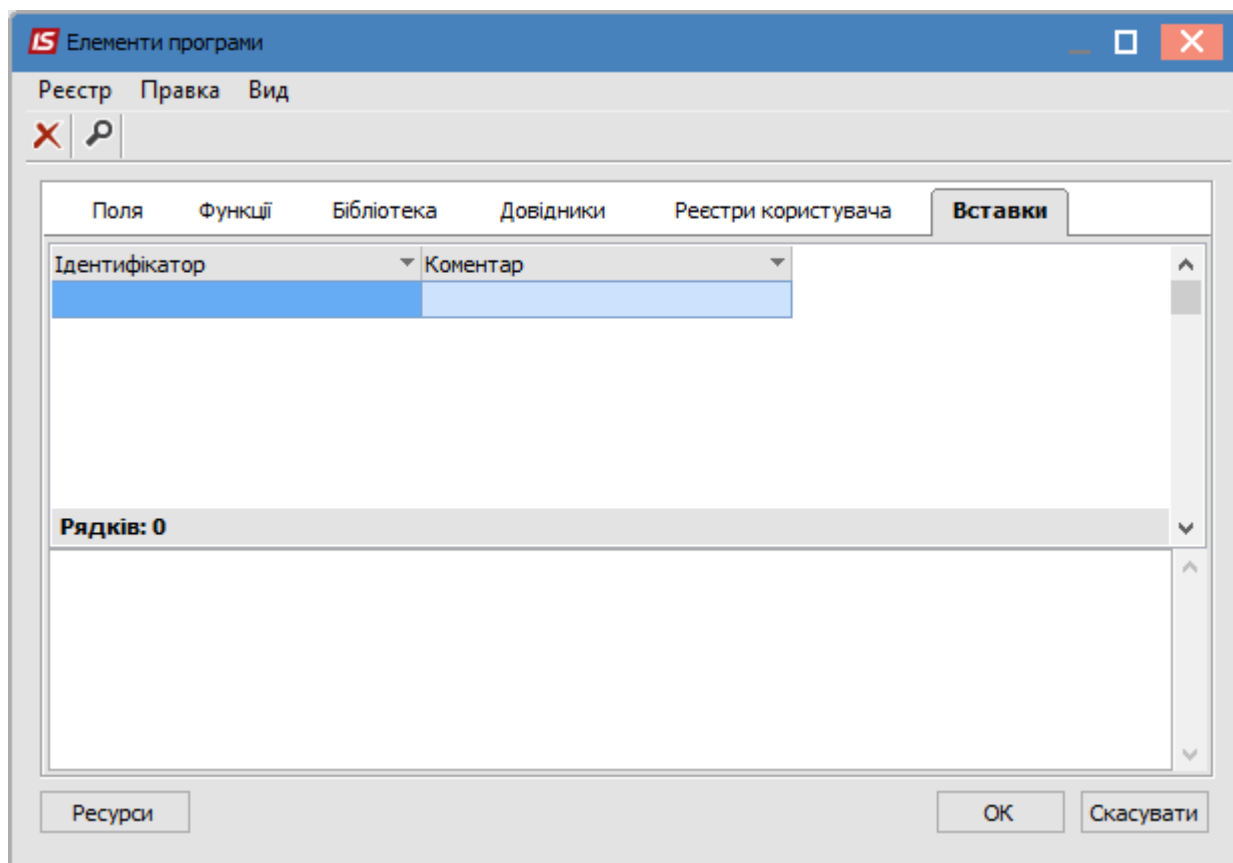


Рис. 10 – Вставки

2 Засоби створення вихідних форм

Система дозволяє самостійно створювати вихідні форми, використовуючи як вбудований інструментарій підготовки вихідних форм, так і стандартні засоби настройки і генерації звітів. При цьому для розробки нової вихідної форми за основу можна взяти вже існуючу форму.

Система містить дизайнер звітів - Генератор FastReport. Заснований на SQL-запитах, які відбирає дані з таблиць системи. Перелік і наповнення таблиць представлений в додатках. При розробці вихідних форм за допомогою такого генератора потрібен досвід написання SQL-запитів і інформація про базу даних системи.

Крім перерахованих коштів можуть бути використані продукти сторонніх розробників (наприклад, Crystal Reports), що перебувають в обігу до бази даних системи через SQL-запити. Також формування звітів може бути реалізовано за допомогою об'єктів доступу до даних системи (LAO) і продуктів Microsoft Office (Word, Excel).

2.1 Загальні відомості для створення і коригування, перевірки звітів

Активний список шаблонів звітів налаштовується, використовуючи редактор меню. Виклик редактора меню звітів проводиться з вікна **Вибір вихідної форми** або **Реєстр вихідних форм** по комбінації клавіш **Ctrl + F4** або командою меню **Реєстр / Меню...**

Реєстр **Налаштування меню звітів** містить дві панелі: **Всі звіти** та **Обрані звіти**, які розташовуються вертикально або горизонтально (уявлення налаштовується по пункту меню **Вид / Горизонтальний / Вертикальний**). На панелі **Всі звіти** представлений повний список звітних форм, доступних в поточному модулі - поставляються розробником і налаштованих користувачем. Панель **Вибрані звіти** містить меню звітів - активний список звітних форм, що відображається у вікні **Вибір вихідної форми**. Копіювання необхідних звітів в активному списку проводиться по клавіші **F5**. Шаблони, перенесені на панель **Обрані звіти** (тобто включені в меню звітів), перейменовуються (клавіша **F4**), переміщуються в списку вгору або вниз (клавішами **Ctrl + ↑ / Ctrl + ↓** упорядковано за ознаками, формуючи групи звітів. Для цього вводяться заголовки груп (по клавіші **Ins**), а потім необхідні звіти копіюються в ці групи (по клавіші **F5** зі списку панелі **Всі звіти**).

Звіти, створені користувачами, видаляються по клавіші **F8**. При видаленні звіту з панелі **Всі звіти** він автоматично видаляється з **Вибраних звітів**. Звіти, що поставляються в системі (в меню виділені чорним кольором) видалити не можна.

Після додавання файлу з шаблоном в каталог **REP** (знаходиться на серверній частині) необхідно перебудувати список звітів по комбінації клавіш **Alt + S** або через пункт меню **Реєстр / Перебудувати список звітів**, після чого шаблон з'являється на панелі **Всі звіти**.

У налаштуванні програм загальносистемні функції розділені по групах для зручності пошуку.

2.1.1 Виклик меню звітів

Для генерації звіту або друкованої форми первинного документа необхідно викликати меню звітних форм, доступних в даному модулі. Меню звітів викликається наступними способами:

- З будь-якого первинного документа по клавіші **F9** або за пунктом меню **Документ / Друк** викликається вікно **Вибір вихідної форми**, в якому запропоновано форми для друку поточного документа. Або з реєстру документів по комбінації клавіш **Ctrl + F 9** або командою меню **Реєстр / Друк документа**.
- З будь-якого реєстру по клавіші **F9** або командою меню **Реєстр / Друк** викликається вікно **Вибір вихідної форми**, в якому запропоновано форми для друку поточного реєстру.
- За комбінації клавіш **Alt + F9** або командою меню **Звіт / Формування звіту** (якщо в даному реєстрі такий пункт є) викликається вікно **Вибір вихідної форми**, в якому запропоновано форми для друку різних звітів за даними поточної підсистеми.
- Практично в кожній підсистемі пропонується модуль **Звіти**, при завантаженні якого відкривається вікно **Реєстр вихідних форм**.

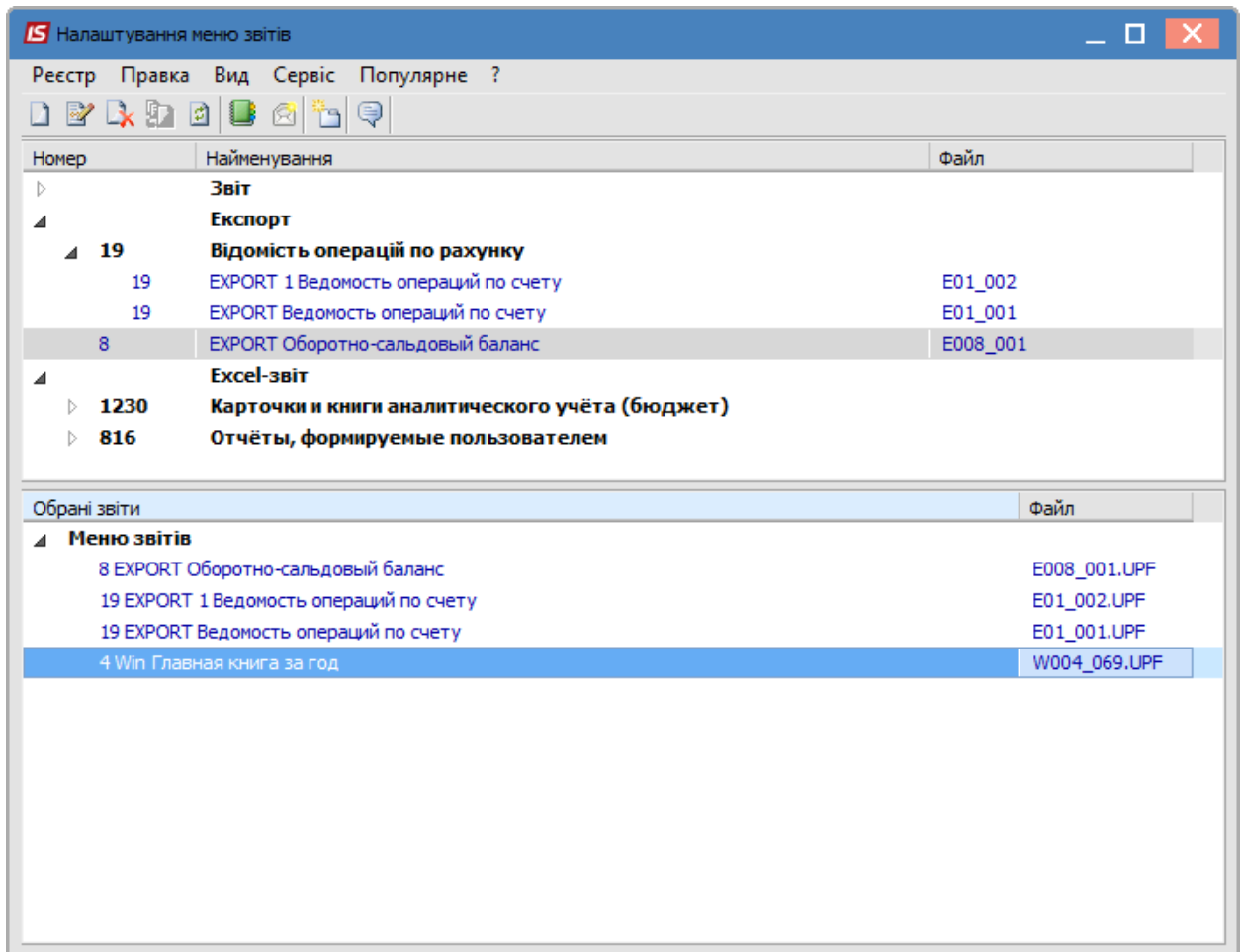


Рис. 11 – Работа с редактором меню звітів

Перехід з будь-якого вікна **Вибір вихідної форми** до вікна **Реєстр вихідних форм** проводиться по клавіші **F3**.

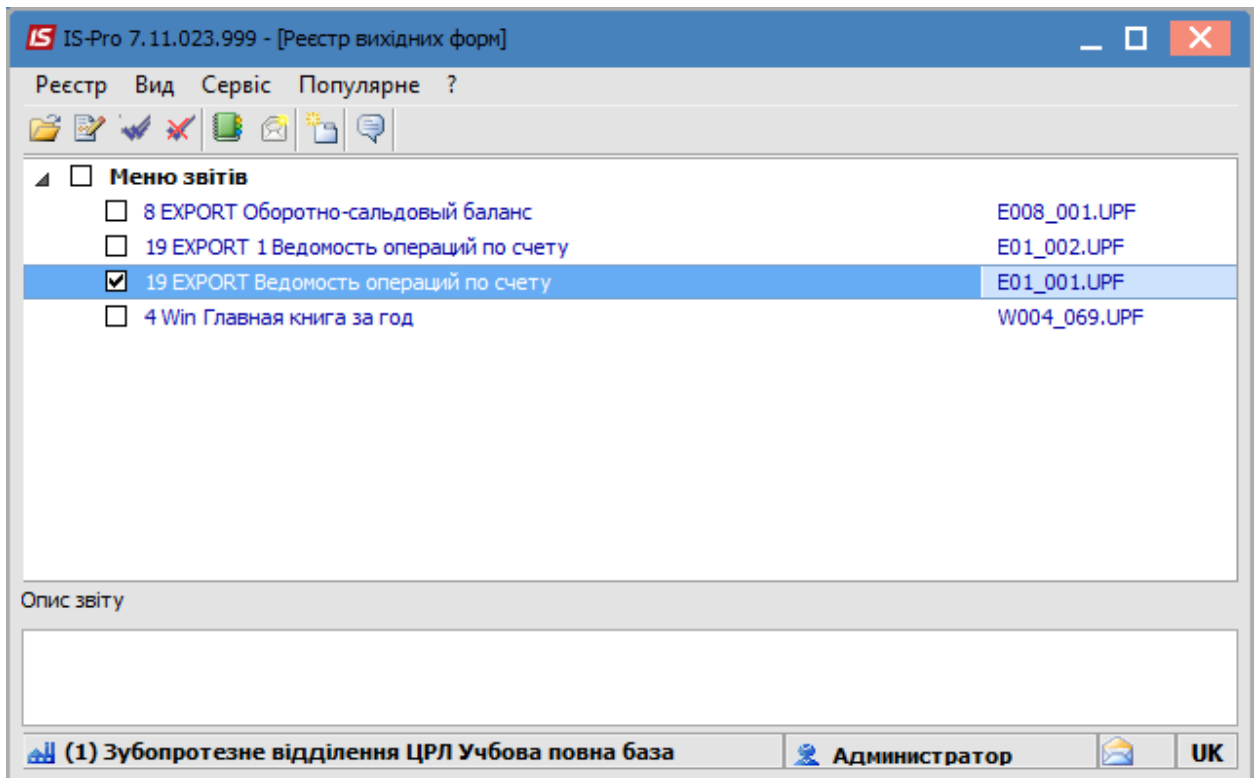


Рис. 12 – Вікно вибору вихідної форми

Реєстр вихідних форм складається з двох горизонтальних панелей. На верхній панелі знаходиться активний список доступних шаблонів. На нижній панелі відображається коментар до звіту. Для шаблонів, створених користувачем (в меню виділені синім кольором), по клавіші **F4** проводиться редагування.

2.1.2 Імпорт-експорт шаблонів

Імпорт/експорт шаблонів звітів може використовуватися, наприклад, в разі, коли шаблон створений програмістом необхідно передати в робочу систему.

Імпорт шаблонів здійснюється у вікні **Налаштування меню звітів** при активній панелі **Всі звіти** по пункту меню **Реєстр / Імпорт** або по комбінації клавіш **Alt+I**. У вікні **Імпорт шаблонів** вказується ім'я каталогу, в якому знаходиться імпортований звіт. У списку файлів з шаблонами по клавіші **Пробіл** виділяється необхідний.

Експорт шаблону здійснюється по комбінації клавіш **Alt+E**. У вікні **Експорт шаблону** вказується шлях до каталогу, в який буде поміщений файл шаблону.

2.1.3 Генерація звітів

Для генерації звіту або друкованої форми документа викликається вікно **Вибір вихідної форми** або **Реєстр вихідних форм**, вибирається найменування необхідного звіту і натисканням клавіші **Enter** підтверджується вибір.

Якщо відобразиться діалогове вікно для уточнення додаткових параметрів звіту (наприклад, період або валюта звіту) заповнюються або вибираються з довідників необхідні дані. Параметри звіту можуть бути обов'язковими або необов'язковими. До тих пір, поки не заповнені обов'язкові параметри, кнопка **ОК** залишається неактивною.

Необов'язкові параметри призначені для різних вибірок (наприклад, вибір по контрагенту). Якщо параметр не заповнений, звіт виводиться для всіх значень даного параметра (в даному випадку - по всіх контрагентах).

Після обробки даних з'являється вікно **Генератора звітів** з готовим звітом - текстовим або графічним, в залежності від виду обраного шаблону.

2.1.4 Перегляд і друк звітів.

Генератор звітів відкривається спільно з вікном **IS-pro**. Тому, сформувавши звіт, його переглядають, і, при необхідності, є можливість переключитися в вікно **IS-pro**, запустити формування іншого звіту і знову повернутися в графічний генератор звітів. Генератор підтримує багатовіконність, тому можливий перегляд в ньому одночасно декількох готових звітів.

Графічний **Генератор звітів** не просто забезпечує власне генерацію звітів за готовими шаблонами з використанням призначених для користувача даних. З багатифункціонального інтерфейсу **Генератора звітів** доступні операції збереження сформованого звіту, відкриття будь-якого звіту з архіву звітів для перегляду, експорту звіту в формати Word і Excel документів, друку всього звіту, або окремих його частин.

Підприємство: Зубопротезне відділення ЦРЛ Учбова повна база

Головна книга
По рахунку 10 Основні засоби
За 2017 рік.

Місяць	Кор. рахунок	Обороти		Сальдо	
		Дебет	Кредит	Дебет	Кредит
На 01.01.2017				164'320.00	
Січень				164'320.00	
Лютий				164'320.00	
Березень				164'320.00	
Квітень				164'320.00	
Травень				164'320.00	
Червень				164'320.00	
Липень				164'320.00	
Серпень				164'320.00	
Вересень				164'320.00	
Жовтень				164'320.00	
Листопад				164'320.00	
Грудень				164'320.00	
Всього за	2017 рік			164'320.00	

Головний лікар: Шинкаренко Валентин Петрович
Головний бухгалтер: Шестопал Олеся Василівна

Страница 1/19

Рис. 13 – Графічний генератор звітів

Якщо осередок був визначений як редагований, то на екрані його вміст виділяється синім кольором.

Інтерфейс графічного **Генератора звітів** дозволяє редагувати текстову і графічну інформацію. Для цієї мети в звіт вбудована панель, що забезпечує основні функції редагування і викликається по кнопці **Виклик редактора** панелі **Вид**.

Існує можливість змінити шрифт і накреслення символів, орієнтир для вирівнювання рядків або чисел, обрамлення навіть для тих осередків звіту, які не були зазначені як редаговані при налаштуванні шаблону звіту. При цьому доступний весь набір Windows - шрифтів, встановлених на комп'ютері. При необхідності повністю видаляються окремі фрагменти звіту. Виділивши в осередку окремі символи, копіюють в буфер обміну, вирізають або видаляють їх, скориставшись контекстним меню, що відкривається натисканням правої кнопки миші. Символи з буфера обміну вставляються в інший фрагмент звіту, за допомогою того ж контекстного меню. Необхідні ділянки тексту звіту знаходять з використанням функції **Пошук тексту**, доступної через пункт меню **Вікно** або по клавіші **F7**.

Для друку звіту використовується панель **Друк**. На відміну від друку текстових звітів, при друку графічних звітів спеціальні настройки принтерів в **IS-pro** не мають значення, так як використовуються стандартні настройки Windows.

Якщо осередки шаблону виходять за робочу область листа, то при друку генератор видає попередження про те, що звіт буде надрукований в повному обсязі.

Встановлення шрифту для друку штрих-кодів в звітах.

Для встановлення шрифту досить перевстановити робочу станцію (після встановлення пакету) або вручну встановити в системі шрифт EanGnivc.ttf. Шрифт знаходиться в папці WIN пакета. Встановлення шрифту в Windows: Пуск (Start) / Налаштування (Settings) / Панель управління (Control Panel) / Шрифти (Fonts) / Файл (File) / Встановити шрифт (Install New Font).

2.1.5 Експорт в звітах

У генераторі звітів **FastReport** реалізована можливість експорту готового звіту в **Microsoft Excel** або **Microsoft Word**. Для цього призначена панель **Експорт**. При експорті в **Microsoft Word** текст поміщається в таблицю, рядки та клітинки якої відповідають рядкам і осередкам в звіті. Файл зберігається у форматі **RTF**. При експорті в **Excel** всі осередки експортуються в загальному форматі.

Всі звіти, при необхідності, експортуються в **M.e.doc** (функціями передачі даних для експорту).

2.1.6 Архів звітів

Якщо в налаштуванні шаблону звіту був встановлена ознака **Розміщувати в архів** або **Видавати запит**, то при поверненні з режиму перегляду звіту в попередній режим, сформований звіт буде збережений в архіві звітів відповідно автоматично або з попереднім запитом.

Звіти, поміщені в архів звітів, проглядаються і виводяться на друк таким же чином, як і при їх генерації.

Доступ до архіву звітів здійснюється з пункту меню **Сервіс / Архів звітів** або по комбінації клавіш **Shift+F9** в будь-якому модулі. Програма виводить список всіх сформованих і збережених в архіві звітів із зазначенням найменування звіту, часу його

формування та довжини звіту в листах. По клавіші **Enter** викликається режим перегляду і друку звіту.

Архів звітів забезпечений функцією сортування по найменуванню, дату і час. Для включення функції сортування з пункту меню **Вид архіву звітів** вибирається пункт **Сортування** і спосіб сортування (**Код, Найменування** або **Дата, час**). Список звітів в архіві сортується за вказаною параметру. Непотрібні звіти видаляються по клавіші **F8**.

2.1.7 Шаблони

2.1.7.1 Створення користувацьких шаблонів

Разом з системою **IS-pro** розробниками поставляються шаблони основних звітів і друкованих форм. Набір цих шаблонів, як правило, достатній для початку роботи. Однак для обліку специфіки конкретного підприємства, необхідно налаштовувати готові і створювати нові шаблони. Новий шаблон створюється через **Налаштування меню** і на активній панелі **Всі звіти** по клавіші **Ins** (або по пункт меню **Реєстр / Створити**). У вікні **Створення форми звіту**:

Вибирається один з можливих **Типів** (**Експорт, FastReport -Звіт, XML - звіт, Excel - звіт, Word -Звіт, Звіт OLAP** і звіт и Excel 2007,2010)

2.1.7.2 Початок створення звіту

Звіт - вибирається вид створення звіту - **Звіт Fast Report**

З запропонованого списку (викликається по клавіші **F3**) вибирається **Шаблон**. Список можливих шаблонів залежить від підсистеми і модуля, де проводиться настройка звіту. Важливо зробити правильний вибір, оскільки саме обраний шаблон визначає спосіб складання звіту, список доступних даних і набір вбудованих запитів.

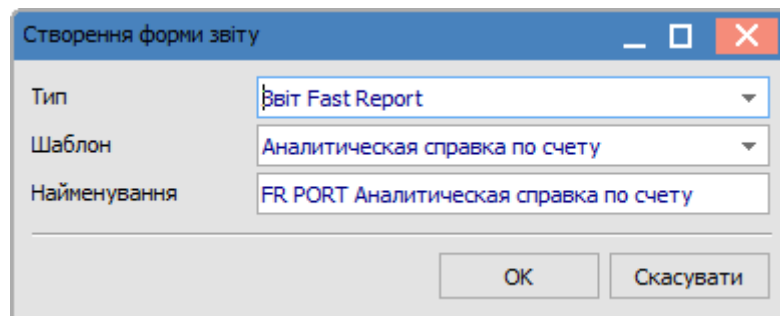


Рис. 14 – Створення нового шаблону

За кнопці **ОК**, в залежності від обраного **Типу**, запускається текстовий або графічний дизайнер звітів або ж спеціальний режим настройки експорту даних (див. Відповідні розділи опису).

За комбінація клавіш **Shift + Ins** при використанні елементів готового шаблону, курсор встановлюється на звіт, який слід взяти за основу (або вибирається пункт меню **Реєстр / Породити...**). Вибирається **Тип** звіту, який необхідно породити за зразком вибраного шаблону, вказується його **Найменування**. Всі елементи шаблону-зразка копіюються в новий шаблон. Далі в ньому проводяться необхідні зміни.

Виклик **Дизайнера звітів** для редагування готового шаблону здійснюється з панелі **Всі звіти** в **Настройці меню звітів** по клавіші **F4** або з вікна **Вибір вихідної форми** по

клавіші **F4**. Редагуються (і видаляються) тільки шаблони звітів, налаштовані користувачем. У списку шаблонів вони виділені синім кольором. Шаблони, що поставляються розробником (в списку відзначені чорним кольором), не коригуються і не видаляються.

2.1.7.3 Основні принципи створення шаблонів

Для того щоб навчитися проектувати шаблони, необхідно представляти структуру шаблону і процес формування звіту з шаблону.

Завдання звіту - вивести сукупність даних, відповідних даної задачі, в наочній, зручною для аналізу формі.

2.1.7.3.1 Фрагменти звіту

Шаблон звіту складається з окремих фрагментів, в які можуть бути включені поля бази даних. Кожен з фрагментів виконує певне завдання в процесі генерації звіту і представлений в звіті певним числом рядків. Фрагменти розділяються на типи: **Шапка** і **Підніжжя** друкованого аркуша, **Заголовки**, **Підсумки** і **Детальні**.

При проектуванні шаблону фрагменти можуть бути об'єднані в **Групи** з рівнями підпорядкування, які утворюють ієрархічну структуру. У кожену групу безпосередньо включається довільне число фрагментів, крім детальних. При цьому в групі зобов'язаний бути присутнім хоча б один детальний фрагмент або підпорядкована група.

1. Детальний рядок

Фрагмент, який відповідає певній групі записів бази даних, визначених для даного виду шаблону розробником. Кожному шаблону відповідають свої детальні рядки, які включають в звіт. Їх кількість обмежена, вони мають свої назви, які визначаються шаблоном. Кожному детальному рядку відповідають свої поля, які будуть виводитися тільки в цьому рядку.

Фрагмент детальної інформації в залежності від найменування виводиться стільки разів, скільки записів зберігається в БД.

Розділ бази даних, відповідний призначенням звіту визначається розробником.

2. Тема

Фрагмент для виведення заголовка таблиці і підзаголовків груп даних. Фрагмент заголовка видається при зміні значень полів, визначених як ключі групи, в яку вони входять.

Фрагмент **Тема** виводиться на початку своєї групи перед **Детальними** фрагментами.

3. Шапка

Фрагмент **Шапка** будь-якої групи завжди виводиться на початку листа і призначений для оформлення верхніх колонтитулів. Якщо видача фрагмента відбувається в середині або кінці листа, фрагмент буде надрукований на новій сторінці. Йому можуть передувати тільки фрагменти **Шапка** охоплюють груп.

4. Підсумок

Фрагмент, який виводиться після кожної групи даних або в кінці всього звіту, використовується для формування підсумкових значень. Видається при зміні значень полів, визначених як ключі групи, в яку він входить.

5. Підніжжя

Фрагмент **Підніжжя** виводиться в кінці листа, призначений для оформлення нижнього колонтитула або кінця сторінки. Після нього можуть слідувати тільки фрагменти **Підніжжя** охоплюють груп.

2.1.7.3.2 Угрупування даних

Дані, що виводяться в детальний фрагмент звіту, групуються за певною ознакою (**ключу**) таким чином, що кожна група містить всі записи при певному значенні **ключа**. Наприклад, угруповання ТМЦ по складах, так, щоб в заголовку кожної групи стояла назва складу, в групі виводилися записи, відповідні поточному складу, а в підсумковому рядку - підсумкові дані по складу. Механізм угруповання реалізований наступним чином. В процесі проектування шаблону, при створенні групи, для неї визначається так званий ключ закриття групи - сукупність найменувань полів, за якими повинна проводитися угруповання (в нашому прикладі - код складу). У процесі виведення **Детального** фрагмента проводиться перевірка значення ключа. Якщо при виведенні чергового запису значення ключа змінилося, то це призводить до завершення групи і висновку до звіту підсумку по групі. Якщо вказано кілька ключів групи, то, перш за все, дані розбиваються по першому значенню ключа, потім всередині цього розбиття відбувається угруповання по другому ключу і так далі.

2.1.7.3.3 Табульовані осередки

Сукупність даних, що виводиться в детальний фрагмент, може мати змінну структуру. Наприклад, кожної товарно-матеріальних цінностей на складі можуть бути присвоєні додаткові характеристики (країна-виробник, матеріал, марка, сорт і так далі), причому для різних груп ТМЦ кількість і значення цих характеристик можуть бути різними. Таким чином, з'являється необхідність формування в таблиці даних звіту додаткових (табульованих) колонок з найменуваннями і значеннями, залежними від групи даних, що виводяться. Це реалізується в такий спосіб: в шаблоні звіту для груп, в яких міститься детальний фрагмент з полями змінної структури, визначається так званий ключ табуляції - найменування цих **полів**. Ключ табуляції поширює свою дію на всі групи підлеглого рівня. **Генератор звітів** виводить поля ключа табуляції в послідовні, зміщені вздовж рядка, значення, так що в результаті утворюються необхідну кількість додаткових колонок.

Подробиці створення табульованих осередків в звіті див. у описі дизайнера.

2.1.7.3.4 Сорткування даних

Користувач має можливість також задати сортування даних за певним параметру (Наприклад, здійснити сортування за кількістю або за ціною). Це здійснюється зазначенням ключа сортування при проектуванні шаблону.

Подробиці створення угруповань див. у описі дизайнера.

2.1.7.3.5 Програми користувача

Для створення простого звіту досить налаштувати структури звіту, ввести його зовнішній вигляд і визначити поля виведення даних. Якщо в формі звіту застосовуються реквізити користувача або потрібна більш складна настройка, наприклад, завдання власних запитів, фільтрів, сортування, необхідні алгоритми налаштовуються за допомогою програм користувача.

Три види програм мають фіксовані імена: **Ініціалізація** (Програма ініціалізації звіту), **Обробка сортування користувача**, (Програма фільтрації і сортування звіту) і **Завершення** (Програма завершення звіту). Ці програми не вимагають підключення до фрагментів звіту.

Призначені для користувача процедури, задані в програмі під ім'ям **Ініціалізація**, викликаються перед початком формування звіту. Програма може використовуватися для установки початкових значень полів або опису функцій для виведення додаткових запитів перед формуванням звіту.

Процедури програми **Завершення** виконуються на стадії завершення звіту. Вони можуть бути використані, наприклад, для виведення будь-яких діагностичних повідомлень за результатами формування звіту.

У програмі з найменуванням **Обробка сортування користувача** задаються умови додаткової фільтрації даних, вона викликається при формуванні кожного запису тимчасової бази, якщо застосовується сортування користувача.

Всі інші процедури, зокрема, алгоритми розрахунку реквізитів користувача, накопичення підсумків і т.п. задаються в призначеній для користувача програмі з довільним найменуванням. Кількість призначених для користувача програм, заданих в одному шаблоні, не обмежена.

Для того щоб алгоритм користувацької програми виконувався при генерації звіту, вона обов'язково повинна бути підключена до фрагментів типу Детальна рядок, Підсумок або Підніжжя.

При написанні програм використовується спеціальна мова типу Basic (описана в п.2.3).

У програмі **Обробка сортування користувача** для фільтрації даних використовується спеціальний системний реквізит **REC_FILTER - Ідентифікатор відфільтрованої записи**. За замовчуванням значення реквізиту дорівнює 0, що означає включення запису в звіт. Для того щоб відфільтрувати запис, йому необхідно присвоїти значення 1. Насправді той же результат отримують, якщо в призначеній для користувача програмі використовувати реквізит **_RepSuppress / Ознака невидачі фрагмента на друк**. Однак, виділення умов відбору записів по можливості в окрему програму Обробка сортування користувача дозволяє підвищити наочність шаблону і полегшує його настроювання. Для програмного придушення виведення підсумків на друк необхідно в програмі користувача, прив'язаною до підсумкового фрагменту, встановити **_RepSuppress = 1**.

Приклад.

Для того, щоб в звіті по картотеці не виводити картки з нульовою кількістю, слід заповнити програму Обробка сортування користувача:

```
IF SPKrtQt = 0 THEN REC_FILTER = 1 ELSE REC_FILTER = 0
```

Якщо в даному шаблоні не планується введення власної нестандартної сортування, введіть програму користувача:

```
IF SPKrtQt = 0 THEN _RepSuppress = 1 ELSE _RepSuppress = 0
```

УВАГА! Не забувайте виконати прив'язку програми користувача до детального рядку!

Підключення призначених для користувача програм до фрагментів звіту

Призначені для користувача програми можуть підключатися до фрагментів типу Детальна рядок, Підсумок або Підніжжя.

Доступ до полів фрагментів останніх двох типів має деякі особливості і здійснюється через так звані аліасні поля.

Аліасні поля призначені для виведення у підсумкових фрагментах таких полів, значення яких розраховуються програмою користувача. Причому програма користувача прив'язана до підсумкового фрагменту або підніжжя і викликається під час їх виведення. Аліасні поля є, по суті, псевдонімами, які підставляються у формули замість полів підсумкового фрагмента або підніжжя. Ім'я створюваного аліасного поля формується з імені вихідного поля шляхом додаванням префікса " RA _". Ці поля слід використовувати в програмах для роботи з підсумками або підніжжям.

2.2 Дизайнер звітів FastReport

2.2.1 Етапи побудови звіту

Виділяють наступні компоненти процесу побудови типового звіту:



Рис. 15 – Етапи створення звіту

2.2.1.1.1 Дані

Являють собою сукупність даних, на основі яких будується звіт. Організація доступу до даних з БД здійснюється компонентами **FastReport** без участі програміста.

Крім даних, що зберігаються в БД, **FastReport** використовує будь-які джерела (масив, файл, вміст StringGrid і ін.). В цьому випадку програміст сам піклуватися про доступ до такої інформації. Для цього використовується набір подій, що дозволяють здійснити передачу даних в компоненти **FastReport**.

Крім доступу до даних, визначених у проекті, **FastReport** дозволяє створювати нові компоненти в реальному часі. У **FastReport** принципи створення компонентів доступу до даних максимально наближені до тих, що використовуються в середовищі Delphi. Так само, як і в Delphi, на форму кладеться компонент, і в інспекторі об'єктів налаштовуються його властивості.

2.2.1.1.2 Параметри

На цьому етапі здійснюється запит параметрів у користувача (наприклад, діапазон дат, за яким необхідно вивести дані). Деякі звіти обходяться без цього етапу або використовують фіксовану установку параметрів (без запиту їх значень в діалозі).

У **FastReport** є можливість запиту параметрів, якщо звіт використовує дані з запиту (Query). Для діалогу з користувачем використовується діалогове вікно. Крім того, для запиту параметрів використовується форма, розроблена в середовищі Delphi. Правда, при необхідності будь-яких змін в логіці роботи доведеться перекомпілювати проект.

FastReport, крім цього, дозволяє користувачеві самому розробляти форму діалогу. Процес нагадує побудова форми в середовищі Delphi: є набір стандартних елементів

управління, які розташовуються на формі діалогу і налаштовуються їх властивості. За допомогою вбудованої мови **FastReport** дозволяє реалізувати необхідну логіку роботи діалогу і передати введені значення ядру генератора.

Можливість створення власних діалогів дуже корисна - укупі з іншими можливостями (створення джерел даних, використання вбудованої мови) вона дозволяє створювати звіти, максимально відв'язаних від середовища Delphi. Це дозволяє створювати нові звіти і модифікувати існуючі без переписування або перекомпіляції проекту.

2.2.1.1.3 Форма

Форма звіту являє собою набір елементів, що описують вид готового звіту. Для угруповання елементів по їх розташуванню в готовому звіті застосовуються бенди (англ. Band - смужка). Бенди поділяються на два види: службові (заголовок звіту, сторінки та ін.) І бенди, що утворюють багаторядкову частину звіту (далі - дата-бенди). Дата-бенди підключаються до джерел даних, і їх вміст виводиться стільки разів, скільки є рядків даних в джерелі.

Для побудови форми звіту використовується візуальне середовище розробки - дизайнер / Інтерфейс дизайнера виконаний на сучасному рівні з використанням панелей інструментів (**toolbars**), розташування яких змінюється. Для зручності маніпуляції властивостями об'єктів звіту використовується інспектор об'єктів, аналогічний делфійському.

2.2.1.1.4 Обробка

Під обробкою розуміється обробка вхідних даних, модифікація форми звіту або окремих її компонентів в процесі побудови звіту. Найпростіший приклад такої обробки - висновок від'ємних сум червоним кольором. Більш складний приклад обробки - друк суми, яка підраховується в підвалі групи, в її заголовку.

Реалізувати подібну обробку можна, вдавшись до написання обробників подій в Delphi - саме так і зроблено в **FastReport**. Цей спосіб не є універсальним, тому що не дозволяє створювати нові звіти поза середовищем Delphi без переписування і перекомпіляції проекту. Саме тому в **FastReport** застосована вбудована мова - спрощений аналог Pascal. Скрипти, написані на цій мові, по суті справи, є обробниками подій, що викликаються перед промальовуванням об'єктів. Це дає можливість виконувати досить складну обробку інформації без написання коду в Delphi, і відповідно, без жорсткої прив'язки звіту до проекту.

Можливості вбудованої мови **FastReport** досить широкі. З скрипта доступні всі властивості і методи об'єктів звіту, а також змінні, поля таблиць БД. В скрипті створюються змінні і масиви, які доступні в усьому звіті. Про можливості вбудованої мови говорить той факт, що така досить складна задача, як друк сум групи в її заголовку (сама сума вважається в підвалі групи) засобами мови **FastReport** робиться просто.

2.2.1.1.5 Готовий звіт

Готовий звіт являє собою продукт діяльності ядра **FastReport** - то, що користувач бачить при натисканні кнопки **Попередній перегляд**. На відміну від багатьох генераторів звітів, які зберігають вміст сторінок звіту у вигляді метафайлу (тобто зображення у форматі **EMF**), в **FastReport** готовий звіт представлений набором об'єктів, що описують вміст кожної сторінки звіту. Це дозволяє модифікувати готовий звіт, завантажуючи

потрібну сторінку в дизайнер. Крім того, можна описувати реакцію на клацання миші на необхідному об'єкті в режимі попереднього перегляду звіту. Це дозволяє легко організувати роботу програми, при якій клацання на об'єкті звіту викликає генерацію нового звіту з більш детальними відомостями по вибраному об'єкту.

Вікно попереднього перегляду **FastReport** відрізняється від використовуваних в інших генераторах. Зокрема, в **FastReport** використовується схема перегляду документа і навігації, прийнята в **Microsoft Word**: у вікні можна бачити відразу кілька сторінок. Крім того, реалізована можливість пошуку тексту у всьому документі.

2.2.1.2 Документація по FastReport

Документація по **FastReport** знаходиться в каталозі \ **progr** \ **Fastreport**.

2.2.1.3 Приклад внесення змін до звіту

Наприклад, потрібно внести зміни до звіту № 1807 «Дні народження» - додати до даних посада працівника.

1. Заходимо в модуль « **Особисті картки** ». Вибираємо пункт меню **Реєстр / Друк** (клавіші **F9**). На екрані відкривається вікно " **Вибір вихідної форми** ".
2. У вікні " **Вибір вихідної форми** " вибираємо пункт меню **Реєстр / Меню...** (комбінація клавіш **Shift + F4**).
3. У вікні « **Налаштування меню звітів** » знаходимо відповідну форму звіту № 1807 « **Дні народження** »:

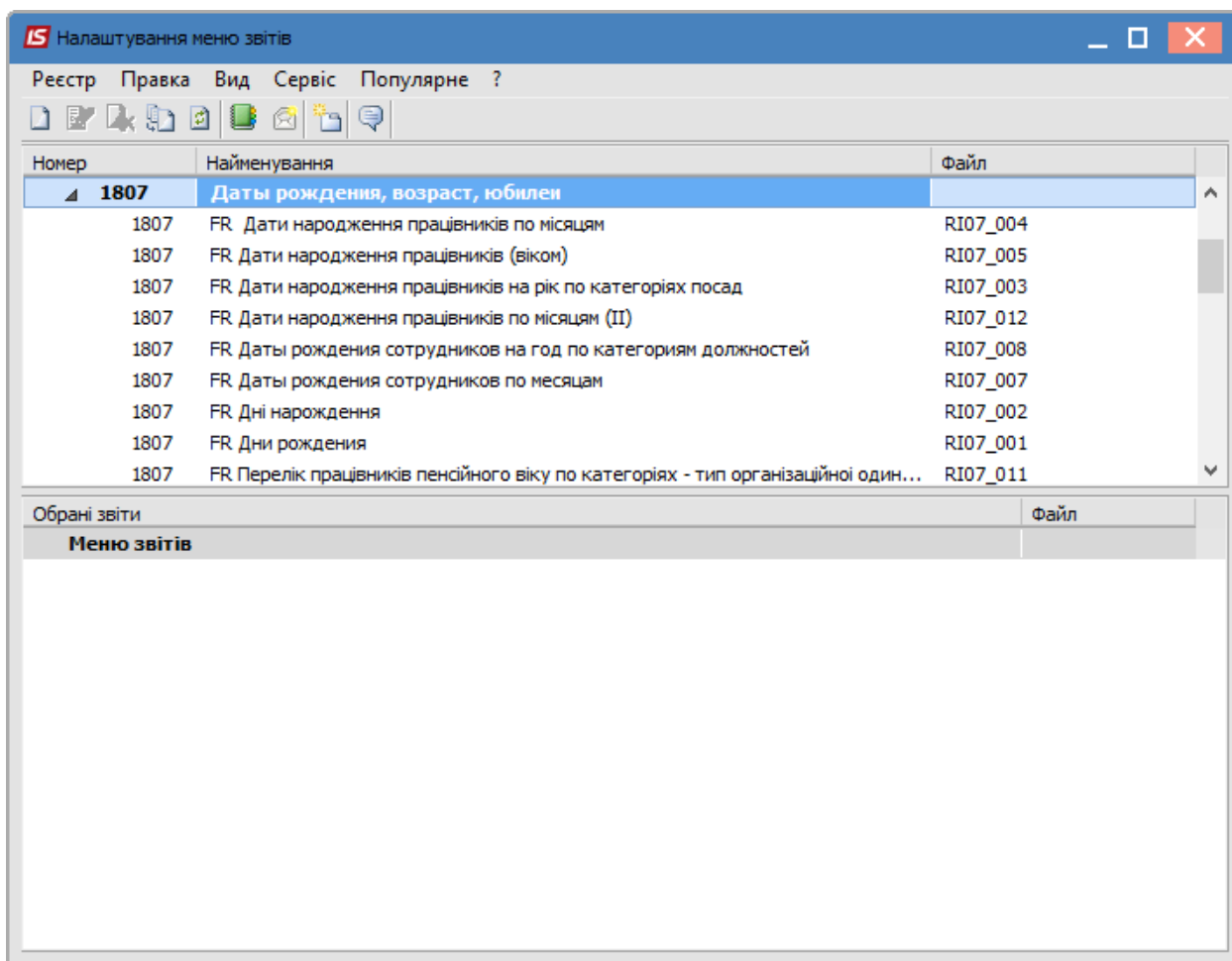


Рис. 16 – Налаштування меню звітів

4. Необхідно встановити курсор на звіт і вибрати пункт меню **Реєстр / Породити** (комбінація клавіша **Shift + Ins**).
5. У вікні « **Породження форми звіту** » внести найменування нового звіту, після чого натиснути кнопку « **ОК** ».
6. На екрані розгортається вікно редагування звіту:

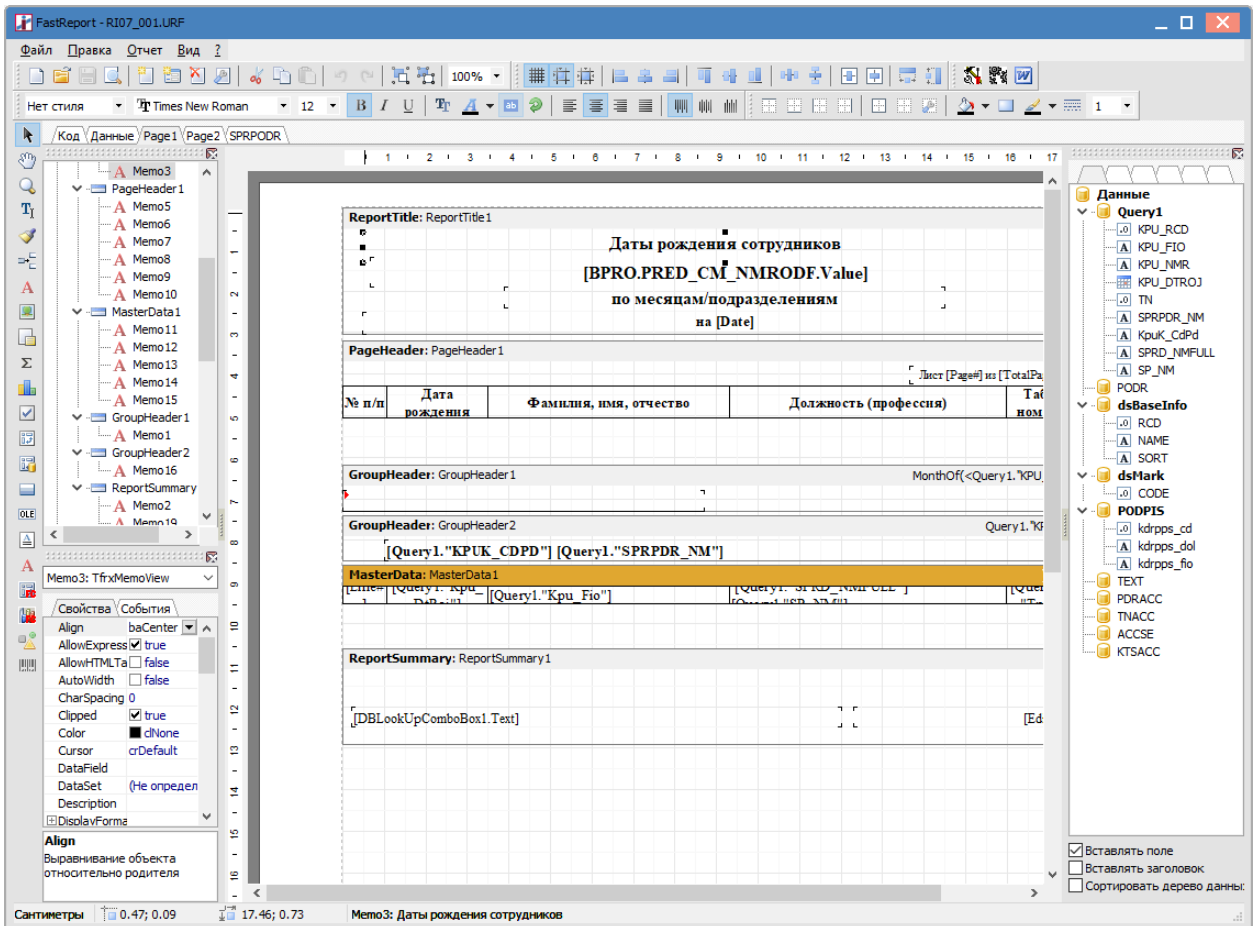


Рис. 17 – Закладка «Page1»

7. Звіт формату -FR складається з декількох сторінок:
 - сторінка з програмним кодом (на рис. закладка " Код ");
 - сторінка з деталями (мето-поля), які друкуються (на рис. закладка " Page1 ");
 - сторінка, яка містить вікно запуску звіту і SQL-запит, який заповнює деталі структури (на рис. закладка " Page2 ");
8. Переходимо на закладку " Page2" на запит Query1 (в нижній частині вікна) і розкриваємо його для редагування натисканням клавіші **Enter**. Відкривається редактор SQL-запитів.

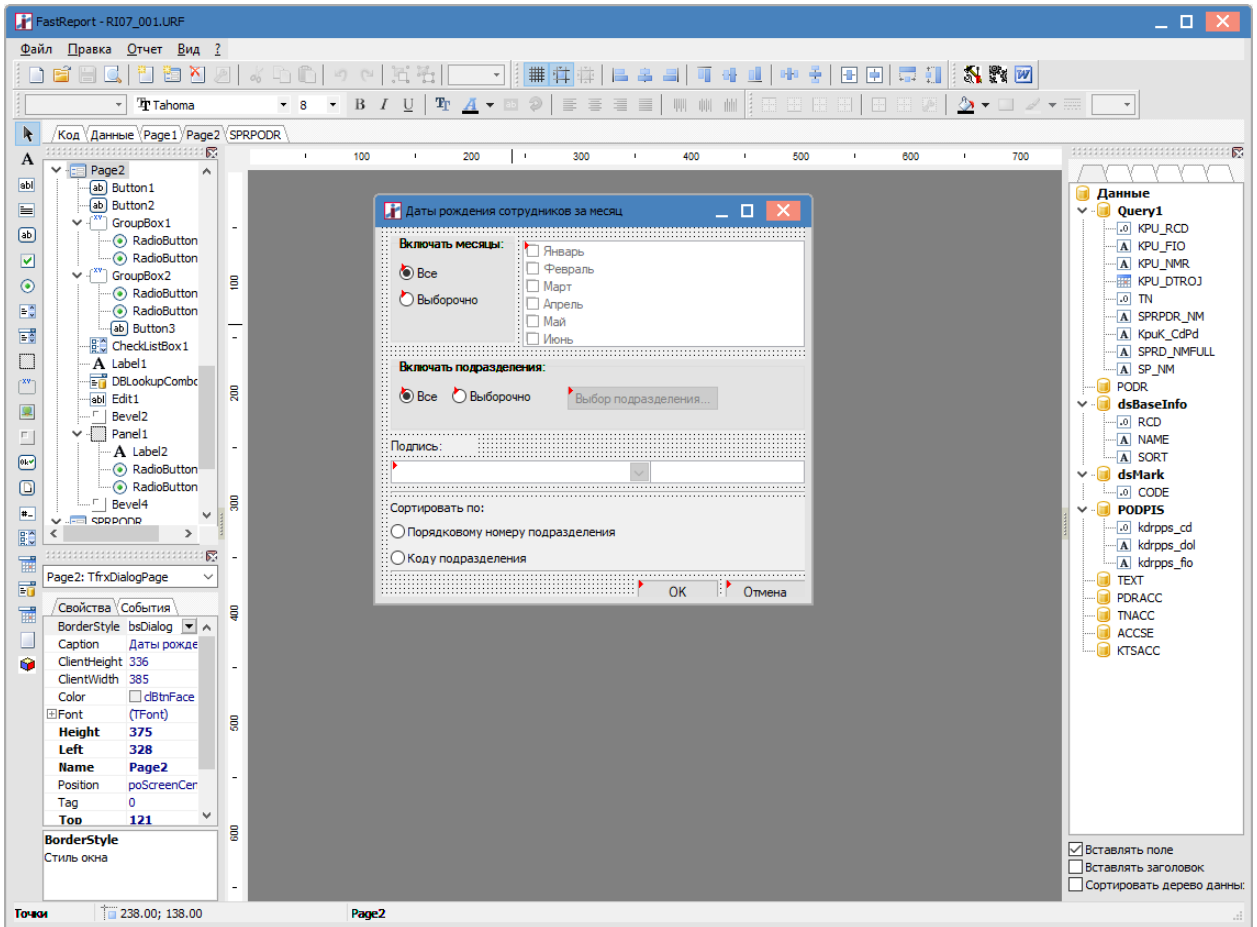


Рис. 18 – Закладка «Page2»

9. У SQL-запиті не вистачає поля з найменуванням посади, тому доповнюємо перелік полів (група **SELECT**) запиту, текстом «DOL.SPRD_NMFULL». Ім'я поля, яке необхідно додати визначається з бази даних (**Карти даних** (електронний опис бази даних) або опис бази в документі **Керівництво програміста**).
10. Зберігаємо зміни запиту, натискаючи на кнопку « **ОК** » (вона має форму зеленої галочки в верхньому лівому кутку вікна редагування SQL-запиту).
11. Переходимо на закладку " **Page1** " в центральну частину вікна. У рядку **MasterDate1** в поле [**Query1. "SPRPDR_NM"**] натискаємо клавішу **Enter** для його редагування. Відкривається вікно « **Редактор тексту** ».
12. У вікні « **Редактор тексту** » ставимо курсор на першу позицію. Натискаємо на кнопку « **Вставити вираз** ». Зі списку виразів вибираємо поле **SPRD_NMFULL** і двічі натискаємо на ньому мишею. Натискаємо « **ОК** ». Зберігаємо зміни у вікні « **Вставити вираз** », натискаючи на кнопку « **ОК** » (вона має форму зеленої галочки в верхньому лівому кутку вікна редагування).
13. При необхідності в рядку **PageHeader1** можна змінити назву стовпця, в якому додавали поле. Наприклад, змінюємо назву **Підрозділ** на **Підрозділ і посаду**.
14. Зберігаємо зміни в звіті « **Зберегти** » (комбінація клавіша **Ctrl + S** або натиснути на малюнок з дискетою).
15. Копіюємо звіт в меню звітів (**Реєстр / Копіювати F5**).
16. Звіт готовий до виконання.

2.2.1.4 Каталог USER_DATA

У звітах **FastReport**, написаних на мові Pascal, реалізована можливість використання директиви Uses, яка дозволяє підключення до програми звіту зовнішніх модулів, що представляють собою текстові файли з процедурами і функціями на мові Pascal. Це дозволяє оформляти часто використовувані процедури і функції в окремі бібліотеки.

Пошук підключаються файлів за замовчуванням буде проводитися в підкаталозі UNITS, каталогу USER_DATA серверної частини IS-pro.

Каталог USER_DATA з усім вмістом включається в резервну копію.

Приклад модуля myUnit. pas:

```
procedure s_Hello;  
begin  
  ShowMessage ('Hello from MyUnit.pas');  
end ;  
begin  
end.
```

Здійснюється виконання скриптів. Файли для користувача скриптів зберігаються в каталозі USER_DATA \ SQL, серверної частини IS-pro. Файли мають розширення SQL. Призначені для користувача скрипти виконуються при відновленні резервної копії і при оновленні серверних об'єктів в БД підприємства.

Список скриптів в файлі USER_DATA \ SQL \ USERSCRIPTS.LST серверної частини IS-pro.

Приклад USERSCRIPTS.LST:

```
Script1.Sql  
Script2.Sql  
Script3.Sql  
Script4.Sql
```

Порядок виконання скриптів визначається порядком проходження в файлі.

Приклад користувальницького скрипту для MSSQL Script1.Sql:

```
Create table TESTTABLE  
(ID int identity (1,1),  
Operation varchar (50))  
GO
```

Примітка.

Для СУБД MSSQL обов'язково в кінці скрипта написати "GO".

Для СУБД Oracle обов'язково в кінці скрипта написати "/".

2.3 Експорт зовнішньої звітності.

2.3.1 Загальні відомості

Для експорту звітів, побудованих за допомогою дизайнера необхідно скласти програми, які запускаються при формуванні фрагментів звіту.

Програми для звітів складаються на мові PascalScript. У програмах доступні функції, необхідні для формування файлу електронного документа. Повний перелік функцій наведено нижче.

Загальні правила побудови програми:

- На початку, при ініціалізації експорту, викликається функція MNSOpen, для ініціалізації файлу електронного документа.
- Потім формується заголовок файлу.
- У процесі видачі детальних рядків має бути сформовано вміст форми документа.
- В кінці експорту потрібно сформувати кінець фрагмента форми, кінець файлу і викликати функцію MNSClose, яка завершить сеанс роботи з файлом електронного документа.
- Програми формування файлу експорту прив'язуються до детальним фрагментами звіту. Можуть бути використані будь-які фрагменти (заголовки, детальні, підсумки).

2.3.2 Перелік доступних констант, функцій і процедур

2.3.2.1 Константи

Константи режимів відкриття файлу

MNSOMAPPEND: BYTE = \$ 01;

Дописати в кінець файлу, якщо існує.

MNSOMCREATE: BYTE = \$ 00;

Перезаписати файл, якщо існує.

MNSOMDIALOG: BYTE = \$ 10;

Видати діалог вибору файлу.

Константи роздільників груп

MNSDLMBLOCK: STRING = '###';

Кінець блоку.

MNSDLMFILE: STRING = '===';

Ознака кінця файлу.

MNSDLMPART: STRING = '@@@';

Кінець фрагмента.

2.3.2.2 Функції та процедури

DOCGETSTR (NAM: STRING): STRING;

DOCGETSTR (" ");

Отримати строкове значення поля з документа.

MNSCLOSE ;

Закрити поточний відкритий файл.

MNSGETFL_ID (INN: STRING): STRING;

MNSGETFL _ID (" ");

INN - ПІН фізичної особи

Дати ідентифікатор ФО.

MNSGETFORMDETAILNM (RZD, STR, GRF: STRING): STRING ;
MNSGETFORMDETAILNM (" ", " ", " ");

RZD - п'ятизначний номер (код) розділу форми

STR - п'ятизначний номер (код) рядку форми

GRF - двозначний номер графі форми

Дати найменування показника форми.

MNSGETMNS _ID (MNS: STRING): STRING ;

MNSGETMNS _ID (" ");

MNS - коду інспекції за класифікатором СОНО

Дати ідентифікатор МНС.

MNSGETUL_ID (INN: STRING; KPP: STRING): STRING;

MNSGETUL_ID (" ", " ");

INN - ПІН організації

KPP - код причини постановки на облік, якщо " - не виводиться

Дати ідентифікатор ЮЛ.

MNSOPEN: BOOLEAN;

MNSOPEN;

Відкриває останній відкритий файл, в разі помилки згенерує виняток.

MNSOPENFILE (FILENAME: STRING; MODE: INTEGER): BOOLEAN;

MNSOPENFILE (FILENAME, MODE);

FILENAME - ім'я файлу

MODE - режим відкриття

MNSOMCREATE - перезаписати файл, якщо існує

MNSOMAPPEND - дописати в кінець файлу, якщо існує

MNSOMDIALOG - видати діалог вибору файлу

MNSOMWIN1251 - кодування WINDOWS

Відкриває файл, в разі помилки згенерує виняток.

MNSPUTCOMMONINFO (DOCNUM: STRING; DOCDATE: TDATETIME; DOCPREDST: STRING);

MNSPUTCOMMONINFO (" ");

DOCNUM - номер документа

DOCDATE - дата формування документа

DOCPREDST - параметр "документ видається

Вивести в поточний відкритий файл загальні відомості інформаційної частини повідомлення.

MNSPUTDATE (APARAM: STRING; AVALUE: TDATETIME);

MNSPUTDATE (" ");

Вивести поєднання параметр-значення в поточний відкритий файл. В кінець рядка дописується пара символів \$ 0 d \$ 0 а. Рядок передається з WIN кодування в OEM кодування.

MNSPUTDELIMITER (ADELIMITER: STRING);

MNSPUTDELIMITER (" ");

Вивести роздільник в поточний відкритий файл.

MNSPUTDETAIL (RZD, STR, GRF, VAL: STRING);

MNSPutDetail (" ", " ", " ", " ");

RZD - п'ятизначний номер (код) розділу форми

STR - п'ятизначний номер (код) рядку форми

GRF - двозначний номер графі форми

VAL - значення показника

Вивести показник форми.

MNSPUTDOCID (A_ID: String; DocNum: String; ADate: TDateTime = 0);

MNSPutDocID (" ", " ");

A_ID - ідентифікатор ЮЛ, ФЛ або МНС

DocNum - номер документа

ADate - дата, за замовчуванням - поточна

Вивести ідентифікатор документа в поточний відкритий файл.

MNSPUTFILEID (A_ID: String; ADate: TDateTime = 0);

MNSPutFileID (" ");

A_ID - ідентифікатор ЮЛ, ФЛ або МНС '+ # 13 +

ADate - дата, за замовчуванням - поточна

Вивести ідентифікатор файлу в поточний відкритий файл.

MNSPUTNPFLINFO ;

Вивести в поточний відкритий файл відомості про платника податків - юридичну особу.

MNSPUTNUMBER (AParam: String; AValue: Extended; ALen: Integer; AMT: Integer; AFill0: boolean);

MNSPutNumber (" ",);

ALen - довжина всього числа з крапкою і знаком -

AMT - к-ть знаків після зайнятої

AFill 0 - доповнювати число до довжини нулями зліва

Вивести поєднання Параметр-Значення в поточний відкритий файл. В кінець рядка дописується пара символів \$ 0D \$ 0A. Рядок перекодується з Win-кодування в Oem.

MNSPUTOVERHEAD (KolDoc: Word ; MNSVer: String);

MNSPutOverhead ();

KolDoc - кількість документів

Вивести в поточний відкритий файл службову частину повідомлення.

MNSPUTPARAM (AParam: String; AValue: String);

MNSPutParam (" ");

Вивести поєднання Параметр-Значення в поточний відкритий файл. В кінець рядка дописується пара символів \$ 0D \$ 0A. Рядок перекодується з Win кодування в Oem.

MNSPUTSTRING (AParam: String; AValue: String; ALen: Integer);

MNSPutString (" ",);

ALen - довжина всього числа з крапкою і знаком -

Вивести поєднання Параметр-Значення в поточний відкритий файл значення - строкового типу. Значення форматується відповідно до довжини, якщо потрібно - буде

скорочуватися. В кінець рядка дописується пара символів \$ 0D \$ 0A. Рядок перекодується з Win кодування в Oem.

MNSPUTTEXT (AText: String);

MNSPutText (" ");

Вивести рядок в поточний відкритий файл. В кінець рядка дописується пара символів \$ 0D \$ 0A. Рядок перекодується з Win кодування в Oem.

SETKOLDOK (Kol: Integer);

SetKolDok ();

Встановити кількість документів.

SETKOLPOK (Kol: Integer);

SetKolPok ();

Встановити кількість показників форми.

STRINGTOCOMMA (AText: String): String;

StringToComma (" ");

Замінити прогалини на коми.

2.3.3 Приклад використання функцій експорту зовнішньої звітності в звіті Fast Report

Відкрити на редагування будь-який звіт. У обробнику події OnStartReport робимо відкриття файлу, наприклад:

MNSOpenFile ('C: \ MNStempoFile', mnsomCreate);

У обробнику події OnStopReport робимо закриття файлу:

MNSClose ();

У місцях формування даних для звіту використовуємо функції вивантаження, наприклад:

MNSPutText (Memo4.Text);

MNSPutDelimiter (mnsdlmPart);

Таким чином, результируючий файл може виглядати так:

```
var curs: real;
```

```
procedure Memo6OnBeforePrint (Sender: TfrxComponent);
```

```
begin
```

```
if <main. "VLCUR_KOT"> = <cross. "CURTYPE_RCD">
```

```
then curs: = <main. "VLCUR_CURS">
```

```
else curs: = 0;
```

```
end;
```

```
procedure OnStartReport (Sender: TfrxComponent);
```

```
begin
```

```
MNSOpenFile ('C: \ MNStempoFile', mnsomCreate);
```

```
MNSPutText (' Тема звіту ');
```

```
MNSPutDelimiter (mnsdlmPart);
```

```
end;
```

```
procedure OnStopReport (Sender: TfrxComponent);
begin
  MNSPutDelimiter (mnsdlmPart);
  MNSPutDelimiter (mnsdlmBlock);
  MNSPutDelimiter (mnsdlmFile);
  MNSPutText (' Фінал звіту ');
  MNSClose ();
end;

procedure Memo3OnAfterData (Sender: TfrxComponent);
begin
  MNSPutText (Memo3.Text);
end;

procedure Memo4OnAfterData (Sender: TfrxComponent);
begin
  MNSPutText (Memo4.Text);
  MNSPutDelimiter (mnsdlmPart);
end;

procedure Memo14OnAfterData (Sender: TfrxComponent);
begin
  MNSPutText (floatToStr (curs) + " + Memo14.Text);
end;

begin
end.
```

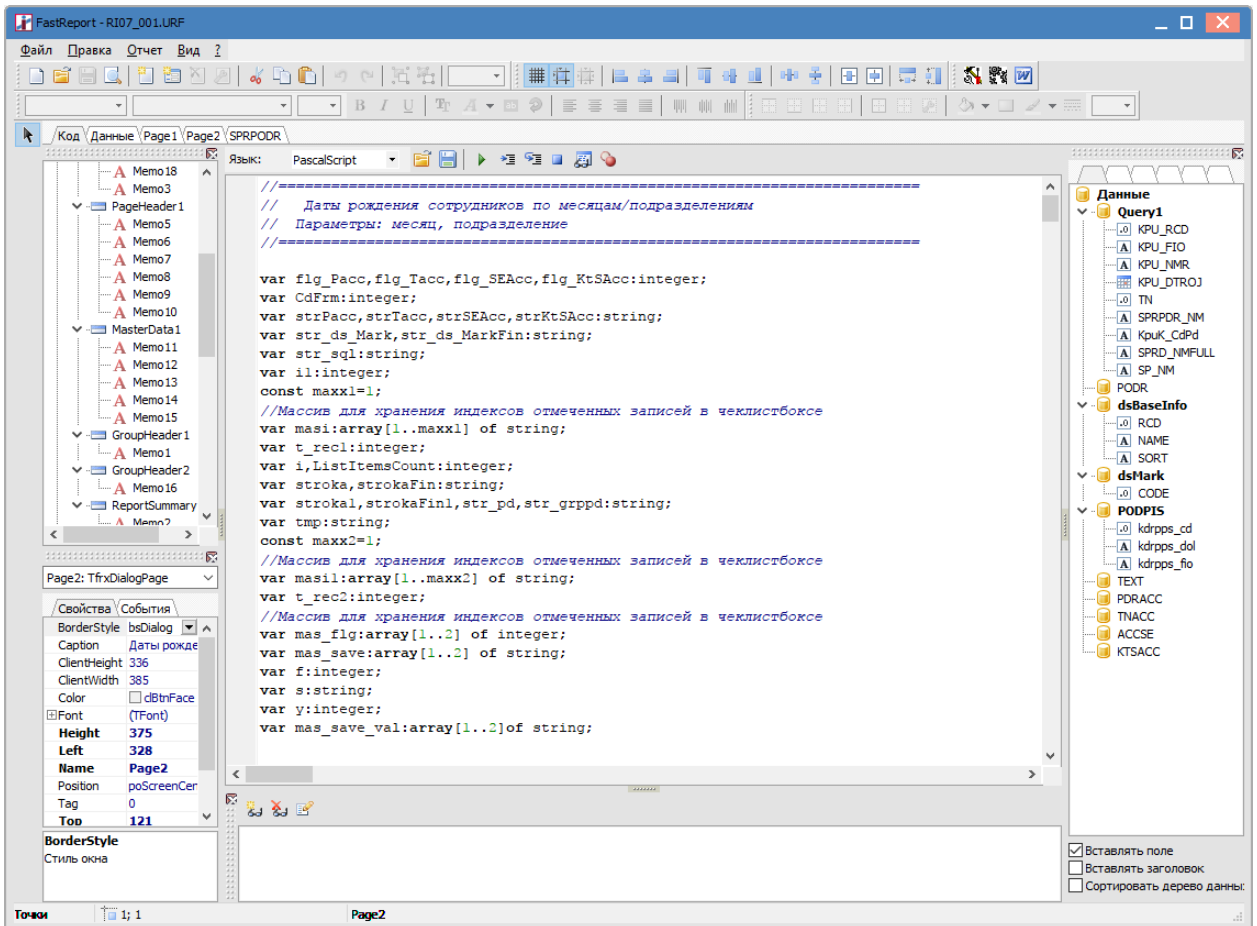


Рис. 19 – Дизайнер звітів і вікно редактора програми експорту

В результаті виконання звіту формується наступний файл - C:\MNStempoFile.
Лістинг файлу:

заголовок звіту

@@@
USD

Долар США

Множник курсу: 1

@@@
5.05 5.05000

5.05 5.05000

...

5.05 5.05000

EUR

ЄВРО

Множник курсу: 1

@@@

6.4321 6.43210

6.3933 6.39330

...

6.405925 6.40592

RUR

Російський рубль

Множник курсу:

@@@

0.18857 0.18857

0.18847 0.18847

...

0.1889 0.18890

@@@

###

===

фінал звіту

2.4 Імпорт файлів MS Office Word (*.doc) в FastReport

Для прискорення реалізації звітів використовується імпорт файлів **MS Office Word (*.doc)** в **FastReport**. У редакторі звітів на панелі інструментів **Мастеру** за допомогою кнопки **Імпорт з MS Office Word** відкривається вікно імпорту.

Імпорт з MS Office Word проводиться згідно налаштування:

1. **Тип - Повний, Тільки таблиці, В обрану таблицю, Тільки текст ;**
2. **Сторінками - Все або обрана** (в першому випадку відсутні сторінки в FR створюються автоматично. У другому - необхідну сторінку в FR можна створити прямо в вікні імпорту) ;
3. **Текст - по абзацах або за словами.**

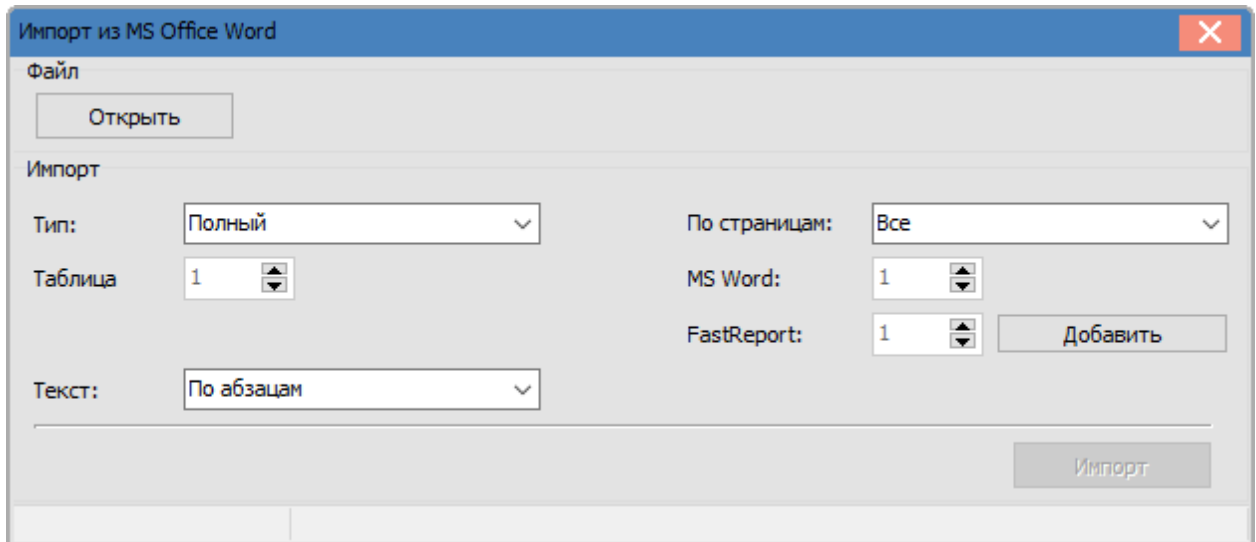


Рис. 20 – Імпорт з MS Office Word

У MS Word немає поняття «рядок» в звичному «досовському» трактуванні, а є тільки поняття «абзац», який є одним рядком, який переносять автоматично по деяким законам, наприклад, в залежності від ширини листа. При цьому в **MS Word** дозволяється форматування (зміна шрифту, стилю і т.д.) будь-якій частині тексту, аж до одного символу. У свою чергу в **FastReport** форматування допустимо тільки в межах **TfrxMemoView**. Тому реалізовано два типи імпорту тексту:

- **По абзацах** (абзац Word = TfrxMemoView) ;
- **П про словами** (слово Word = TfrxMemoView).

Перший тип працює швидко, але форматування тексту може не відповідати оригіналу в разі, якщо в межах абзацу застосовуються різні формати тексту.

Другий тип працює значно повільніше, але форматування відповідає аж до окремих слів.

Після імпорту може знадобитися незначна коректування звіту.

2.5 Розробка звітів MS Office (Excel, Word)

Система **IS-pro** підтримує створення, редагування і запуск шаблону **Excel**. Для правильної взаємодії з системою **IS-pro** при розробці шаблону слід дотримуватися певних правил:

- на зовнішній вигляд і вміст шаблону не потрібно встановлювати обмеження;
- VBA програми формування звіту повинні починати роботу всередині підпрограми ReportMain;
- підпрограма ReportMain служить «точкою входу» в програмний блок звіту. Система **IS-pro** завантажує шаблон, відшукує в його модулях підпрограму ReportMain, запускає її і чекає завершення підпрограми. Параметром підпрограми ReportMain передається менеджер COM об'єктів системи. За допомогою менеджера можливий доступ до COM об'єктів системи для отримання інформації, необхідної звіту. Об'єкт менеджера можна використовувати до закінчення виконання

підпрограми ReportMain, після її закінчення об'єкт буде зруйнований і всі звернення до нього приведуть до помилки;

- вся програма формування звіту повинна розташовуватися всередині підпрограми ReportMain.

Створення даних звітів починається з вибору при створенні форми звіту типу - Excel звіту або Word - звіту.

Реалізовано новий тип звітів **Excel 2007, 2010** для реалізації додаткового функціоналу, який надає **Microsoft Excel 2007 та 2010**. Ці звіти мають розширення **xltm** (системні звіти) і **xlum** (призначені для користувача звіти). Даний тип звітів працює тільки в середовищі **Microsoft Excel 2007** і вище.

У рядку **Шаблон** відображаються таблиці, склад яких залежить від підсистеми в якій створюється звіт.

У рядку **Найменування** - вводиться найменування створюваного звіту.

Після натискання клавіші **Ок** проводиться запуск програми (Excel або Word відповідно). Відкривається файл що є шаблоном звіту. Для формування звіту необхідно по пункту меню програми **Сервіс / Макрос / Редактор Visual Basic** запустити редактор формування програми звіту.

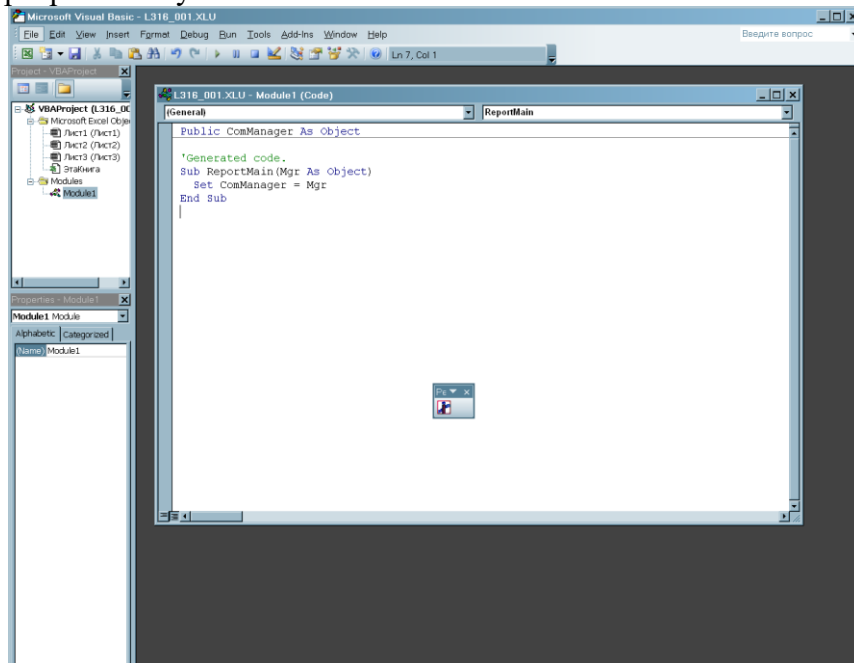


Рис. 21 – Редактор

У редакторі вже сформований кістяк програми. Далі за допомогою **Visual Basic** створюється програма побудови звіту. Доступ до даних здійснюється за допомогою SQL запитів, COM-об'єктів.

Для формування образу шаблону звіту його можна копіювати з існуючих звітів (по комбінації клавіш **Ctrl + C, Ctrl + V**).

2.5.1 Приклад програми Excel звіту

Приклад програми Excel звіту (VBA script)

```
'Звіт складається з даних двох таблиць
'Ptnggrp - список груп контрагентів
'Ptnrk - список контрагентів
'Зв'язок між таблицями
'Ptnrk.ptnggrp_rcd = ptnggrp.ptnggrp_rcd
'Поля таблиць
'ptnggrp
'Ptnggrp_rcd - ід записи групи
'Ptnggrp_cdk - код групи
'Ptnggrp_nmk - найменування групи
'
'ptnrk
'Ptnggrp_rcd - ід групи
'Ptn_cd - код контрагента
'Ptn_nm - найменування контрагента
'
'Порядок побудови звіту:
'Сканується список груп контрагентів
'Для кожної групи вибирається список відповідних контрагентів

'Примірник менеджера COM об'єктів
'Необхідний для запиту примірників COM об'єктів системи
Public ComManager As Object

'Generated code.
'
'Головна точка входу в програму побудови звіту
'Ця процедура буде викликана системою IS-pro
'Після відкриття шаблону звіту
Sub ReportMain (Mgr As Object)
On Error GoTo ErrorHandler

'Запам'ятати екземпляр об'єкта
'Переданий системою
Set ComManager = Mgr

' Запуск побудови звіту
MakeReport

Exit Sub

ErrorHandler:
MsgBox ("Error:" & Chr (13) & Err.Description)
End Sub

Private Sub MakeReport ()
' Групи контрагентами
```

```
Dim PtnGrpObj As Object
' Контрагенти
Dim PtnObj As Object
' Лист звіту
Dim RepSheet As Worksheet
'Початкова рядок листа звіту
Dim row As Integer

'В разі виникнення помилки буде видано повідомлення з текстом помилки
On Error GoTo ErrorHandler

'Лист книги, на якому буде побудований звіт
Set RepSheet = ActiveSheet

'Запросити у менеджера екземпляри двох об'єктів виконання SQL запитів
Set PtnGrpObj = ComManager.GetObjByName ("Sys", "ISysSqlQuery")
Set PtnObj = ComManager.GetObjByName ("Sys", "ISysSqlQuery")

' Запит для читання груп
PtnGrpObj.Text = "select ptngrpk_rcd as rcd, ptngrp_cdk as cd, ptngrp_nmk as nm from
ptngrpk"

' Запит для читання контрагентів групи
PtnObj.Text = "select ptn_cd as cd, ptn_nm as nm from ptnrk where ptngrpk_rcd =: grp_rcd"

'Відкрити запит для груп
PtnGrpObj.OpenObj

'Сканувати групи
If PtnGrpObj.First Then
row = 3
Do
'Висновок найменування групи
RepSheet.Range (" B " & CStr (row) & ": C " & CStr (row)). Merge
RepSheet.Range ("A" & CStr (row) & ": C" & CStr (row)). Font.Bold = True
RepSheet.Range ("A" & CStr (row) & ": C" & CStr (row)). BorderAround (xlContinuous)
RepSheet.Range ("A" & CStr (row)). Value = PtnGrpObj.GetFieldValue ("cd")
RepSheet.Range ("B" & CStr (row)). Value = PtnGrpObj.GetFieldValue ("nm")
row = row + 1
'Встановлення коду групи для запиту списку контрагентів
Call PtnObj.SetParamValue ("grp_rcd", PtnGrpObj.GetFieldValue ("rcd"))
'Відкрити запит списку контрагентів
PtnObj.OpenObj
' Сканувати список
If PtnObj.First Then
Do
' Висновок запису контрагента
RepSheet.Range ("A" & CStr (row) & ": B" & CStr (row)). Merge
RepSheet.Range ("A" & CStr (row) & ": C" & CStr (row)). BorderAround (xlContinuous)
RepSheet.Range ("A" & CStr (row)). Value = PtnObj.GetFieldValue ("cd")
```

```
RepSheet.Range ("C" & CStr (row)). Value = PtnObj.GetFieldValue ("nm")
row = row + 1
Loop While PtnObj.Next
End If
'Закрити об'єкт списку контрагентів
'Після закінчення читання списку для групи
PtnObj.CloseObj
Loop While PtnGrpObj.Next
End If
Exit Sub
ErrorHandler:
MsgBox ("Error:" & Chr (13) & Err.Description)
End Sub
```

2.5.2 Приклад програми Word звіту

Звітом є форма договору.

```
'Generated code.
Sub ReportMain (Mgr As Object)
Set ComManager = Mgr
Set EnvObj = ComManager.GetObjByName ("Sys", "ISysEnvironment")
'MsgBox (EnvObj.Dog_Ptn)
'Selection.Text = EnvObj.PTN_READ_INT (Selection.Text)
'Call EnvObj.PTN_READ_INT (EnvObj.Dog_Ptn)
'MsgBox (EnvObj.PtnR_NmSh)
'MsgBox (EnvObj.NUMTOSTR (EnvObj.U_ttttt))
```

```
With Selection.Find
.ClearFormatting
.Text = "& <*> &"
.Replacement.Text = ""
.Forward = True
.Wrap = wdFindContinue
.Format = False
.MatchCase = False
.MatchWholeWord = False
.MatchAllWordForms = False
.MatchSoundsLike = False
.MatchWildcards = True
If.Execute Then
Do
'MsgBox (Selection.Text)
Selection.Text = EnvObj.GETDOGTXTVALUE (Selection.Text)
Selection.EndOf
Loop While Selection.Find.Execute
End If
End With
End Sub
```

Перелік полів для формування договору:

Поля для функції ДОГОВОРУ

Номер договору - & DOGNMR &
Вхідний номер договору - & DOGVXDNMR &
Дата договору - & DOGDATE &
Об'єкт угоди - & DOGOBJ &
Коментар - & DOGCMT &
Центр відповідальності договору - & DOGCO &
Дата початку договору - & DOGDATEBEG &
Дата кінця договору - & DOGDATEEND &
Спосіб оплати - & DOGSPOPL &
Дата оплати - & DOGDATEOPL &
Котирування валюти - & DOGVALKOT &
Валюта договору - & DOGVAL &
Курс - & DOGVALKURS &

Реквізити свого боку

Найменування боку - & DOGSIDENM 1 &
Найменування контрагента - & DOGSIDEPTN 1 &
Адреса - & DOGSIDEADR 1 &
Контактна особа - & DOGSIDECONT 1 &
Телефон - & DOGSIDEPHN 1 &
Банк - & DOGSIDEBANK 1 &
МФО - & DOGSIDEMFO 1 &
Рахунок - & DOGSIDEACC 1 &
ПІН - & DOGSIDEINN 1 &

Реквізити іншого боку

Найменування боку - & DOGSIDENM 2 &
Найменування контрагента - & DOGSIDEPTN 2 &
Адреса - & DOGSIDEADR 2 &
Контактна особа - & DOGSIDECONT 2 &
Телефон - & DOGSIDEPHN 2 &
Банк - & DOGSIDEBANK 2 &
МФО - & DOGSIDEMFO 2 &
Рахунок - & DOGSIDEACC 2 &
ПІН - & DOGSIDEINN 2 &

Поля користувача - & ім'я &

Приклад:

Договір N & DOGNMR &
майнового найму (класична оренда)

& DOG DATE & р

м. Київ

& DOGSIDEPTN1 &, в подальшому Наймодавець, в особі & DOGSIDECONT1 &, що діє на підставі Статуту, з одного боку, і & DOGSIDEPTN2 &, в подальшому Наймач, в особі & DOGSIDECONT2 &, що діє на підставі & DOGSELSPR &, з іншого боку, керуючись взаємною угодою і чинним законодавством, уклали договір про наступне.

1. ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1. За цим договором Наймодавець зобов'язується надати Наймачеві майно у тимчасове користування, а Наймач зобов'язується оплатити його і повернути майно.

1.2. Наймодавець надає Наймачеві наступне майно: & DOGOBJ &.

1.2.1. Наймодавець передає Наймачеві наступні документи на майно: & DOGINSTR &.

Оформлення документів, необхідних для передачі майна в найм і для нормального користування майном оформлюються & DOGINSTR &.

1.3. Наймодавець надає Наймачеві майно у тимчасове користування строком на & DOGINSTR &.

1.4. Крім вищевказаного Наймодавець надає Наймачеві наступні послуги: & DOGINSTR &.

1.5. Наймач користується отриманим за цим договором имуществом самостійно в місці: на свій розсуд.

1.6. Охорона, зберігання, обслуговування і т.п. здійснюється & DOGINSTR & & DOGINSTR &, засобами & DOGINSTR &, за рахунок & DOGINSTR & & DOGINSTR &, в наступному порядку: & DOGINSTR &.

1.7. Власником переданого в оренду майна є & DOGINSTR &.

1.8. Майно передається Наймачеві в місці: & DOGINSTR & і його здача-приймання оформляється & DOGINSTR & (Додаток N 1).

(Розпискою, актом, протоколом)

1.9. Наймач після закінчення дії цього договору зобов'язується повернути майно, отримане за цим договором, наймодавця в місці: & DOGINSTR &, в стані: & DOGINSTR &, в терміни: & DOGINSTR &, а здача-приймання майна оформляється & DOGINSTR & (Додаток N 2).

(Розпискою, актом, протоколом)

1.10. Наймач зобов'язується користуватися найманим майном в порядку: & DOGINSTR &. (Вказати мету найму, порядок користування та режим і т.п.)

2. ВЗАЄМНІ РОЗРАХУНКИ СТОРІН

2.1. За користування майном Наймач зобов'язується виплатити наймодавця найману плату в розмірі & DOGSUM & рублів за & DOGINSTR &.

2.2. Крім вищевказаних платежів Наймач виплачує Наймодавець суму в розмірі & DOGINNUMB & рублів за & DOGINSTR & за надані Наймодавцем відповідно до п.1.4. послуги.

2.3. & DOGINSTR & за рахунок & DOGINSTR & здійснює страхування надається за цим договором майна на суму & DOGINNUMB & рублів в порядку & DOGINSTR &.

2.4. Наймач видає наймодавця в рахунок майбутніх розрахунків аванс в сумі & DOGINSTR & в порядку & DOGINSTR &.

2.5. За погодженням сторін в рахунок найманої плати (на весь період дії договору або на будь-який період) можуть передаватися послуги, права користування, майно і т.п.

2.6. Плата за користування майном вноситься незалежно від фактичного користування майном, якщо сторони додатково не визначають інше, крім випадків, коли Наймач не може користуватися майном з причин:

- хвороба користувача;
- форс-мажорні обставини;
- & DOGINSTR &.

2.7. Орендна плата за відповідний рік перераховується рівними частками щоквартально, не пізніше 20 числа першого місяця наступного кварталу.

2.8. Щорічно, після закінчення відповідного календарного року, сторони перевіряють правильність платежів по найманої плати і результати перевірки оформляють актом звірки розрахунків. За даними зазначеного акта в межах першого кварталу проводяться необхідні перерахунки.

2.9. Наймана плата підлягає перегляду у зв'язку зі зміною централізовано встановлюваних цін і тарифів, а також в інших випадках, передбачених чинним законодавством та додатковою угодою сторін.

3. Права та обов'язки сторін

3.1. Наймодавець зобов'язаний:

а) передати Наймачеві майно, що є предметом цього договору, в стані, що дозволяє його нормальну експлуатацію, відповідному його фактичній вартості, і цим договором;

б) крім вищевказаного, в тому ж порядку передати: & DOGINSTR &;

в) забезпечити передачу Наймачеві нормативно-правової, нормативно-технічної та наукової інформації;

г) проінструктувати про порядок користування і посприяти наймодавця по користуванню найманим майном;

д) оформити документи & DOGINSTR & в терміни: & DOGINSTR &;

е) у разі надання несправного (некомплектного) майна замінити (доукомплектувати) його або виправити не пізніше доби з моменту виявлення відповідних недоліків.

Зазначений строк не входить до терміну для обчислення неустойки і штрафних санкцій;

ж) попередити Наймача про приховані властивості (недоліки в т.ч.) найманого майна, які можуть завдати шкоди Наймач або його майну;

з) провести за власний рахунок і власними засобами і силами & DOGINSTR & ремонт (& DOGINSTR & обслуговування) майна в терміни: & DOGINSTR &.

У разі, якщо наймана майно виходить з користування Наймача, зазначений термін в термін дії цього договору не включається;

і) Наймодавець не може достроково розірвати цей договір в односторонньому порядку;

к) & DOGINSTR &.

3.2. Наймач зобов'язаний:

а) своєчасно проводити платежі, передбачені цим договором;

б) використовувати отримане за цим договором майно відповідно до його цільового призначення;

- в) під час приймання майна перевірити його справність і комплектність;
 - г) за власний рахунок утримувати майно в такому порядку:
& DOGINSTR & з урахуванням його природного зносу;
 - д) якщо наймана майно вибуває з ладу раніше повного амортизаційного терміну служби або & DOGINSTR &, то Наймач відшкодовує наймодавця недовнесених їм платежі за оренду майна. Залежно від фактичного стану майна та обставин вибуття з ладу, Наймодавець може не стягувати недовнесених найману плату;
 - е) & DOGINSTR &.
- 3.3. Наймодавець має право:
- а) отримати найману плату відповідно до цього договору;
 - б) отримати інформацію про стан найманого майна;
 - в) розірвати цей договір у разі, якщо Наймач порушує свої обов'язки в період користування найманим майном;
 - г) & DOGINSTR &.
- 3.4. Наймач має право:
- а) на здачу орендованого майна в піднайом, інше тимчасове користування іншим організаціям (в тому числі безоплатне), в разі:
 - невикористання майна з причин: & DOGINSTR &;
 - хвороби або & DOGINSTR & користувача майном;
 - & DOGINSTR &.
- Здача майна в тимчасове користування іншим організаціям не впливає на зміст фінансових відносин з Наймодавцем;
- б) на проведені їм віддільні поліпшення найманого майна;
 - в) на компенсацію вартості здійснених ним невіддільних поліпшень найманого майна;
 - г) на зниження найманої плати в разі зниження споживчих властивостей майна з причин, не залежних від Наймача, на суму, пропорційну зміненим властивостям;
 - д) на відшкодування витрат на поліпшення майна, проведені за згодою (в т.ч. без згоди) Наймодавця, в складі: вартості матеріалів, робочої сили і & DOGINSTR &, в т.ч. на відновлення майна після його природного зносу;
 - е) & DOGINSTR &.

4. ВІДПОВІДАЛЬНІСТЬ

- 4.1. У разі невиконання або неналежного виконання зобов'язань винна сторона зобов'язана відшкодувати другій стороні збитки у відповідності з чинним законодавством & DOGSELSPR &.
- 4.2. За прострочення сплати найманої плати Наймач сплачує & DOGINSTR & відсотків від несплаченої суми в необхідний термін.
- 4.3. За прострочення передачі майна за цим договором (у т.ч. надання неякісного, некомплектного майна) Наймодавець сплачує & DOGINSTR & відсотка від вартості непереданого в термін майна за кожен день прострочення.
- 4.4. Неустойки, передбачені цим договором, стягуються незалежно від відшкодування збитків.
- 4.5. У разі псування або загибелі найманого майна Наймач зобов'язаний: & DOGINSTR &.
- 4.6. Виплата неустойок і відшкодування збитків не звільняє сторони від виконання договірних зобов'язань в натурі.
- 4.7. Додаткові заходи відповідальності: & DOGINSTR & & DOGINSTR &.

5. ДІЯ ДОГОВОРУ

5.1. Цей договір вступає в силу з & DOGDATEBEG & і діє & DOGINSTR & (для майна - не більше 10 років; будови - не більше 5 років; обладнання - не більше 1 року)

5.2. Після закінчення зазначеного в п. 5.1. терміну Наймач має право на пролонгацію договору на новий термін, який сторони визначають додатково на умовах не погіршують становище сторін та за умови, що Наймач протягом терміну дії цього договору не порушував прийняті на себе зобов'язання.

5.3. Договір може бути змінений або розірваний в наступному порядку: Зацікавлена сторона повинна направити іншій стороні пропозицію про це. Сторона, яка отримала таку пропозицію, повинна дати відповідь не пізніше 10 днів після одержання пропозиції. У разі недосягнення згоди щодо зміни, розірвання договору, а також у разі неодержання відповіді спір вирішується за заявою зацікавленої сторони Державним арбітражем.

5.4. Цей договір може бути розірваний за ініціативою однієї сторони і поза порядком, зазначеного в п.5.3. в наступних випадках:

- погіршення стану найманого майна з причин, не залежних від Наймача;
- & DOGINSTR &.

5.5. У разі досягнення угоди про переукладання цього договору і до укладення нового договору діють норми чинного законодавства.

6. Додаткові умови

6.1. Спори сторін вирішуються в & DOGINSTR & & DOGINSTR &

6.2. & DOGINSTR & & DOGINSTR &

6.3. & U DOG &

7. Інші умови

7.1. На відносини сторін, не врегульовані цим договором, поширюються норми чинного цивільного законодавства.

7.2. Цей договір складено в 2 примірниках: по одному кожній стороні.

7.3. До цього договору в якості невід'ємних його частин додаються:

- Додаток N 1 - акт здачі-приймання майна;
- Додаток N 2 - акт здачі-приймання палива, сировини, матеріалів, незавершеного виробництва, готової продукції, фонди економічного стимулювання.

7.4. Реквізити сторін:

наймодавець

наймач

Адреса: & DOGSIDEADR1 &
Телефон: & DOGSIDE PHN2 &
р / с & DOGSIDEACC 1 &
& DOGSIDEBANK1 &
& DOGSIDEMFO1 &

Адреса: & DOGSIDEADR 2 &
Телефон: & DOGSIDE PHN2 &
р / с & DOGSIDEACC2 &
& DOGSIDEBANK2 &
& DOGSIDEMFO2 &

(Підпис)

М.П.

(Підпис)

М.П.

3 Робота з СОМ об'єктами системи IS-pro

Для організації взаємодії зовнішнього застосування з середовищем **IS-pro** в поставку системи включений ряд СОМ об'єктів, службовців для створення з'єднання з системою і отримання примірників об'єктів.

СОМ-об'єкти **IS-pro (Component Object Model** - компонентна об'єктна модель) призначені для організації зовнішнього доступу до прикладних функцій і даних системи.

Все СОМ-об'єкти системи відповідають вимогам автоматизації і можуть використовуватися з будь-яких мов, що підтримують роботу з інтерфейсами автоматизації.

СОМ об'єкти можуть містити набір предметних функцій, а так само, можуть повторювати функціональність набору даних і містити набір стандартних методів навігації і модифікації даних.

Для отримання примірника певного об'єкта зовнішню програму має виконати наступні кроки:

- Встановити з'єднання з системою, вказавши при цьому бажане підприємство. Після успішної установки з'єднання буде запущена робоча станція **IS-pro**, яка працює в режимі СОМ сервера.
- в результаті успішного з'єднання зовнішню програму отримує екземпляр об'єкта - менеджера СОМ об'єктом. Менеджер відповідає за створення екземплярів всіх інших СОМ об'єктів системи.
- зовнішню програму, за допомогою менеджера, створює необхідні СОМ об'єкти.

Звільнення об'єктів проводиться штатними засобами застосовуваного мови програмування.

Приклад програми на VBA, що виводить перші два рядки реєстру контрагентів.

```
' Менеджер СОМ об'єктів  
Dim ComManagerObj As Object
```

```
'З'єднання з IS-pro  
Dim ConnectObj As Object
```

```
'Запит для читання записів картотеки  
Dim QueryObj As Object
```

```
On Error GoTo ErrorHandler
```

```
'Створення об'єкту для з'єднання з системою  
Set ConnectObj = CreateObject ("ISStBoot.SysStartup.1")
```

```
'Запуск створення з'єднання  
'- станція в каталозі c: \ ispro  
'- користувач adm  
'- пароль master  
Set ComManagerObj = ConnectObj.Connect ("c: \ ispro", "adm", "master")
```

```
'З'єднання встановлено
'Вхід в підприємство №1
Call ConnectObj.SelectFirm (1)

'Всі етапи з'єднання завершені, можна працювати з системою

'Запросити об'єкт для виконання довільного SQL запиту
'Модуль об'єкта - Sys
'Найменування об'єкта - ISysSqlQuery
Set QueryObj = ComManagerObj.GetObjByName ("Sys", "ISysSqlQuery")

'Виконати запит читання картотеки контрагентів
QueryObj. Text = " select ptn _ nm from ptnrk "
QueryObj. OpenObj

'Прочитати перший запис
QueryObj. First
MsgBox (QueryObj.GetFieldValue ("ptn_nm"))

'Прочитати другий запис
QueryObj.Next
MsgBox (QueryObj.GetFieldValue ("ptn_nm"))

'Звільнити об'єкт
QueryObj.CloseObj
Set QueryObj = Nothing

ErrorHandler:
MsgBox ("Error:" & Chr (13) & Err.Description)
```

3.1 COM об'єкт для виконання SQL запитів

Об'єкт призначений для виконання довільних SQL запитів в схемі даних поточного підприємства і схемою загальносистемних даних. Облік розташування таблиці в загальній схемі даних або схемою підприємства, виконується автоматично.

Опис системних COM-об'єктів **IS-pro**

1. Назва об'єкту: ISysSqlQuery

Модуль: Sys

Бібліотека типів: sys _ c. dll

склад об'єкта

BSTR Text

Читання / запис тексту SQL запиту.

SetParamValue (BSTR ParamName, VARIANT Val)

Установка значення параметра запиту.

ParamName - найменування параметра, без символу ":"
Val - значення параметра

VARIANT GetFieldValue (BSTR FieldName)

Повертає значення поля запиту.
FieldName - найменування поля

VARIANT_BOOL OpenObj

відкрити набір

VARIANT_BOOL HRESULT CloseObj

Закрити набір

VARIANT_BOOL First

Стати на перший запис

VARIANT_BOOL Last

Стати на останній запис

VARIANT_BOOL Prior

Стати на попередній запис

VARIANT_BOOL Next

Стати на наступний запис

У разі необхідності використання параметрів в запиті, параметри необхідно іменувати відповідно до правил іменування полів і випереджати параметр в тексті запиту символом ":", наприклад where rcd =: new_rcd порівнюється значення поля "rcd" зі значенням параметра "new_rcd".

Приклад використання об'єкта

```
Private ConnectObj As Object  
Private ManagerObj As Object  
Private QryObj As Object
```

```
Private Sub DoWork ()  
On Error GoTo ErrorHandler  
Set ConnectObj = CreateObject ("ISStBoot.SysStartup.1")  
Set ManagerObj = ConnectObj.Connect ("D: \ bpw", "pva", "")
```

```
Call ConnectObj.SelectFirm (1)
```

```
Set QryObj = ManagerObj.GetObjByName ("Sys", "ISysSqlQuery")  
QryObj.Text = "select spr_nm from sspr where spr_rcd =: rcd"  
Call QryObj.SetParamValue ("rcd", 1)  
QryObj.OpenObj
```

```
QryObj.First  
While QryObj.Next
```

```
If Not IsNull (QryObj.GetFieldValue ("Spr_Nm")) Then  
MsgBox (QryObj.GetFieldValue ("Spr_Nm"))  
End If  
Wend  
QryObj.CloseObj
```

```
Exit Sub  
ErrorHandler:  
MsgBox ("Error:" & Chr (13) & Err.Description)  
End Sub
```

Опис прикладних COM-об'єктів **IS-pro**

3.2 Запуск ресурсних модулів з програми BASIC

В звітах Word і Excel для звернення до функцій середовища **IS-pro** слід використовувати такі виклики:

RES_ADD_MODULE (<Найменування модуля>)

Додати модуль в список запускаються

RES_DEL_MODULES

Очистити список запускаються модулів

RES_LOAD_MODULES

Завантажити все модулі з списку

RES_UNLOAD_MODULES

Завершити роботи з модулями з списку

Приклад звернення до ресурсів системи з звіту:

```
Sub ReportMain (Mgr As Object)
```

```
Set ComManager = Mgr
```

'Запросити інтерфейс зв'язку з середовищем **IS-pro**

```
Set Env = ComManager.GetObjByName ("Sys", "ISysEnvironment")
```

'Завантажити модуль

'Загальносистемні ресурси (SYS_R)

```
Env.RES_ADD_MODULE ("SYS_R")
```

```
Env.RES_LOAD_MODULES
```

'Вважати інформацію про контрагента

```
Env.PTN_READ_CD ("001")
```

'Завершити роботу з модулем

```
Env.RES_UNLOAD_MODULES
```

```
End Sub
```

Списки доступних модулів і їх вмісту надаються модулем доповнення до MS Office.

3.3 Проблеми при розробці звітів

Word і Excel звіти:

1. При створенні звіту Excel помилка

===== Помилка =====

Виклик методу: VBProject

Джерело: Microsoft Office Excel

опис:

Програмний доступ до проекту Visual Basic не є довіреною

Owner: SYSRXLT.CPP, File: SYSRXLT.CPP, Line: 654

Для усунення помилки необхідно в програмі Excel за пунктом меню **Сервіс / Макрос / Безпека** на закладці **Надійні видавці** зазначити параметр **Довіряти доступ до Visual Basic Project**.

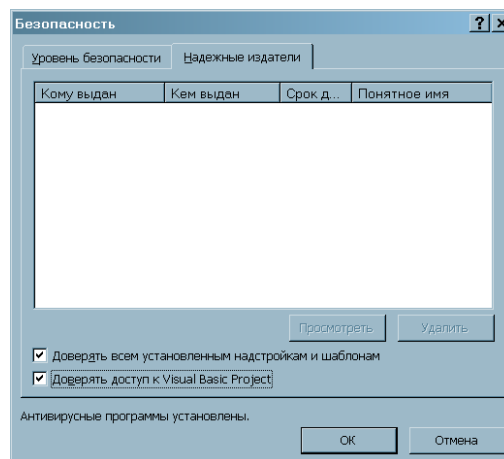


Рис. 22

2. При створенні звіту Word помилка

===== Помилка =====

Виклик методу: VBProject

Джерело: Microsoft

опис:

Відсутня довіра до програмованого доступу до проекту Visual Basic.

Owner: SYSRWRD.CPP, File: SYSRWRD.CPP, Line: 464

Для усунення помилки необхідно в програмі Word за пунктом меню **Сервіс / Макрос / Безпека** на закладці **Надійні видавці** зазначити параметр **Довіряти доступ до Visual Basic**

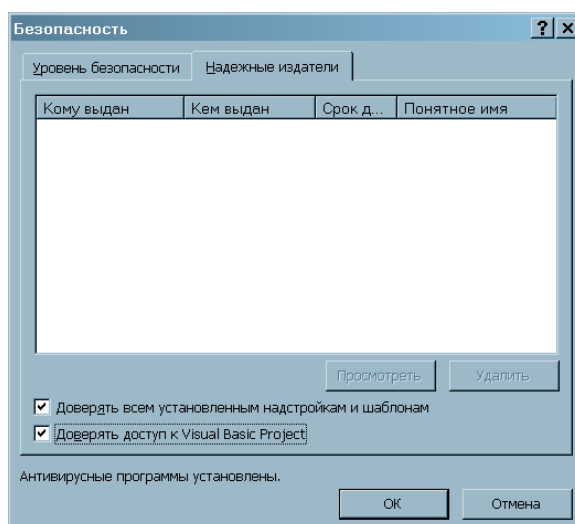


Рис. 23

4. Використання в програмі користувача розрахункових функцій головної книги

УВАГА! НЕ ПІДТРИМУЮТЬСЯ:

розрахункові функції ACC_..., що діють в рамках функції ACC_CALC;
отримання оборотів і залишків TURN_..., BALANCE_...;
перебору по реєстру проводок ENT_GET_FCALC/ ENT_GET_NEXT !

Ці функції замінені розрахунковими функціями OSB_..., що діють в рамках функції OSB_CALC.

4.1 Програма отримання оборотів і сальдо по рахунку в програмах користувача

4.1.1 Перед заданням параметрів необхідно очистити буфер даних для підготовки розрахунку.

CALL

OSB_DATA_CLEAR()

4.1.2 Наступним в програмі йде блок формування параметрів підготовки розрахунку. Оптимальним буде варіант, коли в цьому блоці декількома викликами функції **OSB_DATA** формується весь список даних для розрахунку.

Варіанти завдання параметрів розрахунку:

для отримання сальдо і оборотів по синтетичному рахунку (321):

CALL OSB_DATA ("321",0,"",0)

Для отримання сальдо и оборотів детального рахунку (321/2) з вказаною аналітичною картою (ID_An1):

CALL OSB_DATA("321/2",ID_An1,"",0)

для отримання оборотів по рахунку (321/2) в кореспонденції з рахунком (301):

CALL OSB_DATA("321/2",0,"301",0)

для отримання оборотів рахунка (321/2), враховуючи всі кор.рахунки:

CALL OSB_DATA("321/2",0,"*",0)

для отримання оборотів рахунка (321/2) в розрізі кор.рахунків та їх аналітики:

CALL OSB_DATA("321/2",0,"*",CONST_ULONG_MAX)

4.1.3 Далі в програмі йде блок формування параметрів підготовки розрахунку. Оптимальним буде варіант, коли в цьому блоці декількома викликами функції **OSB_DATA** формується весь список даних для розрахунку.

Варіанти завдання параметрів розрахунку:

об'єкт за період I квартал, всі валюти:

Obj1= OSB_CALC("01/01/17","31/03/17",")

об'єкт за період II квартал, всі валюти:

Obj2= OSB_CALC("01/04/17","30/06/17","")

об'єкт за період I квартал у зазначеній валюті:

Obj3= OSB_CALC("01/01/17","31/03/17","USD")

Система дозволяє підготувати одразу декілька об'єктів в розрізі різних періодів або валют.

Примітка – можливе використання розширеної функції **OSB_CALC2** для отримання розрахунків з додатковими параметрами.

4.1.4 При викликанні першої функції отримання сальдо або оборотів, розраховується сальдо по рахункам в рамках вказаного об'єкту, підготованого **OSB_CALC** (див. п.п. 4.1.3). Наступні виклики розрахункових функцій отримають дані з буферу розрахунку.

Приклади функцій отримання вхідного сальдо:

VX_Dt1 = **OSB_DT_BSLD**(Obj1,"321/1", 0,"") ' вх.сальдо дебету (далі Дт) 321/1
VX_Dt2 = **OSB_DT_BSLD**(Obj2,"321/2,Id_An1,"") ' вх.сальдо Дт 321/2 і аналіт.карт.
VX_Kt1 = **OSB_KT_BSLD**(Obj1,"321/1",0,"") ' вх.сальдо кредиту (далі Кт) 321/1
VX_Kt2 = **OSB_KT_BSLD**(Obj2,"321/2,Id_An1,"") ' вх.сальдо Кт 321/2 и аналіт.карт.

Приклади функцій отримання вихідного сальдо:

IX_Dt1 = **OSB_DT_ESLD**(Obj1, "321/1", 0, "") ' вих.сальдо по дт 321/1
IX_Dt2=**OSB_DT_ESLD**(Obj2,"321/2",Id_An1,"") ' вих.сальдо по дт 321/2 і аналіт. карт.

Приклади функцій отримання оборотів:

OB_Dt1=**OSB_DT_TURN**(Obj1, "321/1", 0,"",0,"") ' обороти по дт 321/1
OB_Kt1=**OSB_KT_TURN**(Obj1, "321/1", 0,"",0,"") ' обороти по кт 321/1
OB_Dt2=**OSB_DT_TURN**(Obj2,"321/2",0,"301",0,"") ' обороти по дт 321/2, в кор. 301
(см. правила заповнення **OSB_DATA** вище)
OB_Kt2=**OSB_KT_TURN**(Obj2, "321/2", 0,"301",ID_An12,"") ' обороти по дт 321/2, в кор. 301 з аналіт. ID_An12
(див. правила заповнення **OSB_DATA** вище)

4.2 Приклади використання розрахункових функцій в програмах користувача

4.2.1 Для отримання сальдо и оборотів по дебету зазначеного рахунку 321

DATN=DATE_PACK(1,12,2017)

DATK=DATE_PACK(31,12,2017)

CALL OSB_DATA_CLEAR()

CALL OSB_DATA("321",0,"",0)

Obj= OSB_CALC(DATN,DATK,"")

OB_Dt =OSB_DT_TURN(Obj,"321", 0,"", 0, "")

```
OB_Dt1=OSB_DT_TURN(Obj,"321/1",0,"",0,"")
OB_Dt2=OSB_DT_TURN(Obj,"321/2",0,"",0,"")
```

4.2.2 Для отримання сальдо по дебету и оборотів по дебету и кредиту рахунку 321/1 по аналітичній картці (ID- ІД аналіт. картки)

```
DATN=DATE_PACK(1,12,2017)
DATK=DATE_PACK(31,12,2017)
ID= ANLCRD_CREATE ("321/1","3507010","1111","","","","","","","","")
```

```
CALL OSB_DATA_CLEAR()
CALL OSB_DATA("321/1", ID, "", 0 )
```

```
Obj=OSB_CALC(DATN, DATK,"")
VX_Dt=OSB_DT_BSLD( Obj, "321/1", ID, "" )
OB_Dt=OSB_DT_TURN( Obj, "321/1", ID, "", 0, "" )
OB_Kt=OSB_KT_TURN( Obj, "321/1", ID, "", 0, "" )
IX_Dt=OSB_DT_ESLD( Obj, "321/1", ID, "" )
```

4.2.3 Для отримання оборотів по дебету 321/1 в кореспонденції с 701

```
DATN=DATE_PACK(1,12,2017)
DATK=DATE_PACK(31,12,2017)

CALL OSB_DATA_CLEAR()
CALL OSB_DATA("321/1",0,"701",0)
```

```
Obj=OSB_CALC(DATN, DATK,"")
```

```
OB_Dt =OSB_DT_TURN(Obj,"321/1", ID,"701", 0, "")
```

4.2.4 Для отримання оборотів по дебету та кредиту 321/1 в розрізі кореспондуючих рахунків:

```
DATN=DATE_PACK(1,12,2017)
DATK=DATE_PACK(31,12,2017)
CALL OSB_DATA_CLEAR()
CALL OSB_DATA("321/1",0,"*",0)
Obj=OSB_CALC(DATN, DATK,"")
```

```
Korr_Sch = OSB_COR_FIRST (Obj, "321/1", 0 )
WHILE Korr_Sch <> ""
    OB_Dt =OSB_DT_TURN(Obj,"321/1",0 , Korr_Sch,0,"")
    OB_Kt =OSB_KT_TURN(Obj,"321/1",0, Korr_Sch,0,"")
    Korr_Sch = OSB_COR_NEXT (Obj, "321/1",0)
WEND
```

4.2.5 Для отримання оборотів по дебету і кредиту 321/1 в розрізі всіх аналітичних карток:

```
DATN=DATE_PACK(1,12,2017)
DATK=DATE_PACK(31,12,2017)
```

```
CALL OSB_DATA_CLEAR()
CALL OSB_DATA("321/1",0,"",0)
Obj= OSB_CALC(DATN,DATK,"")
```

ID=

```
OSB_ANL_FIRST ( Obj, "321/1")
  WHILE ID<>0

    OB_DT=OSB_DT_TURN(Obj,"321/1",ID , "",0,"" )
    OB_KT=OSB_KT_TURN(Obj,"321/1",ID , "",0,"" )
    ID = OSB_ANL_NEXT (Obj,"321/1")
  WEND
```

4.2.6 Для отримання оборотів по дебету та кредиту 321/1 по аналітичним карткам зі вказаними реквізитами:

```
DATN=DATE_PACK(1,12,2017)
DATK=DATE_PACK(31,12,2017)
```

```
CALL OSB_DATA_CLEAR()
CALL OSB_DATA("321/1",0,"",0)
Obj= OSB_CALC(DATN,DATK,"")
```

```
ID = OSB_ANL_CDFST (Obj,0, "321/1","3507010", "", "", "", "", "", "", "", "", "", "")
```

```
WHILE ID <> 0
```

```
  ‘ ID — карточка последнего уровня картотеки
```

```
  IF ANLCRD_GET_LEV(ID) = ANLCRD_LCNLSCH("321/1") THEN
```

```
    OB_Dt=OSB_DT_TURN(obj, "321/1", ID,"" , 0, "")
```

```
    OB_Kt=OSB_KT_TURN(obj, "321/1", ID,"" , 0, "")
```

```
  END IF
```

```
  ID = OSB_ANL_LNEXT (obj,1,"321/1")
```

```
WEND
```

4.2.7 Для отримання перебору проводок по рахунку

```
DATN=DATE_PACK(1,12,2017)
DATK=DATE_PACK(31,12,2017)
CALL OSB_DATA_CLEAR()
```

```
CALL OSB_DATA("321/1",0,"",0)
Obj = OSB_CALC( DATN, DATK, "" )
CALL OSB_ENTRY_ON ( Obj )
```

```
ID_PRW = OSB_GET_FENTRY (Obj, 1)
```

```
WHILE ID_PRW <> 0
  SCH_DT=ENT_DT(ID_PRW)
  ANL_DT=ENT_DT_ANL(ID_PRW)
  SCH_KT=ENT_KT(ID_PRW)
  ANL_KT=ENT_KT_ANL(ID_PRW)
  SUM_PRW =ENT_SM(ID_PRW)
  ...
  ID_PRW =
OSB_GET_NENTRY ( Obj, 1)
WEND
```

4.2.8 Для отримання перебору проводок по дебету рахунку

```
DATN=DATE_PACK(1,12,2017)
DATK=DATE_PACK(31,12,2017)

CALL OSB_DATA_CLEAR()
CALL OSB_DATA("321/1",0,"",0)
Obj = OSB_CALC( DATN,DATK, "" )
CALL OSB_ENTRY_ON( Obj )
I=0
ID = OSB_ANL_FIRST( Obj,"321/1")
WHILE ID<>0
  I=I+1
  SUM_ANL =OSB_DT_TURN(Obj,"321/1", ID,"", 0, "")
  IF SUM_ANL <>0 THEN
    ID_PRW = OSB_GET_FENTRY ( Obj,I,"321/1", ID, "", 1 )
    DO WHILE ID_PRW <> 0
      SUM_PRW =ENT_SM(ID_PRW)
      ...
      ID_PRW = OSB_GET_NENTRY( Obj,I)
    LOOP
  END IF
  ID = OSB_ANL_NEXT(Obj,"321/1")
WEND
```

4.2.9 Для отримання оборотів по дебету рахунку 321/1 по аналітичній картці (ID1), і зазначених кор.рахунків та їх аналітики

```
DATN=DATE_PACK(1,12,2017)
DATK=DATE_PACK(31,12,2017)
ID1= ANLCRD_ID ("321/1","3507010","1111","","","","","","","")
ID2= ANLCRD_ID ("701","3507010","1111","","","","","","","")
ID3= ANLCRD_ID ("301","1","3507010","00000001","1111","","","","","")

CALL OSB_DATA_CLEAR()
CALL OSB_DATA( "321/1", ID1, "*", CONST_ULONG_MAX )
```

```
Obj=OSB_CALC(DATN, DATK, "")  
OB_Dt1=OSB_DT_TURN( Obj, "321/1", ID1, "701", ID2 , "" )  
OB_Dt2=OSB_DT_TURN( Obj, "321/1", ID1, "301", ID3 , "" )
```

4.3 Опис функцій головної книги

4.3.1 Розрахункові функції по роботі с даними рахунків

OSB_DATA_CLEAR() – функція очищує буфер параметрів розрахунку.

OSB_DATA("Сч", Сч_АНЛ, "СчКОРР", СчКОРР_АНЛ) – функція добавляє в набір параметрів розрахунку вказані данні.

Параметри :

"Сч" - рахунок

Сч_АНЛ - аналітика рахунку

"СчКОРР" - кореспондуючий рахунок

СчКОРР_АНЛ - аналітика кореспондуючого рахунку.

Примітка – параметри "СчКОРР" та СчКОРР_АНЛ втратили своє призначення і не враховуються в зміненому алгоритмі функцій отримання сальдо и оборотів.

OSB_CALC(Дата_Начала, Дата_Конца, "Вал") - функція готує об'єкт розрахунку відповідно до встановлених параметрів, періоду і валюти для фільтрації.

Параметри:

Дата_Начала - початок періоду розрахунків

Дата_Конца - кінець періоду розрахунку

"Валюта" - валюта фільтрації

Серед розрахункових функцій для роботи з даними рахунків добавлена функція «Подготовка Об'єкта расчета расширенная»:

OSB_CALC2 (Дата_Начала, Дата_Конца, "Валюта", Парам1, Парам2, Парам3, 0)

Функція готує об'єкт розрахунку у відповідності з встановленими параметрами, періодом і валютою для фільтрації. У випадку вдалої підготовки вертається номер об'єкту. На основі цього номеру формують свої результати інші функції в цьому розділі.

Параметри:

Дата_Начала — початок періоду розрахунку

Дата_Конца — кінець періоду розрахунку

"Валюта" — валюта для фільтрації

Парам1 — отримання оборотів та сальдо по верхньому рівню аналітики

Парам2 — 0 - фактичний облік / 1 - плановий облік

Парам3 — 1 - Розрахунок в базовій валюті (Запит курсу на дату кінця розрахунку) 0 – додаткові параметри не задіяні

Примітка – Установка параметрів проводиться функцією **OSB_DATA**.

OSB_DT_BSLD(Об'єкт, "Сч", Сч_АНЛ, "Вал")

Функція вертає значення вхідного залишку дебету по заданому рахунку та аналітиці, у зазначеній валюті.

Параметри:

Об'єкт - об'єкт, підготовлений **OSB_CALC**

"Сч" - рахунок

Сч_АНЛ - аналітика рахунку

"Вал" - валюта

Параметри Сч_АНЛ и *"Вал"* можуть не вказуватись. В параметрі *"Сч"* можна вказати консолідууючий або детальний рахунок.

Примітка – Для немультивалютних рахунків при встановленні параметра Валюта-"" або національна валюта результати виконання функцій **OSB_CALC**, **OSB_DT_BSLD**, **OSB_KT_BSLD**, **OSB_DT_TURN**, **OSB_KT_TURN** будуть однозначні. Для мультивалютних рахунків при встановленні параметра «Валюта» - іноземної валюти або національної валюти отримаємо сальдо та обороти рахунку в розрізі зазначеної валюти, при встановленні параметра «Валюта» - "" отримаємо сальдо та обороти в валюті обліку в цілому по рахунку.

OSB_KT_BSLD(*Об'єкт*, *"Сч"*, *Сч_АНЛ*, *"Вал"*) – функція вертає значення вхідного залишку кредиту по заданому рахунку і аналітиці, в зазначеній валюті.

OSB_DT_ESLD(*Об'єкт*, *"Сч"*, *Сч_АНЛ*, *"Вал"*) - функція вертає вихідний залишок дебету по зазначеному рахунку та аналітиці, в зазначеній валюті.

OSB_KT_ESLD(*Об'єкт*, *"Сч"*, *Сч_АНЛ*, *"Вал"*) - функція вертає вихідний залишок кредиту по зазначеному рахунку і аналітиці, в зазначеній валюті.

OSB_DT_TURN(*Об'єкт*, *"Сч"*, *Сч_АНЛ*, *"КорСч"*, *КорСч_АНЛ*, *"Вал"*)- функція вертає значення обороту по дебету рахунку в заданому співвідношенні.

Параметри :

Об'єкт - об'єкт, підготований **OSB_CALC**

"Сч" - рахунок

Сч_АНЛ - аналітика рахунку

"КорСч" - кор.рахунок

КорСч_АНЛ - аналітика кор.рахунку

"Вал" - валюта

Параметри Сч_АНЛ, *"КорСч"*, *КорСч_АНЛ*, *"Вал"* можуть не вказуватись або вказуються частково. В параметрі *"Сч"* можна зазначити консолідууючий або детальний рахунок, а в параметрі *"КорСч"* – тільки детальний рахунок.

OSB_KT_TURN(*Об'єкт*, *"Сч"*, *Сч_АНЛ*, *"КорСч"*, *КорСч_АНЛ*, *"Вал"*) - функція вертає значення обороту по кредиту .

OSB_ENTRY_ON(*Об'єкт*) - Функція встановлює режим збору даних о проводках.
Об'єкт — об'єкт, підготований **OSB_CALC**.

Режим необхідний для роботи функцій **OSB_GET_FENTRY** и **OSB_GET_NENTRY**.

OSB_GET_FENTRY(*Об'єкт*, *Номер_Цикла*) - Функція вертає ID першої проводки розрахунку по зазначеним параметрам.

Параметри:

Об'єкт - об'єкт, підготований **OSB_CALC**;

Номер_Цикла - номер циклу ітерації.

Примітка – Функція працює тільки у випадку, коли для об'єкту розрахунку був встановлений режим збору інформації о проводках функцією **OSB_ENTRY_ON**.

OSB_GET_NENTRY(*ІД Об'єкта, Номер Цикла*) - Функція вертає ID наступної проводки розрахунку за зазначеними параметрами.

Параметри:

Об'єкт - об'єкт, підготований **OSB_CALC**;

Номер_Цикла - номер циклу ітерації.

Примітка – Функція працює тільки в тому випадку, коли для об'єкту розрахунку був встановлений режим збору інформації о проводках функцією **OSB_ENTRY_ON**.

OSB_COR_FIRST_ALL(*Об'єкт, "Счет", Сч_АНЛ*); **OSB_COR_NEXT_ALL**(..);
OSB_COR_LFIRST_ALL(*Об'єкт, Номер_Цикла, "Сч", Сч_АНЛ*);
OSB_COR_LNEXT_ALL(...), котрі відрізняються від вже наявних в цьому розділі тим, що в параметр «Сч» може передаватися як детальні, так і консолідуєчі бухгалтерські субрахунки.

OSB_GET_FPENTRY(*Об'єкт, Номер_Цикла, "Счет", Сч_Анл, "Корр_Счет", Тип_Обор*),

Функція вертає ID першої проводки, відібраної останньою функцією **ACC_CALC**, що задовольняє зазначеним параметрам. Функція працює лише в тому випадку, коли для об'єкту розрахунку **OSB** був встановлений режим збору інформації о проводках функцією **OSB_ENTRY_ON**.

Об'єкт - об'єкт, підготований **OSB_CALC**,

Номер_Цикла - номер циклу ітерації,

"Счет" - рахунок фільтрації (обов'язковий параметр),

Сч_Анл - аналітика рахунку фільтрації,

«Корр_Счет" - кореспондуючий рахунок фільтрації,

Тип Оборота - тип оборотів (1 - по дебету, 2 - по кредиту, 0 – по дебету і кредиту).

4.3.2 Підгрупа функцій для отримання кореспондуючих рахунків:

OSB_COR_FIRST(*Об'єкт, "Счет", Сч_АНЛ*)

Функція вертає перший кор.рахунок за зазначеними параметрами в рамках об'єкту, підготованого **OSB_CALC**.

OSB_COR_LFIRST(*Об'єкт, Номер_Цикла, "Сч", Сч_АНЛ*)

Функція повертає перший кор.рахунок за зазначеними параметрами з урахуванням циклу сканування та в рамках об'єкту, підготованого **OSB_CALC**.

OSB_COR_NEXT(*Об'єкт, "Счет", Сч_АНЛ*) - функція вертає наступний кор.рахунок за вказаними параметрами в рамках об'єкту, підготованого **OSB_CALC**.

OSB_COR_LNEXT(*Об'єкт, Номер_Цикла, "Сч", Сч_АНЛ*)

Функція вертає наступний кор.рахунок за зазначеними параметрами з урахуванням циклу сканування та в рамках об'єкту, підготованого **OSB_CALC**.

4.3.3 Підгрупа функцій для перебору аналітичних карток рахунку:

OSB_ANL_FIRST(Об'єкт, "Сч")

Функція повертає першу аналітичну картку рахунку, по якій є обороти або сальдо в рамках об'єкту, підготованого OSB_CALC.

OSB_ANL_LFIRST(Об'єкт, Номер_Цикла, "Сч")

Функція повертає першу аналітичну картку рахунку, по якій є обороти або сальдо в рамках об'єкту, підготованого OSB_CALC, з урахуванням циклу сканування.

OSB_ANL_NEXT(Об'єкт, "Сч")

Функція повертає наступну аналітичну картку рахунку, по якій є обороти або сальдо в рамках об'єкту, підготованого OSB_CALC.

OSB_ANL_LNEXT(Об'єкт, Номер_цикла, "Сч")

Функція повертає наступну аналітичну картку рахунку, по якій є обороти або сальдо в рамках об'єкта, підготованого OSB_CALC, з урахуванням циклу сканування.

OSB_ANL_CDFST(Об'єкт, Номер_Цикла, "Сч", "Код1", "Код2", "Код3", "Код4", "Код5", "Код6", "Код7", "Код8", "Код9", "Код10")

Функція повертає ID першої аналітичної картки на рахунку, по якій є обороти або сальдо в рамках об'єкту, підготованого OSB_CALC, відповідній переданим кодам. Код аналітики на кожному рівні може бути заповнений або не заповнений (""). Наступне значення можна отримати за допомогою функції OSB_ANL_LNEXT.

OSB_ANL_LVLFST(Об'єкт, Номер_Цикла, "Сч", НомУровня)

Функція повертає ID першої аналітичної картки на рахунку, по якій є обороти або сальдо в рамках об'єкту, підготованого OSB_CALC на вказаному рівні, з вкладеним циклом. Наступне значення можна отримати за допомогою функції OSB_ANL_LNEXT.

4.3.4 Функції доступу к параметрам проводки

ENT_DATE (ID_проводки)

Функція повертає дату проводки по заданому ID.

ENT_DT (ID_проводки)

Функція повертає номер рахунку дебету проводки по заданому ID.

ENT_DT_ANL (ID_проводки)

Функція повертає ID аналітичної карточки по рахунку дебету проводки із заданим ID.

ENT_KT (ID_проводки)

Функція повертає номер рахунку кредиту проводки по заданому ID.

ENT_KT_ANL (ID_проводки)

Функція повертає ID аналітичної картки по рахунку кредиту проводки із заданим ID.

ENT_NOTE (ID_проводки)

Функція повертає рядок коментаря до проводки із заданим ID.

ENT_OPR_DOC (ID_проводки)

Функція повертає код документа господарської операції, до якої прив'язана проводка із заданим ID.

ENT_OPR_DOCID (ID_проводки).

Функція повертає ID документа господарської операції, до якої прив'язана проводка із заданим ID.

ENT_OPR_DOCNMV (ID_проводки)

Функція отримання зовнішнього номера документа господарської операції, до якої прив'язана проводка.

ENT_OPR_DOCDAT (ID_проводки)

Функція повертає дату документа господарської операції, до якої прив'язана проводка із заданим ID.

ENT_OPR_DOCNMR (ID_проводки)

Функція повертає номер документа господарської операції, до якої прив'язана проводка із заданим ID.

ENT_OPR_ID (ID_проводки)

Функція повертає ID господарської операції, до якої прив'язана проводка із заданим ID.

ENT_OPR_NM (ID_проводки)

Функція повертає найменування господарської операції, до якої прив'язана проводка із заданим ID.

ENT_OPR_TO (ID_проводки)

Функція повертає ID типової операції (шаблону проводок), на підставі якої створена господарська операція проводки.

ENT_POST_CD (ID_проводки)

Функція повертає ідентифікатор користувача проводки із заданим ID.

ENT_PRZ_HND (ID_проводки)

Функція отримує ознаку ручної проводки по заданому ID. Функція повертає 1, якщо проводка корегувалася користувачем, 0 – в іншому випадку.

ENT_SM (ID_проводки)

Функція повертає проводки по заданому ID.

ENT_VAL (ID_проводки)

Функція повертає валюту проводки по заданому ID.

ENT_VALCUR (ID_проводки)

Функція повертає курс валюти проводки по заданому ID.

ENT_USER (ID_проводки)

Функція повертає ім'я користувача, який провів операцію з проводками в Головний журнал по ID проводки.