

ности. Один из возможных форматов представления такой классификации - MuSCoW. Эта аббревиатура расшифровывается так: Must have - необходимые функции; Should have - желательные функции; Could have - возможные функции; Won't have - отсутствующие функции.

6. Выводы

В работе выделены основные проблемы, возникающие на стадии подготовки макетов в редакционно-

издательской деятельности. Рассмотрены существующие системы управления допечатной подготовкой многостраничных изданий, определены их достоинства и недостатки.

Для типового проектирования подобной системы для конкретного предприятия проведен анализ деятельности предприятия, указаны необходимые требования для проектируемого макета системы управления совместной работой над издательскими проектами и установлен перечень планируемых функций системы, проведена их классификация по степени важности.

Литература

1. Раттан К. Кроссмедийные системы в полиграфии и издательском деле. Выбор стратегии [Текст] / Кевин Раттан. - М.: МГУП, 2007г. - 198с.
2. Фрэнк Романо, Ф. Современные технологии в издательско-поли-графической отрасли [Текст] / пер. с англ. М. Бредис, В. Волбленко, Н. Друзьева; под ред. Б.А. Кузьмина. - М.: Принт-Медиа центр, 2006. - 456 с.
3. Гехман Ч. Рабочий поток. Практическое руководство [Текст] / Чак Гехман. - М.: МГУП, 2004 г. - 256 стр.
4. Грекул В.И. Проектирование информационных систем: курс лекций: учеб пособ. для вузов [Текст] / В. И. Грекул, Г. Н. Денищенко, Н. Л. Коровкина. - М.: Интернет-Ун-т Информтехнологий, 2005. - 304 с.

Розглянуто необхідність пошуку інформації у базах даних, та як це виконується у класичному випадку реляційних баз даних. Наведено гідності й недоліки класичного підходу. Запропоновано алгоритм пошуку інформації на основі семантичної інформації

Ключові слова: система пошуку інформації, база даних, запит

Рассмотрена необходимость поиска информации в базах данных, и то как это осуществляется в случае классической реляционной базы данных. Приведены достоинства и недостатки классического подхода. Предложен алгоритм поиска информации на основе семантических данных

Ключевые слова: система поиска информации, база данных, запрос

The necessity of the information retrieval from the databases and how it is implemented in case of relational databases is reviewed in the article. The value and limitations of the classical approach are given. The algorithm of the information retrieval that is based on the semantic data is proposed

Keywords: information retrieval, database, query

УДК 004.652

ПОШУК ІНФОРМАЦІЇ У СЕМАНТИЧНІЙ БАЗІ ДАННИХ ДОКУМЕНТІВ З РІЗНОМАНІТНИМ РЕКВІЗИТНИМ ЗМІСТОМ

С.С. Забара

Доктор технічних наук, професор, завідувач кафедри
Контактний тел.: 096-990-89-38
E-mail: symanyshyn_lesja@mail.ru

І.В. Ізварін

Аспірант*

*Кафедра інформаційних технологій та програмування

Інститут комп'ютерних технологій Відкритого міжнародного університету розвитку людини «Україна»

вул. Львівська, 23, м. Київ, 03115

Головна мета будь-якої бази даних, та СУБД в рамках якої така база даних існує, це зберігання інформації, та надання її за вимогами користувача [1,4]. Вимоги

користувача щодо надання потрібної інформації дуже важко характеризувати, та формалізувати. Розглянемо такий гіпотетичний приклад вимог користувача:

«Знайти всі документи, що мають інформацію про банк «Банк Кіпру», який а) знаходиться у місті Київ, та б) операції з банком проводилися у 2010 році. Вся знайдена інформація повинна також мати перелік товарів та послуг які були надані згідно операцій з банком, та назви й адреси компаній, які ці послуги надали.»

Очевидно, що наданий опис не може бути напряму передано до СУБД, щоб виконати пошук потрібної інформації. Цей опис потрібно обробити та збудувати відповідний SQL запит.

Побудова SQL запиту потребує знання структури бази даних, та значення всіх полів усіх таблиць в базі даних, тобто відповідності термінів у сформульованому вище гіпотетичному прикладі назвам полів таблиць БД.

В будь-якому випадку, щоб виконати пошук потрібної інформації необхідно трансформувати опис у відповідну мову цільової системи. Якщо цільова система - СУБД, то опис потрібно перекласти у речення SQL. У випадку запиту до текстового сховища, як, наприклад, через пошуковий сервіс Google, виконується аналіз запиту, та побудова семантичних навантажених критеріїв для подальшого пошуку у текстових даних за допомогою відповідних алгоритмів [5,6].

Такий класичний підхід до пошуку інформації має свої гідності, та недоліки.

Гідності класичного підходу:

1. Класичний підхід базується на теорії реляційної алгебри [1].
2. Класичний підхід використовує стандартизовану мову запитів SQL.
3. Запити SQL оптимізуються цільовою СУБД, що дозволяє дуже швидко виконувати пошук потрібної інформації.

Недоліки класичного підходу:

1. Зміна структури бази даних, наприклад у випадку додавання до бази даних нового типу документу для зберігання, може привести до повної переробки запитів пошуку інформації.
2. Нові запити кінцевого користувача, які не мають відповідної реалізації, повинні бути проаналізовані, та реалізовані у термінах мови SQL, для чого потребуються відповідні фахівці. Тобто, кінцевий користувач, який не має знань структури бази даних, та не вмів використовувати мову SQL не може отримати відповідь на свій запит.
3. Аналіз вимог та формування SQL запиту потребує значного часу, тому що потрібно проаналізувати до декількох сотень реквізитів (полів таблиць БД) [2], та побудувати складний запит і налагодити його.

Запропонована у роботах [3,4] семантична база даних документів з різноманітним реквізитним змістом дозволяє уникнути тих недоліків які було наведено вище. Наявність двох рівнів семантичної інформації дає змогу уникнути детального аналізу опису проблеми та пошуку відповідних полів потрібної інформації, а також захистить від змін реалізованого програмного забезпечення у випадку додавання нових типів документів.

Наявність семантичних даних також дає можливість спростити формування запиту для пошуку інформації. Розглянемо як семантична інформація допоможе кінцевому користувачу отримати відповідь на приклад наведений на початку:

1. Оскільки сховище інформації зберігає дані для всіх типів документів, у результаті ми отримаємо документи різних типів у яких зустрічається дана назва банку.

2. Якщо користувач має додаткові вимоги (що не наведені у прикладі) щодо потрібних йому типів документів він може перелічити їх на початку формування запиту.

3. Перший рівень семантичної інформації – глобальні блоки – дозволить нам скоротити кількість реквізитів серед яких ми повинні вибрати потрібні.

4. Вибрані реквізити дозволять нам ще звузити результати пошуку, якщо ми задамо конкретні умови для кожного з реквізитів.

5. Після формування запиту, та його виконання користувач матиме змогу вибрати потрібні йому для відображення додаткові реквізити, які не приймали участі у пошуку інформації.

Тепер ми маємо змогу створити алгоритм пошуку інформації у семантичній базі даних. Алгоритм пошуку складається з трьох частин: формування умов пошуку, вибір підмножини документів, підготовка та формування звіту.

Розглянемо першу частину алгоритму, блок-схема якого зображена на рис. 1.

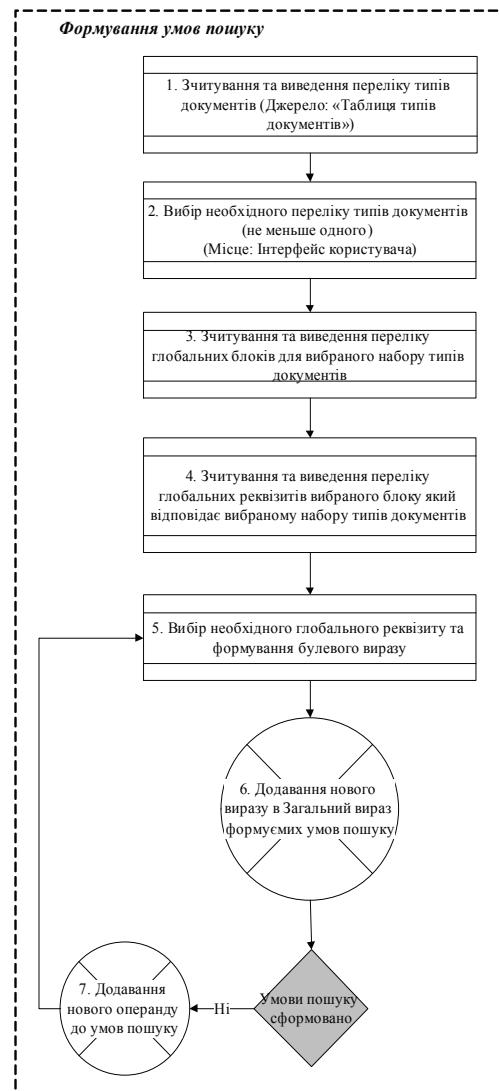


Рис. 1. Блок-схема алгоритму формування умов пошуку

На першому кроці виконується зчитування інформації з семантичного ядра щодо всіх типів документів, які існують у базі даних. Зчитана інформація виводиться на інтерфейс і користувачу надається можливість вибрати потрібні типи документів для пошуку (крок 2). Початково вважається, що пошук буде виконуватися по всіх типах документів.

Третій крок алгоритму виконує зчитування семантичної інформації першого рівня, тобто перелік всіх глобальних блоків для вибраних типів документів, та відображення їх на інтерфейсі. Користувачу надається можливість вибрати потрібні йому глобальні блоки для подальшого уточнення умов пошуку.

На наступному кроці виконується зчитування глобальних реквізитів відповідно до вибраних глобальних блоків та виведення їх на інтерфейс користувача.

На п'ятому кроці користувач вибирає потрібний йому глобальний реквізит, та формує булевий вираз умов пошуку. У випадку нашого гіпотетичного запиту умова для реквізиту «Банк» повинна мати вигляд: «Банк = 'Банк Кіпру'», а для реквізиту «Дата операції»: «Дата операції = 2010».

Сформований булевий вираз додається до повної умови пошуку за допомогою операторів «ТА» («AND») та «АБО» («OR»).

Перша частина алгоритму завершується формуванням повних умов пошуку інформації.

На рис. 2 зображено блок-схему другої частини алгоритму пошуку.

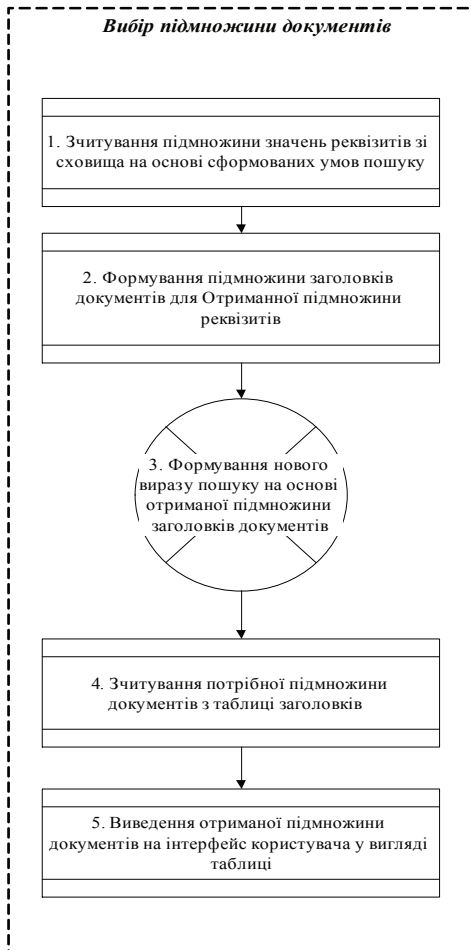


Рис. 2. Блок-схема вибору підмножини документів

На основі переліку значень реквізитів в умові пошуку алгоритм витягує інформацію зі сховища. Ця інформація відповідає всім елементарним умовам пошуку («Банк = 'Банк Кіпру'») і має також референції на заголовки документів до яких ця інформація відноситься.

Другий крок алгоритму вибору підмножини документів отримує відповідну підмножину заголовків документів і на наступному кроці формує новий вираз для пошуку, тому що потрібно вибрати всю інформацію, яка має відношення до даного документу.

Вся вибрана інформація виводиться на інтерфейс користувача для подальшого вибору потрібної додаткової інформації. У нашому прикладі, це та інформація яка не була присутня в умовах пошуку, а саме: перелік товарів та послуг, назва та адреса компанії.

Третя частина алгоритму пошуку відповідає за формування результуючого звіту, і надає можливість вибрати потрібні інформаційні реквізити вибраних документів. Блок-схема алгоритму зображена на рис. 3.

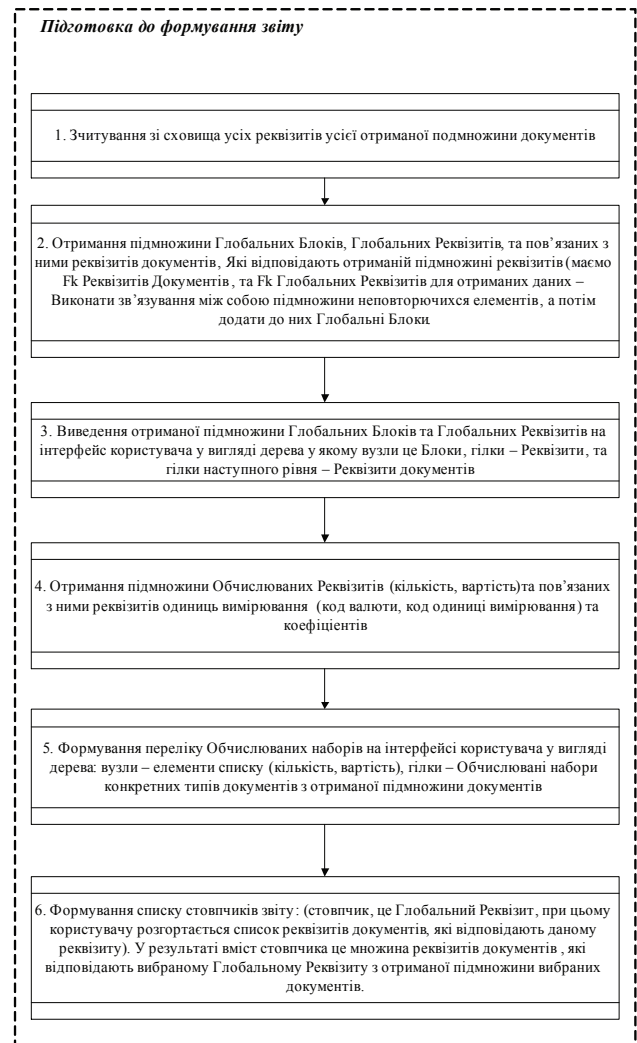


Рис. 3. Блок-схема підготовки формування звіту

Важливо відмітити наступні відмінності у запропонованому підході до пошуку інформації за допомогою семантичної складової бази даних від класичного підходу:

1. Якщо до бази даних буде додано новий тип документів, то він відразу з'явиться на інтерфейсі користувача і буде приймати участь у формуванні умов пошуку (згідно до першого кроку формування умов пошуку, дивіться рис. 1).

2. Усі нові глобальні блоки та глобальні реквізити нових документів також з'являться на інтерфейсі кінцевого користувача і будуть приймати участь у формуванні умов пошуку. Ніяких додаткових програмних засобів для цього не потрібно.

3. Завдяки присутності семантичних даних першого рівня, тобто глобальних блоків, зменшується кількість реквізитів, які будуть приймати участь у пошуку, що спрощує відбір потрібних для пошуку реквізитів, та зменшує час потрібний для формування умов пошуку.

4. Формування умов пошуку відбувається у термінах які відомі й зрозумілі користувачу (давайте знову звернемося до прикладу наведеного на початку статті), а не у термінах SQL мови.

5. Зменшення кількості реквізитів, які приймають участь у формуванні умов пошуку, та які вибирає користувач, приводить до спрощення взаємодії кінцевого користувача з програмним засобом для пошуку.

Реалізація наведеного алгоритму складається з двох частин: шару SQL процедур на боці СУБД, та клієнтського прикладного програмного засобу для інтерфейсу кінцевого користувача.

SQL процедури мають простий вигляд. У якості приклада на рис. 4 наведено код процедури, яка реалізує крок 2 алгоритму формування умов пошуку.

Для того, щоб було можливо виконувати пошук у базах даних, які було створено на основі СУБД що не підтримують універсальний тип даних SQL_VARIANT (Microsoft SQL Server), або ANYDATA (Oracle) було створено проширення представлень (VIEW), щоб створити єдиний програмний інтерфейс для процедур, що виконують зчитування даних зі сховища.

```

1 /*
2
=====
=====
3 Проект: Поисковая система
4 Автор: Изварин
5 Описание : Возвращает перечень блоков по внутреннему имени системы
6 Версия : 1.00
7 История : 17 июня 2009 -- создан
8
9 Входные параметры: SystemId - Id системы
10
=====
=====
11 */
12
13 IF EXISTS (SELECT name FROM sysobjects WHERE name = 'GetBlocks' AND
TYPE = 'P')
14 DROP PROCEDURE [dbo].[GetBlocks]
15
16 GO
17
18 CREATE PROCEDURE GetBlocks
19 @SystemId INT
20 AS
21
22 SELECT
23 f_id,
24 f_name
25 FROM
26 [dbo].[Blocks]
27 WHERE
28 f_id IN (SELECT f_fkBlockId FROM [dbo].[DocumentBlocks] WHERE
f_fkDocId = @SystemId)
29 ORDER BY
30 f_name

```

Рис. 4. Процедура зчитування переліку глобальних блоків для даного типу документу

Замірювана швидкість формування умов пошуку не перевищувала 10-15 хвилин. Цей час складається з часу для зчитування даних з бази даних, формування інтерфейсу кінцевого користувача, та інтерактивної роботи користувача по аналізу потрібних глобальних блоків та реквізитів, та формування додаткових умов для кожного реквізиту. Час пошуку потрібної інформації не перевищував 5-ти хвилин у СУБД Microsoft SQL Server 2005 Enterprise Edition, яка працює під операційною системою Windows Server 2008 Enterprise Edition. Час підготовки та формування звіту також не перевищував 5-ти хвилин значна частина з яких також відноситься до роботи з інтерфейсом [2].

Література

1. Дейт К. Дж. Введение в системы баз данных = Introduction to Database Systems. — 8-е изд. — М.: «Вильямс», 2006. — 1328 с. — ISBN 0-321-19784-4.

2. Забара С. С., Изварин И. В. Семантические базы данных документов с различным реквизитным содержанием // Компьютерно-интегрированные технологии: освіта, наука, виробництво: Науковий журнал / Під редакцією доктора технічних наук В. Д. Рудь. – Луцьк: Видавництво Луцького національного технічного університету, 2011, № 5. – С. 87-92.
3. Изварин И.В. Жизненный цикл информационного потока // Интеллектуальные системы принятия решений и проблемы вычислительного интеллекта: Материалы международной научной конференции. Том 1. – Херсон: ЧНТУБ 2011ю – с. 76-78.
4. Gerald J. Kowalski, Mark T. Maybury. Information Storage and Retrieval Systems. Theory and Implementation. Second Edition. Kluwer Academic Publishers 2002.
5. Greengrass Ed. Information Retrieval: A Survey, 2000.
6. Stefano Cen, Marco Brambilla. Search Computing. Trends and Developments. Springer-Verlag Berlin Heidelberg 2011.

На основі фреймової моделі подання знань запропоновано принципи розробки баз знань та експертних систем універсальною мовою програмування Python. Проаналізовано об'єктно-орієнтовані можливості та засоби інтроспекції Python. Розроблено демонстраційну базу знань та подано приклади запитів до неї

Ключові слова: база знань, онтологія, подання знань, експертна система

На основе фреймовой модели представления знаний предложены принципы разработки баз знаний и экспертных систем на универсальном языке программирования Python. Проанализированы объектно-ориентированные возможности и средства интроспекции Python. Разработана демонстрационная база знаний и даны примеры запросов к ней

Ключевые слова: база знаний, онтология, представление знаний, экспертная система

On the basis of frames for knowledge representation have been proposed principles for the development of knowledge bases and expert systems on general-purpose programming language Python. Object-oriented and introspection capabilities of Python have been analyzed. The demo of knowledge base and the examples of querying to it have been developed

Keywords: knowledge base, ontology, knowledge representation, expert system

УДК 004.822

ЗАСТОСУВАННЯ МОВИ ПРОГРАМУВАННЯ PYTHON ДЛЯ ПОБУДОВИ БАЗ ЗНАНЬ ТА ЕКСПЕРТНИХ СИСТЕМ

В.Б. Копей

Кандидат технічних наук, доцент
Кафедра технології нафтогазового
машинобудування

Івано-Франківський національний технічний
університет нафти і газу

вул. Карпатська, 15, м. Івано-Франківськ, 76019

Контактний тел.: (03422) 4-30-24

E-mail: vkopey@gmail.com

Л.М. Семанишин

Аспірант

Івано-Франківська філія Відкритого міжнародного
університету розвитку людини «Україна»

вул. Набережна, 42а, м. Івано-Франківськ, 76010

Контактний тел.: 096-990-89-38

E-mail: symanyshyn_lesja@mail.ru

1. Вступ

Відомо, що розвиток науки ґрунтується на отриманні об'єктивних і системно-організованих знань, які допомагають людям раціонально організувати свою діяльність і вирішувати різні проблеми, котрі виникають в її процесі. Сучасні комп'ютерні технології зумовили розвиток інженерії знань - області науки про штучний інтелект, яка вивчає методи і засоби отримання, представлення, структурування

і використання знань. Інженерія знань пов'язана з розробкою баз знань і експертних систем [1,2].

База знань - це сукупність фактів і правил логічного висновку (виведення) в обраній предметній області. **Правила логічного висновку** - це правила перетворення вихідної системи фактів (суджень) в нову систему фактів (висновків). Наприклад, за такими правилами з вихідних фактів "всі люди смертні" і "Сократ - людина" можна зробити висновок "Сократ смертний". В даному випадку правилом висновку є