3. Тихонов В. И. Марковские процессы / В. И. Тихонов, М. А. Миронов. – М., 1977. – 488 с.

4. Марков А. А. Исследование замечательного случая зависимых испытаний / А. А. Марков // Изв. Петерб. АН (6). – 2007. – Т. 1. – № 3 – С. 61–80.

5. Бусленко Н. П. Метод статистических испытаний (Монте-Карло) и его реализация на цифровых вычислительных машинах / Н. П. Бусленко, Ю. А. Трейдер. – М. : ГИФМЛ, 1961. – 230 с.

6. Швецов В. И. Математическое моделирование транспортных потоков / В. И. Швецов // Автоматика и телемеханика. – 2003. – № 11. – С. 3–46.

7. Осипов Л. А. Проектирование систем массового обслуживания / Осипов Л. А. – М. : Адвансед Солюшнз, 2011. – 111 с.

8. Козлов И. Т. Пропускная способность транспортных систем / Козлов И. Т. – М. : Транспорт, 1985. – 214 с.

9. Самарский А. А. Математическое моделирование. Идеи. Методы. Примеры / А. А. Самарский, А. П. Михайлов. – М. : ФИЗМАТЛИТ, 2005. – 320 с.

10. Рыжаков А. Н. Современные проблемы математического моделирования в исследовании операций / А. Н. Рыжаков, О. А. Щербина // Динамические системы. – 2006. – № 21. – С. 115–129.

11. Пасічник А. М. Математичне моделювання і оптимізація функціонування митного пункту пропуску / А. М. Пасічник, А. В. Сохацький, О. В. Брюховецький // Вісник АМСУ. – 2007. – № 3. – С. 80–89.

✧ ✧ ✧ ✧ ✧ ✧ ✧

УДК 351.72:356.13(477)

Б. І. Мороз, доктор технічних наук,
декан факультету інформаційних та транспортних систем і технологій Академії митної служби України
Л. В. Кабак, кандидат технічних наук, доцент кафедри інформаційних систем та технологій Академії митної служби України
О. Н. Молотков, кандидат технічних наук, доцент кафедри інформаційних систем та технологій Академії митної служби України
О. В. Трофімов, кандидат технічних наук, доцент кафедри транспортних систем та технологій Академії митної служби України

МЕТОДИ ТА МОДЕЛЬ СИСТЕМИ ПІДВИЩЕННЯ ПРОДУКТИВНОСТІ СЕРВЕРУ ORACLE

Запропоновано методику підвищення продуктивності серверу ORACLE, розроблену на основі методів і моделей експертних систем, яка дозволяє за допомогою пакета STATSPACK проводити аналіз продуктивності роботи серверу ORACLE. Розроблена методика допомагає оцінити параметри конфігурації серверу і давати поради адміністратору БД щодо налаштування параметрів конфігурації серверу з метою підвищення продуктивності.

Предложена методика повышения производительности сервера ORACLE, разработанная на основе методов и моделей экспертных систем, которая позволяет с помощью пакета STATSPACK проводить анализ производительности работы сервера ORACLE. Разработанная методика помогает оценить параметры конфигурации сервера и давать рекомендации администратору БД по настройке параметров конфигурации сервера с целью повышения производительности.

Method for increasing the performance of ORACLE server was proposed. The method is based on expert system models and allows analyzing the productivity with STATSPACK package. This method helps to tune the parameters of server configuration and may give the recommendation server administrator how to improve performance.

Ключові слова. Сервер бази даних, експертна система, продуктивність роботи серверу.

Вступ. На сучасному етапі економічного розвитку спостерігається бурхливий розвиток інформаційних технологій і масове впровадження інформаційних систем в усі галузі виробництва, банківської справи, структури управління, державні установи. З підвищенням продуктивності інформаційних систем різко зростають вимоги до їх надійності, захищеності та продуктивності. Це особливо стосується інформаційних систем, за допомогою яких здійснюється управління на рівні економіки держави. Зокрема, інформатизація процесів справляння податків, митних зборів і формування дохідної частини бюджету держави – нагальна потреба розвитку економіки України.

Як відомо, значну частину сучасних інформаційних систем становлять системи збирання та обробки поточної інформації (OLTP), які працюють на базі технології клієнт-сервер. Тому продуктивність таких систем визначається продуктивністю комп'ютерної мережі та серверу бази даних [1].

В Єдиній автоматизованій інформаційній системі Департаменту митної справи Міністерства доходів і зборів як сервер центральної бази даних використовується ORACLE 10g. Тому від продуктивності його роботи залежить продуктивність роботи інформаційної системи Департаменту митної справи Міністерства доходів і зборів.

У статі розглянуто методи та модель налагодження параметрів конфігурації ORACLE для підвищення продуктивності роботи інформаційних систем.

Постановка завдання. Нинішнього часу роботу налаштування параметрів серверу не повністю автоматизована і проводиться за допомогою STATSPACK utility, version 10.2.0, яка складається з командного файла, написаного мовою PL/SQL [2]. Ця програма періодично запускається адміністратором серверу бази даних і заповнює статистичні таблиці продуктивності, після чого адміністратор аналізує таблиці за допомогою утиліти Enterprise Manager та змінює ті чи інші параметри конфігурації.

Недостатня автоматизація породжує проблеми, характерні для процедур налаштування таких систем. По-перше, параметрів конфігурації дуже багато й адміністратору не зручно щоразу звертатись до довідника. По-друге, між багатьма параметрами існує сильна функціональна залежність, і для підвищення ефективності роботи недостатньо варіювати одними параметрами, не змінюючи значення інших.

Процес налаштування серверів баз даних включає в себе декілька основних етапів, які потрібно пройти, щоб виявити проблему та її причину і внести зміни до параметрів налаштування для локалізації та усунення проблеми зниження продуктивності. На рис. 1 відображено спрощену модель налаштування продуктивності БД.

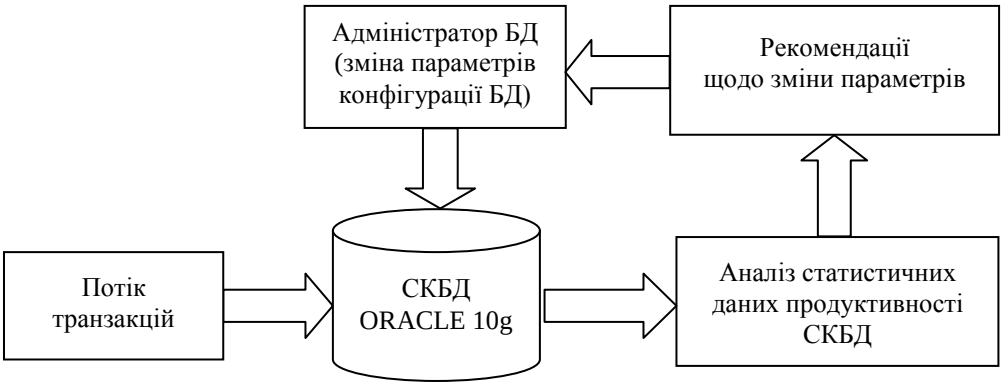


Рис. 1. Модель налаштування параметрів продуктивності БД ORACLE

Постійна боротьба за підвищення продуктивності інформаційних систем і технологій стимулює розвиток ефективних методів і моделей налаштування серверів баз даних. Максимально можлива продуктивність СКБД ORACLE може забезпечуватись лише за умови постійного моніторингу та виявлення і ліквідації "вузьких місць". Технології на базі моделей експертних систем вважаються найбільш гнучкими та пристосованими для розв'язання таких задач.

Отже, для дослідження створено методику, що ґрунтується на базі експертної системи, для налаштування серверу ORACLE. Для розв'язання цієї проблеми потрібно провести аналіз моделей подання знань і розробити таку модель, яка забезпечить ефективне функціонування експертної системи для налаштування параметрів серверу ORACLE.

Результати дослідження. Узагальнена структура експертної системи (ЕС) передбачає такий процес функціонування: користувач, який бажає отримати необхідну інформацію з логічними висновками через інтерфейс користувача, відправляє запит до ЕС; розв'язувач (програма, що формує рішення) за допомогою бази знань генерує і видає користувачу певну рекомендацію, пояснюючи хід своїх міркувань із використанням підсистеми пояснень (рис. 2) [3].

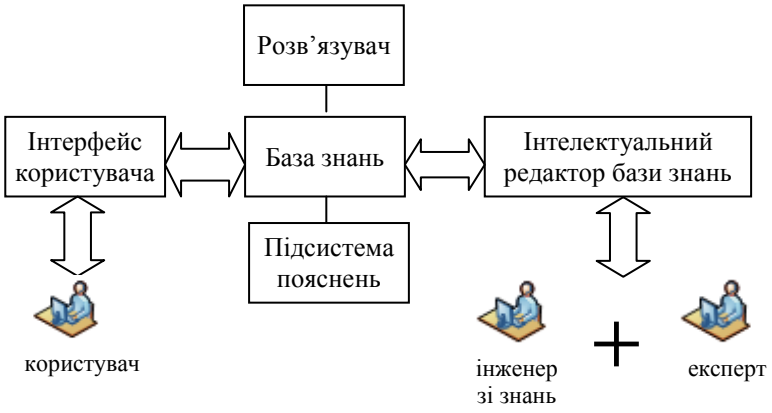


Рис. 2. Узагальнена структура експертної системи

Основою для формування архітектури конкретної експертної системи є система подання знань. Серед базових методів і моделей подання знань виділимо формально-логічні та продукційні моделі. У формально-логічних моделях інформацію для прийняття рішення подано як сукупність фактів і тверджень, що існують у вигляді формули з певною логікою. Знання відображаються сукупністю таких формул, а отримання нових знань зводиться до виконання певних процедур логічного висновку. У продукційних моделях знання подано у вигляді речень типу "ЯКЩО <умова> то <дія>" (продукцій).

Кожній з моделей подання знань притаманні свої переваги та недоліки. Зокрема, формальна логіка формально-логічних моделей відрізняється від людської логіки з нечіткою структурою. Це значно ускладнює процес побудови адекватних моделей подання знань. У продукційних моделях під час накопичення певної кількості продукції вони починають суперечити одна одній, що також ускладнює процес побудови таких моделей.

Розумним компромісом у створенні ефективних систем подання знань є використання базових моделей у поєднанні, тобто гібридних моделей, які можна вдосконалити й спрямувати на розв'язання задач у певній предметній галузі. Як правило, за такого поєднання одна з моделей буде домінуючою, а інша – виконуватиме роль супутньої. З огляду на предметну галузь доцільно використати в ролі домінуючої моделі продукційну модель, а в ролі супутньої – формально-логічну.

Для створення системи подання знань щодо вироблення рекомендацій із налаштування СКБД ORACLE як об'єкта управління запропонуємо модель параметрів СКБД для керівної структури серверу:

$$P = \{ pn, tp, dp, un, rg, vl, df, mp\}, \quad (1)$$

де pn – назва параметра в СКБД;

$tp = \{c, d, o, l, t\}$ – тип параметра,

де c – управління кеш-областю;

d – управління обміном з диском;

z – оптимізація виконання запитів;

r – управління резервним копіюванням та відновленням;

t – управління транзакціями.

$dp = \{1, 2, 3, 4\}$ – рівень динамічності зміни значень параметра,

де 1 – зміна параметра під час сесії;

2 – після перезапуску сесії;

3 – після перезапуску серверу.

un – одиниця виміру параметра; rg – тип подання значень параметра (у вигляді діапазону значень або множини допустимих значень); vl – поточне значення параметра (діапазон значень); df – рекомендоване значення; mp – спосіб управління параметром (за допомогою SQL-операторів або через редагування файлів конфігурації) [4, 5].

Для налаштування СКБД ORACLE застосовується відповідний алгоритм.

1. Прогін тестової процедури, збирання статистичних даних за допомогою пакета STATSPACK, зупинка інстанції ORACLE та встановлення початкових параметрів.

2. Запуск інстанції ORACLE.

3. Вироблення рекомендацій щодо налаштування. Зупинка інстанції ORACLE та встановлення рекомендованих параметрів. Якщо в результаті зміни параметрів можна досягти збільшення продуктивності СКБД, то відбувається зміна параметрів з урахуванням обмежень і перехід на крок 2, якщо подальше поліпшення неможливе, то перехід на крок 4.

4. Запуск інстанції ORACLE з налаштованими параметрами.

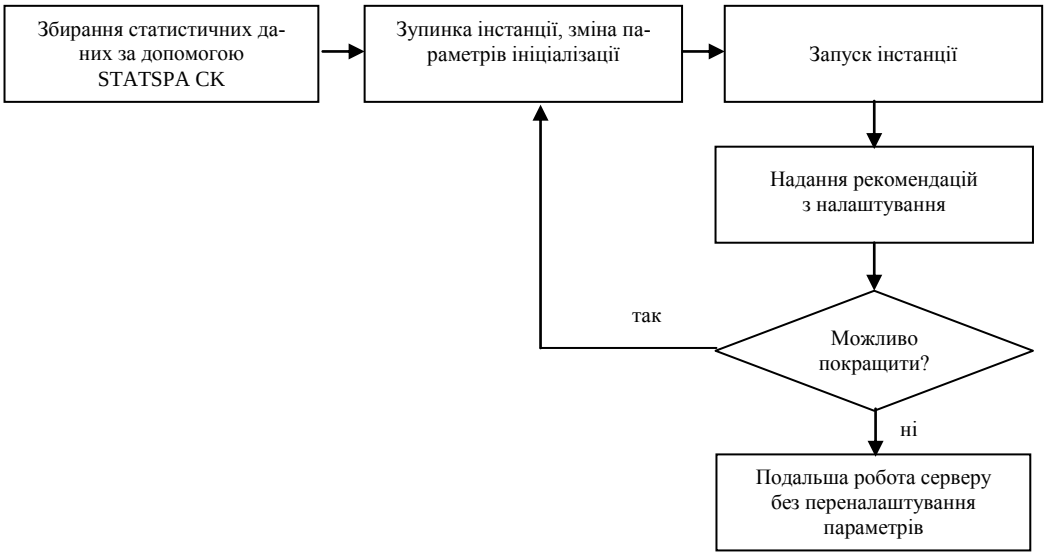


Рис. 3. Алгоритм налаштування СКБД ORACLE

Формування бази знань в експертній системі відбувається за допомогою спеціалізованого модуля – інтелектуального редактора бази знань (рис. 2). Інженер зі знань, ґрунтуючись на консультаціях з експертами, має сформувати структуру бази знань, що відповідає даній предметній галузі. Формування знань здійснюється безпосередньо в самій експертній системі, яка "самостійно" формує необхідні знання або отримує "нове" знання, не задане експертом.

Отже, розробка експертної системи включає в себе такі етапи:

1. Створення сценарію процесу логічного висновку (база фактів на етапі проектування).
2. Заповнення бази правил відповідності до синтаксису правил продукційної моделі.
3. Формування графа логічного висновку.

Сценарій процесу логічного висновку (список фактів ЕС). Найповнішою формою подання процесу логічного висновку є сценарій. Він складається з послідовності фактів бази знань – елементарних станів процесу логічного висновку (СПЛВ). Функціональне призначення СПЛВ – це опис усіх можливих даних і встановлення взаємозв'язків між цими даними. СПЛВ – це таблиця, кожен рядок якої заповнюється інформацією про факт, що має таку структуру:

$$<Факт>: = <Ідентифікатор><Опис><Відносини\ між\ фактами>. \quad (2)$$

Процес створення факту ґрунтується на таких етапах:

а) створення ідентифікатора факту;

б) формування опису факту;

в) створення списку інформаційних елементів, зокрема ідентифікаторів інших фактів і списків ідентифікаторів правил.

Створення ідентифікатора факту. Будь-який ідентифікатор факту має такий формат: <Велика латинська буква> [<число>]. Число нуль не використовується для формування ідентифікатора факту. Для нашої предметної галузі будемо використовувати в ролі ідентифікаторів фактів великі латинські літери, що є абревіатурою назви параметра, а число означатиме порядковий номер факту в таблиці.

Наприклад, для параметра BufferHitRatio, якщо він буде першим у базі фактів, ідентифікатор матиме вигляд "*BHR*[1]*"*.

Формування опису факту здійснюється користувачем і створюється відповідно до смислового навантаження факту. Опис факту використовується підсистемою пояснень експертної системи. Як опис факту будемо використовувати значення параметра та його стан. Це полегшить візуальне сприйняття і надасть таблиці СПЛВ більшої інформативності. Для параметра BufferHitRatio опис матиме такий вигляд:

"Відсоток попадань у буферний кеш нижче порогового значення".

Під час створення списку <відносини між фактами> визначимо всі можливі переходи до фактів, які можна встановити як наслідок з факту, якому належить даний інформаційний елемент, зі структурою:

<Відносини між фактами>:=<Ім'я><Ідентифікатор><Список ідентифікаторів правил>.

Детальніше:

а) ім'я – опис факту, який можна встановити як наслідок з факту, котрому належить даний інформаційний елемент;

б) ідентифікатор факту (Fnext) – як наслідок з факту, якому належить це ім'я;

в) список ідентифікаторів правил показує, в антецеденти яких правил входить факт, що йому належить даний інформаційний елемент, у консеквентах яких міститься факт Fnext. Ідентифікатори правил – числа, нумерація яких починається з одиниці.

Список <Відносини між фактами> може бути неповним, якщо факт кінцевий або ізольований. Факти в нашій предметній галузі переважно будуть ізольовані (кінцеві) через те, що зміна стану параметра в більшості випадків виконується за один крок (одна рекомендація щодо налаштування), тобто в нашому списку <Відносини між фактами> буде лише <Ім'я>. Список <Відносини між фактами> матиме такий вигляд:

"Збільшити значення параметра db_cache_size".

Подано рядок з таблиці СПЛВ, який буде містити структуровану інформацію про факт:

F:= <BHR[1]><Відсоток попадань у буферний кеш нижче порогового значення>
<Збільшити значення параметра db_cache_size >.

На основі даної структури побудуємо таблицю СПЛВ для деяких параметрів налаштування СКБД ORACLE (табл. 1) [2, 6].

Таблиця 1

Таблиця деяких станів процесу логічного висновку (СПЛВ)

№	Структура факту
1.	F: = <*BHR*[1]><Відсоток попадань у буферний кеш нижче граничного значення> <Збільшити значення параметра db_cache_size >
2.	F: = <*LCHR*[2]>< Відсоток попадань у бібліотечному кеші виходить за рамки допустимого діапазону > < Збільшити shared_pool >
3.	F: = <*DDHR*[3]> < Відсоток попадань у кеші словника даних виходить за рамки допустимого діапазону > < Збільшити shared_pool >
4.	F: = <*SR*[4]> <Набір параметрів, що регламентують операції сортування, виходить за рамки допустимого діапазону> <Збільшити sort_area_size або встановити workarea_size_policy = auto і pga_aggregate_target>
5.	F: = <*RSS*[5]> < Набір статистик, що визначають конкуренцію за сегменти відкачування, виходить за рамки допустимого діапазону> <Збільшити ємкість undo табличного простору>

Список правил бази знань ЕС. База правил – це впорядкована множина правил із таким форматом:

$$<Правило>:= <Ідентифікатор><Значення>, \quad (3)$$

де <ідентифікатор> – ідентифікатор правила; <значення> – саме правило, тобто правило, записане відповідно до синтаксису правил продукційної моделі подання знань.

У ролі частини <Правило> будемо використовувати позначення з використанням великої латинської літери "Р" та цифри, що позначатиме порядковий номер правила. У ролі частини <Ідентифікатор> застосуємо за аналогією з СПЛВ великі латинські літери, це абревіатура назви параметра, який оптимізується в даному правилі, але без цифрової частини для запобігання тавтології у списку правил.

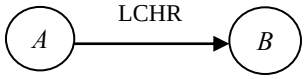
Побудуємо таблицю правил для налаштування деяких параметрів продуктивності серверу ORACLE відповідного формату (3).

Таблиця 2

Список деяких правил бази знань ЕС

№	Правило
1	*<P1>:= <BHR>*<ЯКЩО BufferHitRatio нижче порогового значення, ТО слід збільшити значення параметра db_cache_size >
2	*<P2>:= <LCHR>*<ЯКЩО LibraryCacheHitRatio виходить за рамки допустимого діапазону, ТО слід збільшити shared_pool, налаштувати параметри open_cursors, cursor_space_for_time, налаштувати shared_pool_reserved_size за допомогою v$shared_pool_reserved>
3	*<P3>:= <DDHR>*<ЯКЩО DataDictionaryHitRatio виходить за рамки допустимого діапазону, ТО слід збільшити shared_pool>
4	*<P4>:= <SR>*<ЯКЩО Sortsratio виходить за рамки допустимого діапазону, ТО слід збільшити sort_area_size або встановити workarea_size_policy = auto і pga_aggregate_target = auto >
5	*<P5>:= <RSS>*<ЯКЩО Rollbacksegmentstatistics виходить за рамки допустимого діапазону, ТО слід збільшити розмір сегментів відміни>

Побудова графа логічного висновку. Сформуємо граф логічного висновку для експертної системи, що розроблюється. У ролі моделі подання знань в експертній системі використовується продукційна модель. Вершини графа – факти бази знань, ребра характеризують механізм міркування експертної системи, тобто правила бази знань. Правило "ЯКЩО А, ТО В" відображено у графі ребром, що проходить з вершини *A* до вершини *B* (у списку інформаційних елементів факту *A* наявний факт *B*). Кожному ребру відповідає множина ідентифікаторів-правил, у частині "ЯКЩО" яких міститься вершина-факт, з якої виходить ребро, а в частині "ТО" – вершина-факт, до якої це ребро входить (рис. 4).



A – антецедент правила – "ЯКЩО LibraryCacheHitRadio" виходить за рамки "допустимого діапазону".

B – консеквент правила – "ТО слід збільшити shared_pool".

Рис. 4. Фрагмент графа логічного висновку

Кожна вершина може мати декілька ребер, які входять до неї з інших вершин, і декілька ребер, які виходять з неї до інших вершин. Механізм логічного висновку рухається від вершини до вершини. Після встановлення фактів починається перегляд ребер, які ведуть з установлених вершин-фактів, а також перегляд ідентифікаторів правил, пов'язаних з даними ребрами. Якщо правило, що відповідає ідентифікатору правила ребра, містить у частині "ЯКЩО" інші факти, то виконуються операції з даним фактом згідно з синтаксисом відповідного правила. Якщо результат цих операцій матиме логічне значення *true*, то факт, що стоїть у частині "ТО" правила, вважається встановленим і додається до бази знань. Далі можна розглядати відповідну вершину й ребра, що виходять з неї. Якщо правило, котре відповідає ідентифікатору правила ребра, не містить у частині "ЯКЩО" інших фактів, то факт, що стоїть у частині "ТО" цього правила, вважається встановленим.

Ребра поділяються на:

- ребра з єдиним ідентифікатором правила, що відповідає ребру;
- ребра з двома і більше ідентифікаторами правил.

Побудуємо граф логічного висновку для такого правила:

<P6>:= <RLBS><ЯКЩО *Redologbufferspace виходить за рамки допустимого діапазону*, ТО *слід збільшити розмір буфера журналів повторення, перенести журнали повторення на швидші пристрої, зменшити кількість створюваної redo інформації, налаштувати оптимальну частоту процесу checkpoint*>.

Оскільки антецедент даного правила простий і складається з одного факту, позначимо його через *A*. Консеквент правила складний тому, що містить три факти, один з яких складається з двох частин, два інші – з однієї. Позначимо факти таким чином:

A – "Redologbufferspace виходить за рамки допустимого діапазону";

B_1 – "збільшити розмір буфера журналів повторення";

B_2 – "перенести журнали повторення на більш швидкі пристрої";

C – "зменшити кількість створюваної redo інформації";

D – "налаштувати оптимальну частоту процесу checkpoint".

Кожному ребру відповідає ідентифікатор правила, яке спрацьовує в разі істинності факту *A*. Ребро між частинами B_1 і B_2 факту не має ідентифікатора тому, що воно доповнює факт B_1. Отриманий граф логічного висновку для даного правила являє собою орієнтований граф (рис. 5).

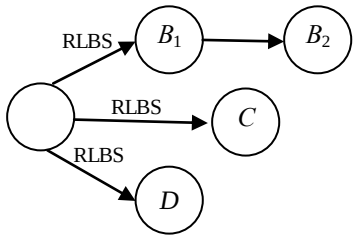


Рис. 5 Орієнтований граф логічного висновку правила

Графи логічного висновку інших правил будуються аналогічно. Будувати граф логічного висновку всієї системи (для всіх правил експертної системи) не має сенсу тому, що одні параметри будуть оптимізуватися незалежно від інших параметрів серверу ORACLE, тобто кожне правило матиме окремий граф. Система логічного висновку складатиметься з множини графів логічного висновку кожного окремого правила. Графи можуть мати спільні вершини у разі однакових антецедентів або консеквентів правил системи. Правила *P*2 і *P*3

мають однакову дію в консеквенті (табл. 2), але різні антецеденти. Побудуємо фрагмент графа логічного висновку експертної системи для таких правил (рис. 6):

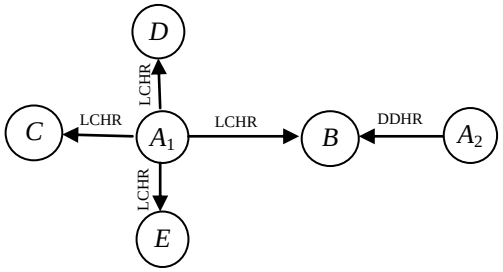


Рис. 6. Граф логічного висновку правил з однаковою дією в консеквенті

На рис. 6 позначення мають такий зміст:

LCHR, DDHR – ідентифікатори правил (ребер);

A_1 – "LibraryCacheHitRatio виходить за рамки допустимого діапазону";

A_2 – "DataDictionaryHitRatio виходить за рамки допустимого діапазону";

B – "збільшити shared_pool";

C – "найбільш часто використовувані процедури закріпити в бібліотекному кеші після старту екземпляра";

D – "налаштувати параметри open_cursors, cursor_space_for_time, session_ cached_cursors";

E – "провести налаштування shared_pool_reserved_size за допомогою v$shared_ pool_reserved".

У класичному пошуку цільової вершини (кінець міркувань експертної системи) використовується прямий механізм логічного висновку, тобто існує одна цільова вершина, після досягнення якої міркування мають припинитися. Однак у нашому випадку може й не бути однієї цільової вершини, тому що цілей може бути декілька. Під час роботи механізму логічного висновку можна досягнути однієї або кількох цілей або не досягнути жодної (через брак інформації для механізму логічного висновку, наприклад через "мізерність" вхідних даних). Тому, якщо хоча б одне правило бази знань спрацювало (відбувся хоча б один перехід у графі), механізм логічного висновку не припиняється. В іншому випадку механізм міркувань припиняє свою роботу і вважається досягнутою цільова вершина пошуку. Якщо вершини *A* і *B* вже помічені як встановлені (до якої входить ребро з *A*), тобто спрацювало, наприклад, правило "ЯКЩО *A,* ТО *B*", то це правило має спрацювати й наступного разу, що може призвести до зациклення роботи механізму логічного висновку. Щоб запобігти таким ситуаціям, використовується певний алгоритм: якщо розглядається вершина *A*, помічена як встановлена, то перш ніж виконати правило, в антецеденті якого розміщена вершина *A*, аналізуються ребра, що виходять з вершини *A,* і якщо вершина *B*, до якої входить дане ребро, також помічена як встановлена, то правила, що відповідають даному ребру і призводять до встановлення факту *B*, не вважаються такими, що спрацювали.

Таким чином будується формально-логічна модель, яка описує подання знань, що допоможе надалі формалізувати структуру й функціонування інтелектуальної системи в цілому.

На рис. 7 запропоновано модель системи надання рекомендацій налаштування СКБД ORACLE у вигляді діаграми класів. Ця модель відображає зв'язки між класами, що входять до системи.

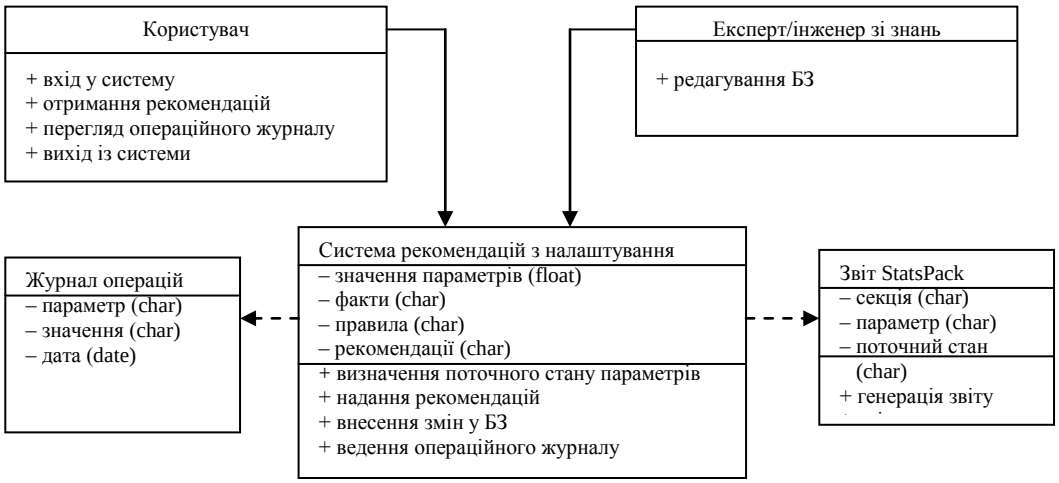


Рис. 7. Модель класів експертної системи

Висновки. У статті проаналізовано основні переваги та недоліки різних моделей подання знань в експертних системах, запропоновано принцип побудови експертних систем на базі гібридних моделей подання знань. У результаті проведених досліджень розроблено метод і модель налаштування СКБД ORACLE 10g на базі запропонованої гібридної моделі. Створена експертна система допомагає аналізувати параметри налаштування серверу СКБД ORACLE 10g за допомогою статистичних даних і давати необхідні рекомендації адміністратору СКБД щодо підвищення продуктивності серверу. Вперше запропоновано гібридну модель експертної системи, яку можна використовувати для налаштування параметрів серверів БД. Отримані результати можуть бути корисними для співробітників Департаменту митної служби, які працюють адміністраторами БД. Далі планується детальніший розглянути параметри налаштування БД, тобто приділити більшу увагу налаштуванням служб ORACLE NET.

Література

1. Плескач В. Л. Інформаційні системи і технології на підприємствах / В. Л. Плескач, Т. Г. Затонацька. – К. : Знання, 2011. – 718 с.

2. Oracle Cloud Computing Solutions [Електронний ресурс]. – Режим доступу : http:// www.docs.oracle.com/cd/B19306_01/rac.102/b14197/monitor.htm# RACAD976.

3. Модель представления знаний посредством объектов для построения интеллектуальных систем поддержки принятия решений / [В. В. Литвин, Д. Г. Досин , Р. Р. Даревич, Т. М. Пугач] // Штучний інтелект – 2006. – № 4. – С. 344–349.

4. Блажко А. А. Модели для автоматизированной оптимизации производительности СУБД / А. А. Блажко, А. Ю. Левченко, А. С. Пригожев // Радіоелектроніка і комп'ютерні системи. – 2010. – № 7. – С. 24–29.

5. Ситник В. Ф. Системи підтримки прийняття рішень : навч. посіб. / Ситник В. Ф. – К. : КНЕУ, 2004. – 614 с.

6. Волков Д. В. Оптимизация информационных систем на основе СУБД Oracle [Електронний ресурс] / Д. В. Волков // Jet Info. – 2004. – № 2. – Режим доступу : http://www.jetinfo.ru/ Sites/new/Uploads/2004_2.7BBAD 6EFC6554E8791CCBF730A438BA8.pdf.