функціональні можливості, забезпечуючи при цьому швидкий і надійний доступ до джерел інформації. Результати досліджень вказують на можливість побудови апріорі нескінченно довгих безнадлишкових рядів на множині ОСП та створення на їх основі нормативної бази для виробництва і проектування пристроїв інформаційно-вимірювальної техніки з поліпшеними технічними характеристиками за надійністю, швидкодією, функціональними можливостями.

1. ГОСТ 8032-84. Предпочтительные числа и ряды предпочтительных чисел. 2. Стахов А.П. Введение в алгоритмическую теорию измерения. – М., 1977. 3. БандирськаО.В. Стандартизація безнадлишкових рядів методом оптимальних структурних пропорцій//Автореф. канд.техн.наук. – Львів: ДУЛП, 2000. 4. Холл М. Комбинаторика. – М.: Мир. – 1970. 5. Різник В.В. Синтез оптимальних комбінаторних систем. – Львів. Вища школа, 1989.

УДК 681.3

І.В. Рішняк
Національний університет "Львівська політехніка",
кафедра "Інформаційні системи та мережі"

МОДЕЛЮВАННЯ ПРОЦЕСІВ ПРОЕКТУВАННЯ СХЕМ РЕЛЯЦІЙНИХ БАЗ ДАНИХ ТА ОЦІНЮВАННЯ РИЗИКІВ ПРИЙНЯТТЯ ПРОЕКТНИХ РІШЕНЬ

У роботі розглядаються основні класи задач, які можуть виникати при проектуванні схем реляційних баз даних у випадках існування часткових невизначеностей, нечіткостей та слабкоструктурованості. Проаналізовано проблемні ситуації, що виникають при різних постановках задач, а також деякі методи вирішення цих ситуацій.

The basic classes of problems which may arise at designing circuits of relational databases in cases of existence of partial uncertainty, discrepancies and weak structuring are considered in the presented of work. Problem situations which arise at different statements of tasks, and also some methods of these situations solution are analysed.

Інформаційне моделювання проблемних областей, яке в своїй основі містить процеси проектування схем баз даних та знань, потребує розробки адекватних технологічних процедур для розв'язання таких задач:

1) якомога точнішого опису атрибутів вказаної предметної області;
2) визначення суттєвих взаємозв'язків та відношень між атрибутами;

3) розгляд можливих змін атрибутів та їх взаємозв'язків між собою з плином часу;
4) надання можливості користувачу швидко отримувати необхідну та якісну інформацію щодо даної предметної області;
5) надання можливості користувачу швидко поповнювати, видаляти та змінювати дані, не порушуючи їх цілісності;
6) надання можливості користувачу отримувати звіти в аналітичній, табличній та графічній формах.

При проектуванні схем реляційних баз даних, які сьогодні є найпопулярнішою платформою створення інформаційних систем, природно виникають задачі, в яких присутні нечіткості та слабкоструктурованості. Для розв'язання цих задач природним є створення експертних систем, які забезпечували б їх розв'язання.

Тут розглянуто деякі класи нових задач проектування схем реляційних баз даних, що відображають реальні ситуації, які зустрічаються на практиці. Викладення подається на даному етапі узагальнено, що дає змогу надалі здійснити відповідну конкретизацію.

Наведемо означення основних понять, що використовуються у статті [2,3,6]:

- Об'єкт реального світу характеризується певною множиною параметрів - *атрибутів* (A_1, A_2, ..., A_n).
- *Предметна область* – деяка сукупність об'єктів реального світу та відношень між ними, характеристики (атрибути) яких та взаємозв'язки між якими подаються відповідною базою даних;
- *Схема відношення* складається з імені відношення R, множини атрибутів (A_1,A_2, ..., A_n), яка якісно описує параметри відношення, та системи залежностей даних на цій множині атрибутів;
- *Схема реляційної бази даних* – множина схем відношень, які входять до складу бази даних;
- *Функціональна залежність* описує зв'язок між двома підмножинами атрибутів відношення і позначається $A_1,A_2,\ldots,A_m \rightarrow A_{m+1},A_{m+2},\ldots,A_n$ (атрибути $A_{m+1},A_{m+2},\ldots,A_n$ функціонально залежать від атрибутів $A_1,A_2,\ldots,A_m$);
- *Булева функція є повністю визначеною,* коли вона на всіх наборах значень булевих змінних приймає чітко визначене значення 0 або 1;
- *Частково визначена булева функція* – булева функція, яка, при деяких значеннях наборів булевих змінних, може набувати значення як 0, так і 1, тобто може бути невизначеною.

Задамо інформаційну модель фрагменту реального світу у вигляді схеми реляційної бази даних.

Припустимо, що маємо систему предикатів $\{P_i\}$, яка описує деякий фрагмент заданої предметної області. Об'єднаємо функціональні залежності, що існують між атрибутами предметної області в систему. Очевидно, що система функціональних залежностей повинна достатньо широко описувати усі властивості фрагменту реального світу, звичайно, за умови, що повністю задані всі предикати і можливі відношення та взаємозв'язки між ними. Отримаєм таку систему:

$$\forall\left(i=\overline{1,n}\right):\left(X_i \to Y_i\right)$$

Згідно з теоремою Делобеля–Кейсі перейдемо від системи функціональних залежностей до задання відповідної булевої функції *f(x)*, яку іменуватимемо булевим еквівалентом.

Припустимо, що задано деяке відношення *r* зі схемою *R*.

Кожному атрибуту *A* схеми *R* поставимо у відповідність булеву змінну x_i, де $1 \le i \le n$.

При заданні залежності вигляду:

$$A_1A_2A_3 \to A_7A_8 \qquad (1)$$

булевий еквівалент для неї можна записати як

$$x_1x_2x_3\overline{x_7x_8} \equiv x_1x_2x_3\overline{x_7} \vee x_1x_2x_3\overline{x_8}, \qquad (2)$$

що є фактичним поданням правила декомпозиції правої частини функціональної залежності.

Будь-яка повністю задана булева функція *f(x)* в просторі E^n має імпліканти нуля *(0)* та одиниці *(1)* (тобто набуває значення 0 та 1 відповідно). Повна множина наборів булевих змінних у просторі E^n розбивається на дві підмножини *A* та *B*, тобто $A \cup B = E^n$, $A \cap B = 0$; множина *A* складається з тих наборів булевих змінних x_i, $1 \le i \le n$, де булева функція *f(x)* набуває значення одиниця, а множина *B* – з тих наборів, де булева функція *f(x)* набуває нульові значення.

Процедура проектування схеми реляційної бази даних розробником (системним аналітиком). Складається з наступної послідовності кроків:

1. Визначення схеми *R* відношення *r* та встановлення залежності між атрибутами.

Для відношення *r* визначається схема цього відношення $R[A_1, A_2, ..., A_n]$, де $A_1, A_2, ..., A_n$ можливі атрибути відношення. Далі кожному атрибуту A_i, ставиться у відповідність булева змінна x_i. Після цього між атрибутами відношення встановлюються відповідні залежності і кожній функціональній залежності ставиться у відповідність булевий еквівалент (використовуючи правила *(1)* та *(2)*). Ця процедура виконується на семантичному рівні, а також із застосуванням інших методів, наприклад, статистичного.

2. Отримання наборів булевих термів і опис предметної області.

В результаті першого кроку отримується система залежностей даних і, відповідно, наборів булевих термів. Диз'юнкція цих термів дасть булеву функцію $f(x_1,x_2,..,x_n)$, яка буде набувати значення *1*, якщо буде підтримуватися хоча б одна з функціональних залежностей і *0*, якщо жодна з залежностей не підтримуватиметься. Згідно з теоремою Делобеля–Кейсі отримана булева функція використовуватиметься як еквівалент інформаційного опису фрагменту реальної предметної області. Очевидно,що для маніпулювання булевим еквівалентом є справедливими всі правила (операції) булевої алгебри.

В практиці проектування схем реляційних баз даних реалізовуються з деякими особливостями. Наявність чи відсутність функціональних залежностей у системі може бути встановлена не достеменно. Тобто, система буде не цілком детермінованою.

При постановці задачі, коли у просторі E^n буде існувати певна часткова невизначеність, в результаті проектування схеми реляційної бази данних, що була описана раніше, отримаємо деяку частково-визначену булеву функцію. Це означає, що для цієї функції ми будемо мати різні набори булевих змінних. Причому поряд із термами на яких булева функція буде отримувати чітко визначені значення 0 чи 1, будуть зустрічатися терми на яких значення функції буде неоднозначним. Тобто, не можна буде впевнено стверджувати, що функція набуває на них конкретного нульового чи одиничного значення.

В результаті отримуємо математичну модель часткової невизначеності, яка формується під час описання схеми реляційної бази даних. Ця модель подається у вигляді часткової невизначеності булевого еквіваленту, за допомогою якого описувалась відповідна схема. Вже на цьому етапі стають очевидними переваги роботи з булевим еквівалентом:

- зображення булевого еквівалента у вигляді однієї інтегрованої булевої функції, а не у вигляді системи функціональних залежностей;
- при розв'язанні задачі часткової невизначеності у недетермінованій системі в булеву функцію значно простіше вводити додаткові зміни;
- з булевим еквівалентом, що важливо, можна працювати за допомогою правил (операцій) булевої алгебри.

Розглянемо детальніше вплив часткової невизначеності на вирішення проблеми. Розв'язання задачі не надто ускладниться при необхідності вести деяку додаткову залежність даних. Проблема виникає у випадку, коли неможливо точно сказати, в яких випадках існує, а в яких не існує описана залежність, тобто якщо має місце фактор випадковості. Найпростішим способом подолання цієї проблеми є використання генератора випадкових чисел. Тобто, булева функція буде отримувати значення нуля або одиниці відповідно до того, яке випадкове число видасть генератор. Але такий метод є прийнятним тільки у випадку незначної кількості залежностей. Якщо кількість невизначеностей зростає, то і зростає ймовірність помилки в описанні схеми реляційної бази даних. Причому, якщо між частково визначеними залежностями існують зв'язки, то похибка буде наростати не лінійно, а, наприклад, за геометричною прогресією. Власне тут розглядається проблема вибору принципів, правил та методів, що дають змогу однозначно відносити ту чи іншу диз'юнкту частково невизначеної булевої функції до імпліканти *0* або *1*.

Розглянемо основні класи задач, що можуть виникати при проектуванні схем реляційних баз даних у випадках існування часткової невизначеності. Основними характеристиками системи у цьому випадку є:

- Залежність від часу. Система може бути статичною або динамічною. Тобто, залежності та зв'язки між атрибутами можуть залишатися незмінними у часі або, навпаки, зникати, з`являтись чи просто змінюватись з плином часу.
- Кількість розробників. Звичайно під розробником можна розуміти як одну людину, так і колектив (як єдиний організм). Кожен з розробників має своє суб'єктивне бачення проблеми та її вирішення і відповідно може мати вплив на думку колеги або ,навпаки, бути залежним від його думки.

- Недетермінованість залежностей між атрибутами відношення, яка виражається у невизначеності залежностей по своїй суті (недостатність інформації. недостовірність інформації, нечіткість формулювань та ін.).

Подамо основні типи задач у вигляді таблиці.

Типи задач, що виникають під час проектування

тип задачі	часові характеристики системи	детермінованість системи	кількість розробників
1	статична	детермінована	один
2	динамічна	недетермінована	декілька
3	статична	детермінована	декілька
4	статична	недетермінована	один
5	статична	недетермінована	декілька
6	динамічна	детермінована	один
7	динамічна	детермінована	декілька
8	динамічна	недетермінована	один

Тип 1. *Система статична, із чітко детермінованими залежностями, проектується одним виконавцем.*

Нехай над створенням схеми реляційної бази даних об'єкта реального світу працює один проектувальник. Припустимо, що він володіє абсолютно повною інформацією про даний об'єкт, про внутрішні та зовнішні фактори, що впливають на його стан, про взаємозв'язок факторів між собою. Іншими словами, розробник має вичерпну інформацію про об'єкт та середовище, що його оточує, а також інформацію про залежності даних між ними.

У такому випадку схема реляційної бази даних виглядає тривіально. Якщо розробник вже має описання предметної області у вигляді предикатів, то, зробивши перехід від системи функціональних залежностей до булевого еквіваленту, він отримає схему реляційної бази даних, що буде повністю задавати об'єкт досліджень. Строго кажучи, така система може існувати тільки на папері і може бути ідеальною моделлю для системи, що описує деяку предметну область.

Наступний тип задач є цілковитою протилежністю описаним раніше, але в той же час ці задачі є найбільш наближеними до реальності.

Тип 2. *Система недетермінована, динамічна, проектування проводиться декількома розробниками.*

Цей випадок є найбільш цікавим і водночас найскладнішим. Очевидно, що проблема полягає у взаємозв'язках між різними видами невизначеностей. Окрім того, що кожен проектант вирішує задачу зі своєї точки бачення проблеми, він не володіє достатньою і достовірною інформацією про атрибути та функціональні залежності між ними. Все це ускладнюється тим, що система змінюється в часі і відповідно, можуть змінюватись як самі атрибути, так і взаємовплив між ними.

Це найбільш загальний тип задач, що виникають при проектуванні схем реляційних баз даних. Очевидно, що задачі цього типу є достатньо складними для розв'язання.

Далі розглядатимамо типи задач, які є частковими випадками задач *типу 2.*

Тип 3. Система із детермінованими залежностями, статична, проектується кількома виконавцями.

Припустимо, що над описанням деякого об'єкта реального світу працює не один, а декілька розробників. В такому випадку проблемну область варто розділити на окремі підобласті, які є фукнціональною або структурною декомпозицією предметної області. Всі розробники працюють окремо, кожен зі своєю конкретною підобластю. Нехай кількість розробників дорівнює *n*. Очевидно, що предметна область буде поділена на *n* підобластей відповідно до профілю розробників і кожен з них подасть своє бачення множини атрибутів та залежностей між даними. У результаті на першому етапі проектування отримаємо *n* булевих еквівалентів, кожен з яких описує деяку підобласть загальної проблеми. Треба відзначити, що в кожній булевій функції кількість булевих змінних, від яких вона залежить, може бути різна.

Розглянемо два випадки. У першому випадку будемо вважати, що предметна область розбита на підобласті так, що жодна пара розробників не має спільних атрибутів. У такому випадку множини всіх атрибутів та булевих змінних об'єднуються. Записується нова булева функція як диз'юнкція булевих еквівалентів по кожній предметній підобласті. Отримана булева функція буде описувати предметну область загалом. Цей випадок є дещо ідеалізованим. Більш реальним є випадок непорожнього перетину множин атрибутів різних розробників. Тобто деякі атрибути є спільними. Отже, виникають колізії, які потребують розв`язання.

Наведемо наступний приклад. Нехай ми маємо двох розробників. Припустимо, що за період роботи розробник, позначимо його $П_1$, визначив множину атрибутів $(A_1,..,A_k..,A_l..,A_n)$, а інший розробник, позначимо його $П_2$, визначив множину атрибутів $(A_1,..,A_k..,A_l..,A_m)$. Тобто атрибути $(A_k,A_{k+1},..,A_l)$ є спільними. Нехай під час дослідження взаємозв'язків між атрибутами розробник $П_1$ встановив таку залежність:

$$A_k A_{k+1} A_{k+2} \rightarrow A_{l-1} A_l$$

для якої він записав відповідний булевий еквівалент:

$$x_k x_{k+1} x_{k+2} \overline{x_{l-1} x_l} \equiv x_k x_{k+1} x_{k+2} \overline{x_{l-1}} \vee x_k x_{k+1} x_{k+2} \overline{x_l}$$

і вивів, що результуюча булева функція *f(x)* при наявності такої залежності є імплікантою 1.

В той же час розробник $П_2$, враховуючи свою множину атрибутів, вивів, що за наявності такої залежності результуюча булева функція набуває значення 0.

Отже, спостерігається колізія, для якої слід розробити коректну процедуру її усунення.

Звичайно розробники можуть обговорити наявну ситуацію і прийти до спільного розв'язку, наприклад, за допомогою введення додаткових булевих змінних, введення нових атрибутів або залежностей даних.

Тип 4. Система статична, з недетермінованими залежностями, розробляється одним проектантом.

Нехай над створенням схеми реляційної бази даних об'єкта реального світу працює один розробник. При проектуванні він виявив, що на одних наборах даних функція *f(x)* завжди є імплікантою *1*, а на інших – завжди є імплікантою *0*. Але поряд зцими наборами існують такі, на яких булева функція *f(x)* може набувати як значення *0*, так і значення *1*, залежно від зовнішніх факторів.

Для вирішення цієї проблеми можна використати генератор випадкових чисел.Але, як вже зазначалось раніше, якщо кількість нечітких залежностей зросте, то ймовірність адекватності спроектованої моделі зменшиться. В даному випадку для розв'язання поставленої проблеми хороші наближені результати можуть дати методи математичної статистики.

Оскільки об'єкт існує в реальному масштабі часу, то за певний проміжок часу накопичиться достатньо велика кількість статистичних даних, які дадуть змогу з достатньо великою точністю, залежно від ситуації, прогнозувати результат тої чи іншої неадекватної залежності. Для таких залежностей доцільно розраховувати математичне сподівання та реалізовувати статистичне оцінювання

Також доцільно розглянути кореляцію атрибутів. Набуття функцією *f(x)* різних значень на одних і тих же термах є значно імовірнішим у випадку, коли атрибути мають між собою певні залежності. Отже, якщо ввести поняття кореляції як числової характеристики, що показує, наскільки випадкові величини залежні між собою, можна з достатньою імовірністю прогнозувати до якої множини (імплікант *1* чи імплікант *0*) віднести той чи інший терм.

Тип 5. Система статична, із недетермінованими залежностями, проектується кількома розробниками.

Розглянемо попередній варіант задачі з врахуванням того, що в проектуванні бере участь не один, а декілька розробників.

Очевидно, що ситуація ускладниться. Кожний розробник працює у своїй конкретній предметній підобласті. У кожного є своя множина атрибутів з визначеними між ними зв'язками та відношеннями. Кожний проектувальник маніпулює відповідно з різною кількістю булевих змінних..

Очевидно, колізії в цьому випадку є достатньо складними навіть, якщо в проектувані бере участь лише два розробники. Оскільки, якщо колізії будуть виникати у випадках коли функція є частково не визначена, доведеться одночасно розв'язувати дві задачі:

- усунення невизначеності;
- вирішення самої невизначеності;

У ситуації, коли між розробниками виникають колізії на тих наборах значень, де кожний розробник має недовизначену функцію, спрогнозувати результат буде практично неможливо. В цій ситуації визначальною є поведінки "арбітра" та його об'єктивне бачення проблеми.

Перейдемо до типів задач, що виникають у динамічній системі, тобто, коли є ймовірність зміни залежностей та зв'язків у часі.

Тип 6. Система динамічна, із детермінованими залежностями, розробляється одним виконавцем.

Припустимо, що над створенням схеми реляційної бази даних об'єкта реального світу працює один системний аналітик. Він має детальну і достовірну інформацію про заданий об'єкт, про всі внутрішні та зовнішні чинники, що впливають на стан об'єкта та про взаємозв'язки цих чинників між собою. Тобто, проектант володіє інформацією про об'єкт та середовище, що його оточує, а також про залежності між ними. Однак у процесі дослідження факторів, що впливають на об'єкт, та самого об'єкта виявилось, що встановлені в початковий момент часу t_0 функціональні залежності змінюються в часі. Вони можуть зникати чи поновлюватись. Тобто в даний момент часу певні залежності даних можуть існувати, а можуть і не існувати.

Для розв'язання такого класу задач можна спробувати встановити певні статистичні закономірності, наприклад:

- періодичність правдивості та хибності виведених закономірностей;
- частоту зміни значення функції *f(x);*
- періодичність "довгих нулів" та "довгих одиниць" і т.д;

Час в даному випадку можна розглядати як дискретну величину. При виконанні певних умов у постановці задачі час можна ввести за допомогою додаткового атрибуту і будувати залежності даних з його врахуванням.

Тип 7. система динамічна, із чітко детермінованими залежностями. у проектуванні бере участь декілька виконавців.

Нехай над описом об'єкта реального світу працюють декілька розробників. Предметна область у такому випадку розбивається на окремі підобласті. Проектанти працюють окремо один від одного. Кожен проектант працює у виділеній йому конкретній підобласті. Нехай кількість розробників дорівнює *n.* Кожен розробник відповідно описує свою множину атрибутів та залежностей між даними. В процесі дослідження своїх підобластей окремі розробники виявили, що задані в початковий момент часу t_0 функціональні залежності

можуть змінюватись в часі, з'являтсь або зникати. В даному випадку можливі колізії описати достатньо проблематично.

Стає очевидним, що колізії можуть бути настільки складними, що реальний метод їх розв'язання може бути доволі складним. У даному випадку без додаткового опрацювання досвіду чи внесення нових відомостей нічого не можна сказати про справедливість тих чи інших припущень. Цілком можливо, що при виникненні таких колізій, за відсутності обґрунтування математичних методів їх розв'язання, слід покладатися на практичний досвід розробника.

Тип 8. Система динамічна, із недетермінованими залежностями, проектується одним виконавцем.

Нехай над створенням схеми реляційної бази даних об'єкта реального світу працює один системний аналітик. При дослідженні об'єкта та факторів, які впливають на стан об'єкта, розробник виявив таку ситуацію. Існують такі функціональні залежності, що передаються відповідними наборами змінних, на яких булева функція *f(x)* може набувати як нульового значення, так і одиничного. Окрім цього розробник виявив, що встановлені в початковий момент часу функціональні залежності можуть змінюватись в часі.

Вирішення колізій для даного типу задач значно ускладнюється тим, що функціональні залежності (як повністю, так і частково визначені) можуть змінюватись у часі і відповідно переходити зі стану повної визначеності у стан часткової невизначеності.

Отже, ми розглянули основні типи задач, що виникають при проектуванні схем реляційних баз даних, у системах, основними характеристиками яких є: невизначеність залежностей та зв'язків між атрибутами відношення; різна кількість виконавців проекту; постійна динаміка розвитку системи, тобто, зміна взаємовпливів між атрибутами у системі.

Підсумовуючи описане раніше, перерахуємо можливі методи розв'язання колізій, що виникають при різних постановках задач і зробимо такі висновки:

1. Метод задання генератора випадкових чисел. Залежно від згенерованог значення булевий еквівалент набуває нульового або одиничного значення. Очевидно, що цей метод є найпростіший та досить швидкодійний. Але він має недоліки, головним з яких єшвидке зростання похибки при достатньо великій кількості невизначеностей. Для складних задач, у яких задіяно велику кількість змінних, цей метод неприйнятний, оскільки похибка може зростати надзвичайно швидко.

2. Метод введення вагової функції. Кожній ситуації ставиться у відповідність число, що відображає ймовірність, з якою ситуація може відбутися. При перемноженні цього числа на вагову функцію отримаємо відповідь на поставлену задачу.

3. Статистичні методи. Вони грунтуються на законах математичної статистики. Ці методи мають суттєві переваги над попередніми методами при умові швидкого накопичення даних. Працюючи в реальному масштабі часу, система збирає необхідну інформацію. Після того, як система за деякий відтинок часу накопичить достатньо велику

Кожній ситуації ставиться у відповідність число, що відображає ймовірність, з якою ситуація може відбутися. При перемноженні цього числа на вагову функцію отримаємо відповідь на поставлену задачу.

3. Статистичні методи. Вони ґрунтуються на законах математичної статистики. Ці методи мають суттєві переваги над попередніми методами при умові швидкого накопичення даних. Працюючи в реальному масштабі часу, система збирає необхідну інформацію. Після того, як система за деякий відтинок часу накопичить достатньо велику кількість стохастичної інформації, в ній починають спрацьовувати закони математичної статистики. Необхідні рішення щодо вирішення колізій приймаються системою згідно цих законів.

4. Експертні системи. На етапі збирання інформації система отримує ззовні дані про те, як людина, враховуючи власний досвід, приймає рішення щодо розв'язання колізій. Відповідно система накопичує надану інформацію, і через деякий час може бути експертом. Зручним є те, що при такому підході в системі зберігається досвід проектувальника. Недоліками третього і четвертого методів є велика затримка в часі при накопиченні початкової інформації та необхідність її періодичного поновлення.

Слід відзначити, що розглянуті в роботі ситуації подані як попередні результати. Запропонований підхід можна використовувати для оцінювання ризиків прийняття рішень в складних системах, що функціонують в умовах слабкої структурованості [5]. Відображення залежності даних у множину "дозволених", "недозволених", "потенційно можливих" викликає певну проблемну ситуацію, яку необхідно оцінити на попередньому етапі проектування. Надалі дослідження будуть продовжені і зосереджуватимуться на формуванні системних відповідей на такі запитання:

- з чим пов'язані ризики прийняття того чи іншого рішення по структурі системи?
- які наслідки можуть викликати певні рішення і як їх слід оцінювати на етапі аналізу проблемної області та проектування схеми реляційної бази даних?
- які процедури зменшення ризиків на етапі інформаційного моделювання та проектування схеми реляційної бази даних?

В цілому проблема оцінки ризиків прийняття того чи іншого проектного рішення щодо структури системи залежності даних є важливою як з теоретичного, так і з практичного погляду, а її вирішенню будуть присвячені наступні публікації.

1. Ульман Дж.Д. Введение в системи баз данних. /Пер. с англ. –М.:Изд."Лори", 2000 2. Дейт К.Дж. Введение в системы баз данных. 7-е издание/Пер. с англ.–М.–СПб.–К.:Изд.дом "Вильямс", 2001 3. Мейер Д. Теория реляционных баз данных –М.: Мир. 1987 4. Bochman D., Steinbach B. Logikentwurf mit XBOOLE: /Vertlag Technik, Berlin 1991 5. Elisa Barnito, Barbara Catania, G.P.Zarri Intelligent Database Systems: Addison Wesley 2001 6. Пасічник В. Структуризація даних на основі аналізу класів предикатів //Вісн. ДУ"ЛП"// –Л.,1998, №330 Abramson, D., Krishnamoorthy, M., Dang, H. Simulated Annealing Cooling Schedules for the School Timetabling Problem (1997). http://www.rdt.monash.edu.au/~davida/papers/cool.ps.Z (5/10/2000) 7. Banks, D., van Edmonton, P. A Heuristic Incremental Modeling Approach to Course Timetabling. ftp://ftp.cs.bgu.ac.il/pub/people/am/HSTT_AI98.ps.Z (5/10/2000) 8. Bertino E., Foscoli P. Index Organizations for Object-Oriented Database Systems. // Knowledge and engineering. – 1995. –

April, – Vol. 7. – N2. – p. 193–209. 9. Boar B.H. The art of strategic planning for information technology // Crafting Strategy for the 90s. – John Wiley&Sons, 1993. – p. 37–43. 10. Caldeira, J.P., Rosa A.C. School Timetabling using Genetic Search (1997). http://laseeb.ist.utl.pt/publications/papers/jpcaldeira/Patat.ps.gz (5/10/2000) 11. Courtois, P. On time and Space Decomposition of Complex Structures // Communications of the ACM. – 1985. – Vol. 28(6). – p. 596. 12. Culberson, J. C. Iterated Greedy Graph Coloring and the Difficulty Landscape (1992). ftp://menaik.cs.ualberta.ca/pub/TechReports/1992/TR92-07/TR92-07.ps.Z (5/10/2000) 13. E. Bertino, B Catania, G.P.Zarri Intelligent Database Systems. – Addison-Wesley, 2001. – 452p. 14. Englemore, R. and Morgan, T. Blackboard Systems. // Wokingham, England: Addison-Wesley. – 1988. – p. 2-20. 15. Erben, W., Keppler, J. A Genetic Algorithm Solving a Weekly Course-Timetabling Problem (1995). http://www-home.fh-konstanz.de/~erben/icptat95.ps (5/10/2000) 16. Erman L., Lark J., Hayes-Roth F. An Environment for Engineering Intelligent Systems. // IEEE Transactions on Software Engineering. – 1998. – Vol. 14(12). – December. – p. 1758. 17. Feigenbaum E.A. The art of artificial intelligence: Themes and case studies of knowledge engineering. // The fifth International Joint Conference on Artificial Intelligence. – Boston: MIT, 1977. – p. 1014–1029. 18. Jacobson I., Christerson M., Jonsson P., Overgaard G. Object-Oriented Sotware Engineering – A Use Case Driven Approach. – Reading, MA: Addison-Wesley; ACM Press, 1992.