УДК 004.94 http://doi.org/10.15407/pp2021.03.040

В.А. Резніченко

60 РОКІВ БАЗАМ ДАНИХ (частина перша)

Наводиться огляд досліджень і розробок баз даних з моменту їх виникнення в 60-х роках минулого століття і по теперішній час. Виділяються наступні етапи: виникнення і становлення, бурхливий розвиток, епоха реляційних баз даних, розширені реляційні бази даних, постреляційні бази даних і великі дані. На етапі становлення описуються системи IDS, IMS, Total і Adabas. На етапі бурхливого розвитку висвітлені питання архітектури баз даних ANSI/X3/SPARC, пропозицій КОДАСИЛ, концепції і мов концептуального моделювання. На етапі епохи реляційних баз даних розкриваються результати наукової діяльності Е. Кодда, теорія залежностей і нормальних форм, мови запитів, експериментальні дослідження і розробки, оптимізація та стандар¬тизація, управління транзакціями. Етап розширених реляційних баз даних присвячений опису темпоральних, просторових, дедуктивних, активних, об'єктних, розподілених та статистичних баз даних, баз даних масивів, машин баз даних і сховищ даних. На наступному етапі розкрита проблематика постреляційних баз даних, а саме, NOSQL-, NewSQL- і онтологічних баз даних. Шостий етап присвячений розкриттю причин виникнення, характерних властивостей, класифікації, принципів роботи, методів і технологій великих даних. Нарешті, в останньому розділі дається короткий огляд досліджень і розробок по базах даних в Радянському Союзі.

Ключові слова. Типи баз даних: ієрархічна, мережева, реляційна, навігаційна, темпоральна, просторова, просторово-темпоральна, просторово-мережева, об'єктів, що переміщуються, дедуктивна, активна, об'єктно-орієнтована, об'єктно-реляційна, розподілена, паралельна, масивів, статистична, багатовимірна, машина баз даних, сховища даних, NoSQL, ключ-значення, стовпчикова, документно-орієнтована, графова, мультимодельна, хмарна, наукова, багатозначна, XML, NewSQL, онтологічна, великі дані.

Етап 1. Становлення баз даних (1960 – 1970)

60-і роки – це період усвідомлення необхідності відокремлення даних від програм, кристалізації вимог до такої незалежної сукупності даних. Відтак, це період зародження й успішного становлення технологій баз даних, формування їхніх методологічних основ, становлення концепції моделі даних і появи перших двох класичних моделей – ієрархічної і мережевої, народження індустрії програмного забезпечення систем баз даних, а також – організаційного формування спільноти спеціалістів цієї галузі.

На початку 60-х років комп'ютери почали впроваджуватися на виробництві. Звісно, це здійснювали великі компанії, здатні придбати таке дороге обладнання. Комп'ютери почали використовуватись для автоматизації виробничих процесів, включно з обліком отримуваної сировини і деталей, виробленої продукції, персонала, тощо. Комп'ютери ставали інструментом збереження і обробки великих обсягів даних. До того ж стало очевидним, що технологія створення автоматизованих систем, за якої існував тісний звязок між даними і програмами, які їх використовують, є нежиттєздатною. Адже будь – які незначні зміни в структурі баз даних призводили до необхідності переписувати програми. З поступовим ускладненням структури даних і зростанням їхнього об'єму, збільшенням кількості користувачів, а також – інтенсивності використання даних, подібний підхід призводив до краху систем.

Це викликало усвідомлення того, що необхідно розірвати такий зв'язок і уможливити незалежне існування даних і програм, що й стало основою появи в інформатицінапрямку, який згодом отримав назву «бази даних».

Аби зрозуміти, в яких умовах зароджувалися бази даних, зазначимо, що це був час комп'ютерів фактично без операційної системи, з 64 КБ оперативної пам'яті. Носіями введення даних були перфокарти і перфострічки, а відповідальними за зовнішню память були переважно магнітні стрічки. І лише деякі компанії могли дозволити собі магнітні диски з об'ємом у 5 МБ і розмірами, що

перевищували тристулкову шафу разом із антресолями. Зрештою спілкування людини з комп'ютером відбувалося через пульт управління або, в кращому випадку, через друкарську машинку.

Завантаження жорсткого диску
на 5МБ компанії IBM

Системи IDS. 1960-го року невелика команда із General Electric, яка автоматизувала бізнес-процеси, розпочала проєктування системи Integrated Data Store (IDS) – інтегроване сховище даних під керівництвом Чарльза Бахмана (Charles William Bachman). Наприкінці 1962 року прототип цієї системи був завершений, а на початку 1964 року випущена перша промислова версія IDS [14-16]. Її поява знаменувала еру баз даних і зірковий шлях Бахмана.

Чарльз Бахман

В IDS було вперше втілено те, що нині вважається основними функціями системи управління базами даних. IDS виконувала функцію посередника між прикладними програмами і файлами, в яких зберігалися дані. Програми не могли напряму маніпулювати даними. Натомість вони мали звертатися до системи IDS, аби та виконувала відповідні дії від їх імені. Як і сучасні системи управління базами даних, IDS дозволяла створювати, зберігати і маніпулювати метаданими, хоча робилося це занадто примітивно. В IDS у найпростішому вигляді були реалізовані функціональні можливості, котрі згодом отримали назву «незалежності даних від програм». Бахман розробив у IDS інноваційну на той час систему «Диспетчер проблем» (Problem Controller), що стала прообразом системи управління транзакціями. В IDS також була спроєктована і реалізована система резервного копіювання і відтворення даних на магнітних стрічках. Зрештою, була передбачена функція заборони доступу до певних частин бази даних конкретним користувачам. IDS стала прообразом системи управління базами даних, що підтримувала мережеву модель даних. Система IDS розвивалась, удосконалювалась і використовувалась упродовж десятків років. І нині IDS використовується в деяких компаніях і демонструє відмінні результати продуктивності на терабайтних масивах даних.

1973 року Бахман був нагороджений найпрестижнішою в галузі інформатики премією Алана Тьюрінга за видатний внесок у технологію баз даних. Він був першим лауреатом премії Тьюрінга без ступеню доктора філософії, першим із досвідом роботи в галузі техніки, а не науки, й першим, чия кар'єра була повністю пов'язана з промисловістю, а не з наукою або науковим середовищем. Він також перший, хто отримав цю премію за роботу з базами даних.

Система IMS. 1965 року компанія IBM отримала замовлення на створення автоматизованої системи для обліку величезної кількості виробів, деталей і матеріалів, що мали використовуватися під час виконання космічної програми НАСА

«Аполон» - польоту людини на Місяць. Ця система попередньо отримала назву Information Control System – IMS (система управління інформацією). Відповідно до [1] в основу IMS було покладено модель даних, розроблену в середині 60-х років компанією North American Rockwell. 1968 року IMS була надана замовнику, а вже 1969 року стала доступною у сфері інформаційних технологій [17 - 19]. Відтоді й практично по сьогодні компанія IBM вдосконалює IMS, переносить на різні платформи і в різні операційні системи, розширює функціональні можливості. Це, власне, була перша успішна спроба створення промислового варіанту СУБД, хоча на той час так не називалася. Головним архітектором IMS був Верн Уоттс (Vern Watts). Він очолював цю роботу з моменту її проєктування аж до своєї смерті 2009 року.

Верн Уоттс

IMS підтримує ієрархічну модель даних. Вона складається зі схеми й екземплярів. На схемному рівні основним будівельним блоком є сегмент, який складається із сукупності полів. Сегменти зв'язуються направленими бінарними зв'язками. Сегмент, із якого виходить зв'язок називається батьківським, а який приєднується – дочірнім. Кожен сегмент може мати не більше одного батьківського сегменту й велику кількість дочірніх. Сегмент без батька називається кореневим, а без дочірніх сегментів – листям. На рівні екземплярів зв'язок між сегментами означає, що один екземпляр батьківського сегменту зв'язується з багатьма екземплярами дочірнього сегменту. Екземпляр ієрархічної структури містить один екземпляр кореневого сегменту. Отже, ієрархічна модель природно представляє зв'язки «один до багатьох». Слід зазначити, що жорстка формальна специфікація ієрархічної моделі даних відсутня, і вона, зазвичай висвітлюється так, як це було визначено в IMS.

Система Total. 1968 року Томас Нic (Tomas Nies), Клод Богардус (Cloud Bogardus) і Том Річлі (Tom Richley) заснували компанію Cincom Systems, а вже 1969-го було випущено першу версію СУБД Total [20].

Томас Нic

На думку багатьох користувачів і спеціалістів, система Total була серйозним конкурентом IMS на комп'ютерах IBM. На відміну від IMS і більшості інших СУБД того часу Total не обмежувалась одним типом комп'ютерів. Порівняно з IMS вправлятися з Total було доволі легко і ефективно.

Базовою структурою даних Total є дворівнева ієрархія, що містить один запис – власника (master) і велику кількість записів – членів (details). Ці типи записів можуть бути пов′язані так, щоби створювати складні структури даних. Ця структура нагадувала мережеву структуру перших версій IDS.

У Total підтримувалося узгодження із Cobol, Fortran, PL/1 і Assembler. Мова

маніпулювання нагадувала специфікацію Codasyl. Було реалізовано механізм захисту бази даних, який включав динамічну реєстрацію, періодичне резервне копіювання (дамп) і рестарт, запобігання одночасному оновленню даних. Підтримувався режим одночасної роботи багатьох прикладних програм. Було також реалізовано механізм незалежності на рівні окремих елементів даних. Для кожної програми можна було виділити доступну підчисельність бази даних з допомогою механізму, подібному підсхемам.

На початок 70-х років Total мала найбільшу кількість користувачів серед усіх діючих тоді СУБД. Вважається, що на початковому етапі компанія Cincom Systems зробила суттєвий внесок у розвиток СУБД.

Система Adabas. Adabas (database system – адаптивна система баз даних) – система управління базами даних компанії Software AG, Німеччина.

Уперше випущена для мейнфреймів IBM 1971 року. Початкова модель даних – на базі інвертованого індексу. Підхід Adabas відмінний від мережевої моделі даних, однак забезпечує можливість підтримки повної мережевої структури за рахунок неявних відносин. На момент створення мова маніпулювання Adabas являла собою розширення мов Cobol і PL/1. У 1980-і роки доповнена елементами реляційної моделі. На злеті популярності реляційних СУБД в середині 80-х років, вона була однією з найбільш затребуваних систем управління базами даних. IDS, IMS, Total, Adabas належать до складу так званих *навігаційних баз даних*. Цей термін був введений Чарльзом Бахманом у його статті [21], присвяченій отриманню премії Тьюринга. Суть цього класу полягає в тому, що записи даних можуть зв′язуватись між собою різними посиланнями, тим самим створюючи складну структуру даних, а мова маніпулювання дозволяє здійснювати довільну навігацію за цими посиланнями для отримання доступу до потрібних записів. Ідея навігаційних систем булла породжена появою магнітних дисків, які, на відміну від магнітних стрічок, перфострічок і перфокарт із послідовним доступом, надавали прямий доступ.

На завершення цього розділу зазначимо, що сам термін «база даних» (database) з′явився на початку 60-х років. На думку Вільяма Олле (T. William Olle) [1] цей термін уперше було введено у вжиток на симпозіумах, організованих компанією System Development Corporation (SDC) у 1963 і 1965 роках, хоча спочатку сприймався у доволі вузькому сенсі. В широкий ужиток термін увійшов лише на початку 1970-х років [22].

Етап 2. Бурхливий розвиток (1970 – 1980)

70-і роки – це роки бурхливого розвитку баз даних , створення основ технології баз даних. Вони ознаменувалися передовсім дослідженнями робочої групи CODASYL по базах даних (CODASYL DBTG), яка специфікувала мережеву модель, мови визначення і маніпулювання даними.В цей період було визначено і вивчено значну кількість моделей даних, включно із семантичними. 1876 року Петер Чен визначив ER – модель. Була специфікована трирівнева архітектура баз даних ANSI/X3/SPARC, яка стала класичною, здійснені дослідження щодо концептуального моделювання предметних галузей.Закладені основи індустріального виробництва СУБД та іншого програмного забезпечення баз даних. Зрештою, були реалізовані численні промислові СУБД, котрі виявилися затребуваними наступні кілька десятків років. 1973 року, як уже згадувалося вище, Чарльз Вільям Бахман був нагороджений найпрестижнішою в галузі інформатики премією Тьюрінга за видатний внесок у технологію баз даних.

До кінця 60-х років наукове співтовариство дійшло висновку, що системи управління базами даних (СУБД) стали центральною ланкою в автоматизованих інформаційних системах. Однак тоді ще не було повного усвідомлення того, що саме являє собою СУБД, які вимоги вони мусять задовольняти, які моделі даних мають підтримувати, яким архітектурним рішенням мають відповідати.

Але вже на початку 70-х років з′явилися перші звіти й статті, де робилися припущення із конкретних систем [23], а також формулювалися вимоги до СУБД [24,25].

Інфологічні й даталогічні моделі. Уже у 60-і роки вчені, котрі працювали в галузі інформаційних систем, зрозуміли, що в комп'ютерній системі мають бути представлені не лише дані, а і їхня семантика. В середині 60-х років шведський вчений Борже Лангефорс (Borje Langefors) ввів поняття інфологічної і дата логічної моделей (infological and datalogical models), які він розвивав упродовж 15 років [26 - 28].

Борже Лангефорс

Даталогічні моделі – це сукупність структурованих і взаємопов'язаних даних і способи оперування ними. Інфологічні моделі – це моделі представлення інформації (тобто – семантики) про дані. Ці терміни використовуються й по сьогодні, хоча з часом з'явився термін «семантична модель» як модель предметної галузі, призначена для представлення семантики предметної галузі на найвищому рівні абстракції. 1999 року Б. Лангефорс отримав престижну премію LEO за видатні досягнення в галузі інформаційних систем Міжнародної асоціації з інформаційних систем. А 2010 року шведська академія з інформаційних систем заснувала премію Б.Лангефорса за кращу докторську дисертацію Швеції в галузі інформатики й інформаційних систем.

Ідеї Лангефорса згодом були розвинуті й адаптовані до технологій баз даних шведським ученим Б. Сундгреном (Bo Sundgren) [29].

Бо Сундгрен

Архітектура баз даних ANSI/X3/SPARC. Із появою СУБД виникло нове поняття – схема даних (опис даних), яке відсутнє у файловій організації даних. Специфікація цієї схеми і маніпулювання даними виконуються вже мовними засобами СУБД – МОД (мова опису даних) і ММД (мова маніпулювання даними). Взаємодія СУБД із прикладною програмою здійснюється за допомогою розробки спеціального інтерфейсного модулю, в якому специфікуються об'єкти бази даних, потрібні цій програмі, а також необхідні операції над цими об'єктами. Зокрема, як це робиться в СУБД Adabas. Прикладна програма звертається до цього модулю через відповідну точку входу і передає йому певні параметри, що уточнюють запит. У відповідь програма отримує потрібні дані. Це так звана однорівнева архітектура. Цей єдиний рівень складає схема бази даних. Наступним кроком до вдосконалення було введення дворівневої архітектури. Суть її полягає в тому, що крім рівня схеми вводиться рівень підсхеми - фрагмента загальної схеми, який створюється для кожного додатку і описує дані, потрібні цьому додаткові. Дворівнева архітектура була прийнята в IMS. І нарешті в ANSI була визначена трирівнева архітектура баз даних, що стала класичною на багато десятиріч і про яку мова піде далі.

У листопаді 1972 року підкомітет SPARC (Standard Planning and Requirements

Committee) комітету X3 (Committee on Computers and Information Processing) Американського Національного Інституту Стандартів (ANSI) створив робочу групу ANSI/X3/SPARC DBMS для дослідження можливостей і розробки рекомендацій по стандартизації СУБД. Спочатку групу очолив Томас Стіл (Tomas B. Steel, Jr), а згодом Діонісіос Цикритзис (Dionysios Tsichritzis).

Діонісіос Цикритзис

Початковим завданням групи було дослідження питання, чи варто взагалі вирішувати проблему стандартизації СУБД. Якщо так, то що саме має бути стандартизоване. В результаті група дійшла висновку, що стандартизації можуть бути піддані лише інтерфейсні складові СУБД [30].

У зв'язку з цим було поставлено завдання визначення множини компонентів, з яких має складатися СУБД, інтерфейси, між якими могли б стати об'єктами стандартизації.В основу виявлення цих компонентів були покладені наступні концептуальні положення. По-перше, існує реальний світ, інформаційна модель якого має знайти своє відображення в базі даних. По-друге, враховуючи конкретні потреби, в свідомості людей відображаються їхні особисті уявлення про те, яким є реальний світ. Зрештою цей реальний світ матеріалізується у вигляді сукупності символів, у текстовому або електронному вигляді. Саме ця триєдність знайшла відображення в запропонованій групою зкомпонентної структури баз даних, що була названа трирівневою архітектурою баз даних ANSI-SPARC, і яка отримала загальне визнання серед розробників СУБД. Ця архітектура є актуальною й досі. Вона передбачає наявність концептуального, зовнішнього і внутрішнього рівнів. Концептуальний рівень призначений для опису концептуальної інформаційної моделі предметної галузі (ПГ). Зовнішній рівень визначає представлену користувачем БД. Це та частина БД, яка відповідає потребам конкретного користувача. Причому ця частина подається в зручному для користувача вигляді. Внутрішній рівень призначений для опису фізичного зберігання БД. Між цими рівнями існують відображення: концептуальний – зовнішній і концептуально – внутрішній. Ця трирівнева архітектура забезпечує необхідні умови для досягнення логічної і фізичної незалежності даних від програм. У свою чергу, дієвість механізмів опису відображень визначає ступінь достатності досягнення вищезгаданих двох видів незалежності. Результати діяльності цієї робочої групи були надані у звітах. 1977 року Томас Стіл отримав «Нагороду за видатні заслуги» (Distinguished Service Award) асоціації ACM.

Пропозиції КОДАСИЛ. Внесок CODASYL у технологію баз даних пов'язують із створенням мережевої моделі даних. 1967 року в КОДАСИЛ (CODASYL - Conference on Data Systems Languages) була створена спеціальна робоча група з питань баз даних (CODASYL Data Base Task Group — DBTG). Одним із першочергових завдань робочої групи було створення засобів управління базами даних для мови Кобол. Згодом ця задача була суттєво розширена і сформульована як розробка концепції, архітектури і мовних специфікацій баз даних загального призначення. 1971 року, усвідомлюючи важливість досліджень зі специфікації мовних засобів баз даних, було створено Комітет КОДАСИЛ із мови опису даних (CODASYL Data Description Language Committee). В результаті діяльності цих двох груп були опубліковані звіти [23, 33, 34], які викликали значний резонанс, були заслужено визнані фахівцями з баз даних і надовго стали зразком специфікації баз даних. У цих звітах, виходячи зі спільних

позицій і у тісному взаємозв'язку, вперше були строго специфіковані:

- мережева модель даних, ідеї якої були закладені Ч. Бахманом у системі IDS, що і отримала назву моделі даних КОДАСИЛ (CODASYL Data Model);
- трирівнева архітектура баз даних, що згодом була прийнята і розвинута в ANSI/X3/SPARC DBMS;
- мови опису даних (МОД) на всіх трьох рівнях (мова схеми, мова підсхеми, мова схеми зберігання);
- включені в МОД такі функції, як функція адміністрування, перевірки достовірності, управління доступом, налаштування, розподілу ресурсів, захисту даних, цілісності даних;
- відображення між схемою і підсхемою, а також схемою і схемою збереження;
- мова маніпулювання даними, призначена для навігації мережевною структурою з метою специфікації необхідного запису для його оновлення, видалення, або ж вставки нового запису.

За результатамироботи комітетів КОДАСИЛ були опубліковані численні матеріали, серед яких відмітимо монографію Вільяма Олле (T. William Olle) [1]. Підкреслимо, що пропозиції КОДАСИЛ були специфіковані для систем із включаючою мовою, тобто вони припускали, що робота з базою даних здійснювалась через мову програмування. Це повністю відповідало прийнятій тоді технології обробки даних і тому сприяло ефективній реалізації в існуючому вичислювальному середовищі.

Вільям Олле

Мережеві СУБД. Відповідно до специфікацій КОДАСИЛ була реалізована низка СУБД, серед яких: IDMS (Integrated Database Management System) компанії Cullinane Database Systems, що стала основною мережевою СУБД для мейнфреймів і найпопулярнішою в 70 – 80 –і роки минулого століття DMS 1100 (UNIVAC), IDS/II (Honeywell), DBMS 10/20 (DEC).

Концептуальне моделювання. У листопаді 1977 року комітет ISO з мов програмування прийняв рішення про створення робочої групи з питань дослідження різних аспектів використання концептуальних схем у системах управління базами даних з метою забезпечення основи для стандартизації в даній галузі. Спочатку цю групу очолив Т.Б.Стіл - молодший, а згодом Д.А.Жардін (D.A.Jardine). Результатом діяльності цієї групи став звіт, випущений 1982 року під редакцією Дж. Грийтусена (Joost J. Van Griethyusen).

Дж. Грийтусен

У звіті описуються роль і зміст концептуальної схеми, а також визначається зв′язок концептуальної схеми з інформаційним моделюванням і семантикою даних. Підкреслюється важливість точного визначення як статистичних, так і динамічних правил у концептуальній схемі:

- це спільна основа однозначного розуміння суті предметної галузі (ПГ) всіма зацікавленими сторонами;
- вона включає лише концептуально релевантні аспекти ПГ;
- це спосіб визначення допустимої еволюції інформаційної бази даних і

дозволеного маніпулювання інформацією про ПГ;

- це базис для інтерпретації зовнішніх і внутрішніх схем;
- це основа відображення зовнішніх схем у внутрішню і навпаки.

Моделі даних. Відповідно [13] термін «модель даних» почав використовуватися на початку 70-х років після публікації фундаментальної роботи Едгара Кодда (E. Codd) [59]. Однак ще в другій половині 60-х років почали з'являтися перші моделі даних. У результаті розвитку технології баз даних було запропоновано чимало засобів і методологій концептуального моделювання. Зокрем, серед них наступні моделі: модель «Об'єктів – ролей» (OPM – Object – Role Model) Екхарда Д.Фолкенберга (Folkenberg, Eckhard D.) [36, 37], яка була розвинута іншими вченими (С. Нейссен, Р.Меерсман, Д.Вермейр, Т.Халпін (Sjir Nijssen, Robert Meersman, Dirk Vermeir, Terry Halpin). ORM передбачає представлення інформаційної моделі у вигляді об'єктів (сутностей), котрі відіграють ту чи іншу роль (представлені у вигляді зв'язків між об'єктами). На відміну від об'єктно-орієнтованого підходу і підходу сутність-зв'язок ORM не передбачає існування атрибутів, вони подаються у вигляді ролей фактів, які разом із правилами моделюються у вигляді природних пропозицій, легких для розуміння і перевірки користувачами.

Екхард Д.Фолкенберг

Модель даних, заснована на бінарних зв'язках. Біля витоків походження моделі бінарних зв'язків (BR - Binary Relations) були праці таких авторів, як Абріаль [38] (семантична бінарна модель), Браччі [39], Дурхольц [40]. Суть цього підходу до моделювання в тому, що будь-який «елемент» інформації представляється з допомогою екземплярів бінарних асоціацій, тобто висловлювань, до складу яких входять тільки два терми. Зокрема, М.Сенко в межах проекту DIAM (Data Independence Access Method) визначив бінарну мережеву модель, розробив на базі цієї моделі мову FORAL і дослідив можливості користувальницького інтерфейсу, який на ній базується [41, 42, 43].

Семантичні моделі. Відзначимо роботи Дж. Сміта і Д. Сміта по моделях абстракції, агрегації і узагальнення даних [44, 45], а також семантичну модель даних SDM Хамера і МакЛеода [46]. У статті [47] наводиться перелік близько 20 семантичних моделей баз даних. Результатом розвитку моделей даних до початку 80-х років наведені в широко відомій монографії Д.Цикритзиса і Фреда Лоховскі (F.Lochovsky) [2].

Фред Лоховскі

ER – модель. Водночас, найбільшу популярність заслужено здобув підхід сутність – атрибут - зв'язок, названий як підхід сутність - зв'язок (ER – підхід). Свій початок він бере від діаграм структур даних Бахмана [48], а також моделі Інглеса [49].

Вперше найширше цю модель описав П.П.Чен (Peter Pin- Shen Chen) [50].

Петер Чен

ER – модель даних стала загальновизнаною в світі і є основою багатьох методик системного аналізу, концептуального моделювання й проєктування баз даних. Вона базується на простій ідеї, що структурна складова концептуальної моделі предметної галузі може бути представлена у вигляді сутностей, атрибутів і зв′язків. Сутність – це будь-який реальний або абстрактний об′єкт довільної природи, який представляє самостійний інтерес. Атрибут – це властивість сутності, що сприяє якісному або кількісному її опису, ідентифікації, класифікації або відображенню її стану. Нарешті зв′язок – це певна асоціація між різними сутностями (класами сутностей), що становить певний інтерес.

Після публікації статті Чена з′явилося чимало статей, присвячених дослідженню різних аспектів ER - моделювання предметних галузей. Наприклад, в загальному випадку припускається існування n – арних зв′язків, а Річард Баркер (Barker Richard) запропонував ER модель тільки з бінарними зв′язками [51], яка має певні переваги.

У зв′язку з широким використанням ER- моделі [3] було запропоновано багато різних її розширень і узагальнень [52 - 55], які зрештою привели до визначення ієрархічної ER- моделі (ER- моделі вищого порядку) [55]. У статті [56] ER-модель розширена елементами семантизації даних. Також була запропонована темпорально-розширена ER- модель [57], яка дає можливість включати темпоральну інформацію в концептуальну інформаційну модель і представляти її в реляційній моделі. Для підтримки темпоральних запитів мова SQL була розширена можливостями визначення, пошуку й управління історичними відносинами. Із часом було запроваджено ще кілька темпоральних er – моделей, огляд яких наведено в [58]. Зрештою, існує просторова ER - модель (див.: «Просторові бази даних»).

Етап 3. Епоха реляційних баз даних (1970 - 1990+)

На початку 80-х років з'явилися перші промислові реляційні СУБД, які до кінця 80-х стрімко завоювали ринок і стали панівними практично на всіх поширених апаратно-програмних платформах і не втратили свою перевагу й по сьогодні. Попри це, основи реляційної моделі даних і реляційних СУБД були закладені в попередньому десятиріччі, родоначальником їх був Едгар Франк Кодд. Він визначив реляційну структуру даних, алгебру і обчислення, заклав основи теорії залежностей і нормальних форм, сформулював вимоги реляційності баз даних. Ці та інші дослідження кінець кінцем привели до створення теорії реляційних баз даних. Бази даних перетворилися з описової науки у формальну.

1981 року Едгара Франка Кодда було нагороджено премією Тьюрінга за фундаментальний і тривалий внесок в теорію і практику систем управління базами даних, особливо реляційного типу. Було відкрито багато проєктів із дослідження і створення експериментальних СУБД, запропоновано велику кількість мов запитів реляційних баз даних, вивчено питання оптимізації виконання запитів, структури зберігання, методів доступу, захисту, збереження цілісності.

1986 року з'явився перший стандарт SQL і відтоді він став єдиною офіційною мовою зовнішнього інтерфейсу реляційних СУБД.

Було проведено численні дослідження з управління транзакціями, за які

1998 року Джеймс Ніколас Грей отримав премію Тьюрінга.

Реляційні бази даних. У 70-х роках учені, які працювали у сфері баз даних, були переконані, що майбутнє баз даних – за створенням усе складніших структур даних, які дозволяли б адекватно представляти інформаційну модель даних довільних предметни х галузей. Висловлювалися думки, що найближчим часом структури будуть настільки складними, що в базах даних співвідношення корисної інформації та тієї, що її підтримує, буде 1 : 30.

Внесок Е.Ф.Кодда в реляційні бази даних. І ось на цьому тлі 1970 року публікується стаття [59] маловідомого на той час британського вченого Едгара Франка Кодда (Edgar Frank Kodd) з компанії IBM, в якій він запропонував простішу структуру даних. Ця структура являла собою одномірну, плоску, нормалізовану таблицю.

Едгар Франк Кодд

Одномірність означає, що існує лише одна, горизонтальна шапка і не може бути вертикальної, як, скажімо, в навчальних планах ВУЗу. Плоскість свідчить про те, що в шапці не має бути полів, що складаються з багатьох підполів. Приміром, аби поле ПІБ складалося з підполів Прізвище, Ім′я, по-Батькові. І, нарешті, нормалізованість свідчить про те, що в осередках таблиці може бути тільки атомарне (єдине) значення. Така структура дістала назву реляційного відношення, бо вона нагадує математичне поняття відношення. Також було домовлено вважати, що такі відносини існують у першій нормальній формі (First Normal Form – 1NF). У цій же праці він обґрунтував існування двох сімейств реляційних мов, які пізніше були названі реляційним обчисленням і реляційною алгеброю.

1871 року Кодд публікує статтю [60], в якій наводить приклад того, як логіка обчислення предикатів може бути використана для створення високорівневої мови реляційної бази даних.

Описана ним мова ALPHA була першою мовою класу реляційного обчислення. Хоча ALPHA не була реалізована, однак вона мала значний вплив на створення наступних комерційних реляційних мов.

1972 року Кодд публікує наступну свою важливу статтю [61], де він:

- дає формальне визначення реляційної алгебри і реляційного обчислення (кортежно - орієнтованого);
- формулює тезу реляційної повноти селективних можливостей мов запитів до реляційної бази даних на основі реляційного обчислення. Ця теза була одностайно сприйнята вченими світу і в подальшому всі створювані мови запитів перевірялися на реляційну повноту;
- наводить алгоритм редукції довільного вираження реляційного обчислення в семантично еквівалентне вираження реляційної алгебри, тим самим встановлюючи її реляційну повноту. Цей результат згодом було названо теорією Кодда. Пізніше – Палермо (Palermo) [62] удосконалив цей алгоритм з точки зору підвищення його ефективності.

Реляційна модель відпочатку критикувалася за простоту її структури. Це, зокрема, відбулося й на конференції 1974 року «SIGMOD Workshop on Data Description, Access and Control», де виникли дебати між прихильниками реляційного і мережевого підходів, головними спікерами яких виступили Кодд і Бахман. Позиція Кодда на цих дебатах відображена у статті [63]. Зрештою, реляційна модель здобула загальне визнання. Це можна пояснити тим, що в ній вдалося сформулювати мови високого рівня (алгебра,

обчислення). А це дозволило найбільш повно вирішити ту основну проблему, яка була поставлена перед базами даних, а саме – досягнення незалежності даних від програм. У свою чергу, підвищення складності структури даних призводить до неминучого зниження рівня мови маніпулювання, що знижує можливості досягнення такої незалежності.

Намагаючись надати додаткові можливості, у праці [64] Е. Кодд запропонував підвищити семантику реляційної моделі, ідеї якої використовуються й нині в комерційних реляційних СУБД.

Теорія залежності і нормальних форм. Реляційна модель дала серйозний поштовх для розвитку проєктування баз даних. Уперше задача логічного проєктування БД дістала строго формальний підхід. Суть цієї теорії полягала в тому, щоб на основі аналізу різних видів залежностей (обмежень цілісності), які існують всередині реляційних відносин і між ними, виявляти небажані ситуації та усувати їх за допомогою обгрунтовани х процедур еквівалентних перетворень. Зазвичай такою процедурою є декомпозиція відносин, тобто розмежування відносин на декілька. Основоположником цієї теорії став Е.Ф.Кодд, опублікувавши праці [65 - 67]. В них він визначив поняття функціональної залежності (Functional Dependency – FD) в реляційному відношенні, сформулював так звані аномалії маніпулювання відношеннями, виявив два небажані різновиди FD і транзитивні FD, котрі породжують ці аномалії. А саме – неповні FDi транзитивні FD, і запропонував процедуру декомпозиції, що усуває ці різновиди FD у результуючих відношеннях. Відношення, де відсутні неповні FD, отримали назву відношень у другій нормальній формі (2NF), а там, де відсутні неповні й транзитивні FD – у третій нормальній формі (3NF).

1981 року Кодд був нагороджений премією Тьюрінга за фундаментальний і тривалий внесок в теорію і практику систем управління базами даних, особливо реляційного типу. Кристофер Дейт написав книгу[68] – історичний огляд наукового внеску Кодда в реляційну технологію.

З точки зору структури функціональних залежностей 3NF все ж мала певні аномалії. З огляду на це, 1974 року Кодд разом із Раймондом Бойсом (Raymond F. Boyce) запропонували підсилити 3NF. Результована нормальна форма дістала назву нормальної форми Бойса – Кодда (Boyce – Kodd Normal Form – BCNF) [69].

Раймонд Бойс

Як слушно зазначав Дейт, спочатку цю нормальну форму визначив Ян Хіт (Ian Heath) у статті [70]. Також зазначимо, що в цій статті він довів теорему про декомпозицію без втрат реляційного відношення за наявності FD, тобто декомпозиції, яка є еквівалентною за даними. Цю теорему було названо його ім′ям (Теорема Хіта). Вона застосовується при проведенні відношень в 2NF, 3NF і BCNF.

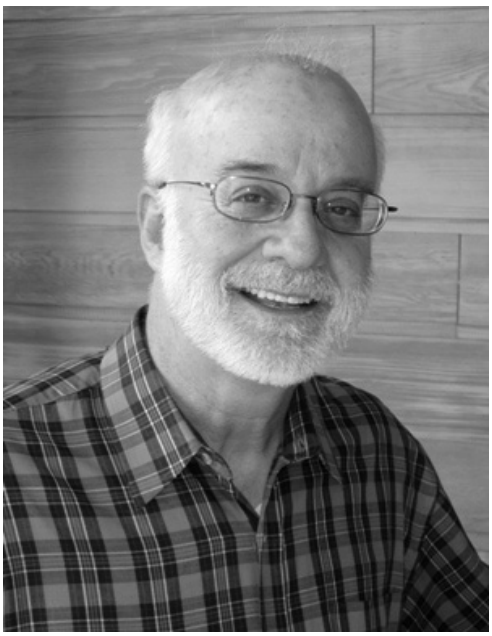
Ян Хіт

1974 року Вільям Армстронг (William Ward Armstrong) у статті [71] запропонував систему аксіом FD (мінімально повний набір правил виводу нових FD із заданих). Вони дістали назву аксіом Армстронга. Ці аксіоми дозволили визначити і дослідити такі поняття з системи FD, як виводимість, повнота, замикання, (мінімальне) покриття, еквівалентність. Отримані в цьому напрямку результати сприяли вирішенню задачі автоматизації проєктування баз даних.

Вільям Армстронг

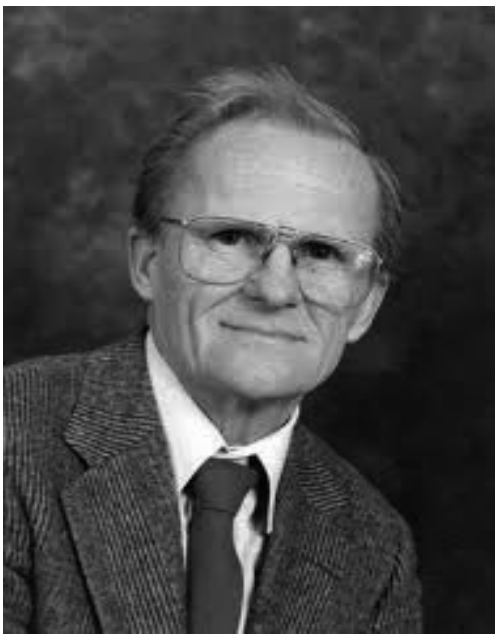
1977 року Рональд Феджин (Ronald Fagin) у статті [72] визначив новий вид залежності – багатозначну залежність (multivalued dependency MVD), наявність якої у відношеннях викликає аномалії маніпулювання. Запропонована ним форма, що усуває цю ситуацію, була названа четвертною нормальною формою (Forth Normal Form – 4NF), а алгоритм приведення в 4NF базувався на доведеній ним теоремі (теорема Феджина). В наступній статті [73] була запропонована повна система аксіом MVD, а також дві аксіоми, які пов′язують FD і MVD (виведення MVD із FD і навпаки).

Зазначимо, що окрім Феджина багатозначну залежність досліджував також Заніоло [74]. Крім того, Делобел [75] визначив поняття «ієрархічної декомпозиції першого порядку», яке також пов′язане з концепцією багатозначної залежності.



Рональд Феджин

1978 року Йорма Ріссанен (Jorma Rissanen) визначив залежність за з′єднанням (join dependency – JD) [76], яка стала узагальненням MVD (MVD є бінарною JD). На її основі Феджин у статті [77] визначив і дослідив проєкційно-з′єднувальну нормальну форму (Projection-Join Normal Form – PJ/NF), яка з часом дістала назву п′ятої нормальної форми (Fifth Normal Form – 5NF).



Йорма Ріссанен

Зрештою, Кристофер Дейт (Christopher Date) визначив шосту нормальну форму (Sixth Normal Form – 6NF) як форму, де відсутні нетривіальні залежності із з′єднання. Як підкреслюють численні дослідники, ця нормальна форма виявилася корисною в темпоральних базах даних. За ствердженням Дейта [78], 6NF рівносильна доменно-ключовій нормальній формі (DK/NF) Феджина (див. далі).

Кристофер Дейт

Наведені вище залежності і нормальні форми належать до так званих класичних. Наведемо ще кілька визначених і досліджених видів залежностей. Із їх докладним аналізом та структурою взаємозв′язків між ними можна ознайомитися в роботі [79]:

- поліпшена 3NF (Improved 3NF) [80];
- нормальна форма елементарного ключа (Elementary Key Normal Form – EKNF) [81];
- нормальна форма суперключа (Super Key Normal Form – SKNF) [82];
- приведена форма 5NF (reduced 5NF – 5NFR) [83];
- нормальна форма без надлишковості (Redundancy Free Normal Form – RFNF) [84];
- нормальна форма із суттєвими кортежами (Essential Tuple Norm Form – ETNF) [85];
- доменно-ключова нормальна форма Феджина (Domain – Key Normal Form – DK/NF) [86];
- ієрархічна залежність [87] та її зв′язок із ієрархічною структурою даних [88];
- залежність по включенню і нормальні форми по включенню (Inclusion Normal Forms) [90 - 92].

Насамкінець зазначимо, що ми навели лише незначну кількість досліджених залежностей. У книзі [93] наведений перелік понад 600 статей, присвячених теорії залежностей і нормальних форм, а в монографії [94] аналізується близько 90 залежностей.

Мова запитів реляційної моделі. Реляційна модель дала суттєвий поштовх дослідженням зі створення мов запитів. У своєму огляді [95] Дональд Чемберлен (Donald D. Chemberlin) запропонував наступну класифікацію мов реляційних баз даних: мови реляційної алгебри, мови реляційних обчислень, графічні мови і мови, орієнтовані на відображення. Даємо короткий огляд мов цих класів.

Дональд Чемберлен

Мови реляційної алгебри. Були запропоновані й експериментально опробовані наступні мови/системи, що базуються на реляційній алгебрі: система VACAIMS [96] розроблена в MIT, системи IS/1 PRTV (Peterlee Relational Test Vehicle) [98], розроблені в науковому центрі IBM в Пітерлі (Англія), система RDMS [99] створена в дослідницькій лабораторії General Motors. В багатьох системах розширюється набір операцій реляційної

алгебри шляхом введення специфічних. Паралельно проводились дослідження з оптимізації виконання виражень реляційної алгебри [101 - 104]. В [105] наводиться широкий огляд досліджень з аналізу складності операцій та оптимізації записів у реляційних базах даних.

Мови реляційного обчислення. Як ми вже зазначали вище, першою мовою запитів реляційної моделі була мова ALPHA Кодда [60], яка безпосередньо базується на реляційному обчисленні. ALPHA дозволяє користувачеві, застосовуючи такі поняття, як змінюваність і квантори, формувати непроцедурні запити. Згодом були запропоновані інші мови, що, як і ALPHA, базувалися на реляційному обчисленні. До них належать QUEL [106], створена в рамках науково-дослідницького проєкту Ingres в Каліфорнійському університеті в Берклі, COLARD (Calculus Oriented Language for Rational Data) [107], RIL [108].

Графічні мови. В мовах, котрі відносяться до графічних, формулювання запитів відбувається не з використанням традиційного лінійного синтаксису, а із заповненням осередків (ячеєк) у бланках таблиць.Мова CUPID (Casual User Pictorial Interface Design – робота з графічним інтерфейсом непрофесійного користувача) [109 - 112] надає користувачеві графічну мову запитів. CUPID має високорівневу меню – образну підмову, яка є зовнішнім інтерфейсом до системи INGRES.

Ідея мови QBE (Query by Example – запит за зразком) [113 - 117], розроблена Моше М. Злуфом (Moshe M/ Zloof), полягає в наступному. Користувачеві надаються чисті бланки таблиць бази даних. Формулювання запиту – це заповнення бланків однією правильною відповіддю, а задача системи – на підставі цього прикладу – вивести всі можливі правильні рядки таблиць. Попри очевидну простоту, було доведено [113], що QBE є реляційно повною мовою. Різновиди цієї мови були реалізовані в СУБД PARADOX, DBASE IV, ACCESS. Остання входить до складу Microsoft Office. М.М.Злуф розробив також мову OBE (Office-by-Example – офіс за зразком) [118], яка стала розширенням QBE для офісних пропозицій.

Моше М. Злуф

Мови, орієнтовані на відображення. !973 року колеги Кодда з лабораторії IBM у Сан Хосе Раймонд Бойс, Дональд Чемберлін і Вільям Кінг (William F. King) розробили мову SQUARE (Specifying Queries As Relational Expressions - специфікація запитів у вигляді реляційних виразів) [119, 120].

Використання SQUARE – подібної мови для опису численних уявлень (поглядів), а також керування цілісністю даних та їх автоматизацією описане в статті [121]. На відміну від реляційного обчислення, SQUARE не використовує кванторів та зв′язаних змінних і тому не потребує відповідної математичної підготовки. В мові запити висловлюються у вигляді природніх примітивних операцій, якими люди користуються під час пошуку інформації в таблицях. Більшість семантичних простих запитів відображаються в мові просто й лаконічно. Разом з тим SQUARE є реляційно повною мовою [120].

У 1974 році Бойс і Чемберлін представили мову SEQUEL (Structured English QUEry Language – структурована англійська мова запитів) [122], яка стала удосконаленим варіантом мови SQUARE. Змішаний синтаксис мови було названо

блочно – структурованим синтаксисом ключових слів англійської мови. SQUARE і SEQUEL були декларативними мовами. Тобто, в них формулюється «що» треба знайти, а не «як» це зробити, характерне для процедурних мов. 1975 року було реалізовано експерементальний варіант SEQUEL на базі розробленого інтерпретатора [123]. Завдання інтерпретатора – мінімізувати виконання операцій доступу до даних під час виконання запитів за рахунок звуження простору пошуку. Задля цього були досліджені спеціальні оптимізуючи алгоритми. Сам інтерпретатор SEQUEL базується на XRM (Extended n-ary Relational Memory [124, 125]) – системі, яка надає ефективний асоціативний доступ до бінарних відношень. Нарешті 1976 року було представлено мову SEQUEL2 [128], в якій уже були включені всі основні засоби для оперування базами даних: визначення, маніпулювання і керування.

Оригінальний підхід був запропонований у мові APPLE (Access Path Producing Language – мова, яка породжує шлях доступу) [129], одним з авторів якого був Карл Роберт Карлсон (Carl Robert Carlson). Мова передбачає використання в запиті тільки імена атрибутів відношень бази даних. Задача системи – на основі структури бази даних визначити численні відношення, необхідні для виконання запиту, і визначити шлях доступу до них.

Карл Роберт Карлсон

Експерементальні дослідження і розробки. Вже на початку 70-х років було реалізовано низку ранніх реляційних систем - MacAIMS (1970 р.), IS/1 (1972 р.) и PRTV, RENDEZVOUS (1974 р.) тощо.

IS/1i PRTV. IS/1 була першою в світі експериментальною реляційною системою баз даних з обмеженими можливостями, реалізованою в науковому центрі IBM у Пітерлі, Великобританія в 1970 – 1972 роках [130]. Із урахуванням результатів, отриманих під час реалізації IS/1, була розроблена СУБД PRTV (Peterlee Relational Test Vehicle) [131], що дозволяла оперувати великими обсягами даних, мала свою власну мову записів ISBL рівня реляційної алгебри і була призначена для одного користувача.

System/R і DB2. 1974 року в дослідницькій лабораторії в Сан Хосе компанії IBM був ініційований проєкт System/R зі створення експериментальної СУБД. Його задачею було продемонструвати можливість створення високопродуктивних промислових реляційних СУБД. За основу було взято мову SEQUEL, яка в процесі обробки була перейменована на SQL, виходячи з юридичних міркувань. До 1975 року було реалізовано повнофункціональну версію System/R для багатьох користувачів [132]. Зрештою, протягом 1978-79 років System/R витримала всебічну практичну апробацію [133, 134], результати якої продемонстрували, що реляційні СУБД здатні забезпечити високу продуктивність. 1979 року проект System/R було завершено. Пізніше коротка історія експериментальних досліджень проекту System/R була викладена Чемберліном і його колегами в статті [135]. Враховуючи отриманий досвід, компанія IBM 1980 року почала, а 1982 року випустила промислову реляційну СУБД під назвою SQL/DS, яка згодом була перейменована на DB2 і підтримується на сьогодні на різних платформах і в різних конфігураціях. Вона стала стратегічним програмним продуктом компанії IBM.

Oracle. 1974 року троє молодих програмістів із американської електронної компанії Ampex Corporation Ларрі Еллісон (Larry Ellison), Боб Майнер (Bob Miner) та Ед Оутс (Ed Oats), окрилені ідеями Кодда, заснували компанію Software Development Laboratories (SDL) для створення реляційного СУБД і взя-

лися за розробку і маркетинг програми. 1979 року компанія дістала нову назву – Relational Software Inc. Того ж року компанія випустила Oracle, першу комерційну реляційну СУБД, де використовувалася мова SQL. Програма дуже швидко набула популярності. 1982 року компанія знову була перейменована на Oracle Systems Corporation. Відтоді Oracle є найбільшим постачальником реляційних СУБД на базі SQL.

Ingress. 1973 року двоє вчених дослідницької лабораторії Каліфорнійського університету в Берклі Майкл Стоунбрейкер (Michael Ralph Stonebraker) і Юджин Вонг (Eugene Wong), зацікавившись дослідженнями Кодда і результатами своїх колег із IBM зі створення System R, вирішили почати власний проект зі створення реляційної СУБД.

Майкл Стоунбрейкер

Юджин Вонг

Розроблювану експериментальну СУБД було названо INGRES (Interactive Graphics and REtrieval System).

За наступні два роки були проведені експериментальні дослідження і розробки. Було прийнято проєктні рішення [136, 137], розроблені структури зберігання і методи доступу [138]. Також було розроблено оптимізаційний алгоритм виконання операцій поєднання відношень, який отримав назву алгоритм Вонга – Юсефі (Wong – Youssefi algorithm) [139], досліджено механізм надання альтернативних поглядів через підстановку в запити користувачів їхніх визначень поглядів [140]. Авторизація і контроль цілісності забезпечувався використанням додаткових предикативів до запиту користувача [141]. Реалізовано механізм безпечного одночасного оновлення бази даних [142], а також система захисту [143]. До 1976 року була реалізована експериментальна версія Ingress [144]Яка підтримувала мову QUEL. 1980 року частина співробітників цієї лабораторії заснували фірму Relational Technology, яка 1981 року випустила промислову СУБД INGRESS. 1986 року INGRESS було переведено на SQL. Кілька ключових ідей з INGRESS досі широко використовуються в реляційних системах. Наприклад, в Non Stop SQL, Sybase і Microsoft SQL Server.

Postgres. Після створення Relational Technology Стоунбрейкер разом із Лоуренсом А. Роу (Lowrence A. Rowe) почали досліджувати можливості усунення обмежень реляційної моделі.

Лоуренс А. Роу

Новий проект дістав назву Postgres (POST inGRES). Були розроблені концептуальні проєктні рішення [145], запропонована об'єктивно – реляційна модель зі складними типами даних [146], розроблені структура збереження даних [147] і система правил [148] (тригерів), яка дозволяє визначати додаткові дії, ініційовані під час виконання операцій вставки, оновлення або видалення в таблицях бази даних.

Попервах мовою запитів Postgres була PostQUEL. Мова була розроблена 1985 року в Каліфорнійському університеті в Берклі під керівництвом М. Стоунбрейкера. PostQUEL базувалася на мові запитів QUEL. 1987 року була реалізована перша версія СУБД Postgres, яка протягом наступних кількох років удосконалювалась [149]. Postgres почала широко використовуватись в економіці, промисловості, медицині, фінансовій справі, астрономії і в багатьох інших галузях. А також використовувалася в навчальному процесі. 1991 року було додано інтерпретатор мови SQL, а 1996 – програмний продукт було перейменовано на PostgreSQL. 2014 року Майкл Стоунбрейкер став лауреатом премії Тьюринга за фундаментальний внесок у концепції і методи, що лежать в основі сучасних систем баз даних [150].

СУБД для ПК. До 1980 року дослідження, експериментальні й промислові розробки СУБД проводилися для великих і середніх комп'ютерів. На початку 80-х з'явилися IBM PC і сумісні з ним ПК, оснащені ОС MS-DOS, що привело до появи СУБД для ПК. 1981 року компанія Ashton-Tale випустила dBase II для ПК. ЇЇ не можна було назвати справжньою СУБД, бо чимало важливих функцій не підтримувались, однак для ПК того часу це було значною подією. Dbase II здобула велику популярність. 1984 року було випущено досконалішу версію dBase III, в 1986 – її розширений варіант dBase III+, а 1998 - dBase IV. Вони стали домінуючими СУБД для IBM PC. Успіх dBase III+ обумовив появу на ринку числених аналогів, сумісних за мовою і структурою файлів бази даних. До них відносяться FoxBASE (1984), FoxPro (1990) компанії Fox Software, Clipper (1985) компанії Nantucker Corporation. Згодом вони були об'єднані популярним серед професіоналів поняттям "xBase". Тенденція створення продуктів – аналогів і велика популярність xBase активізувала діяльність зі створення стандарту. Було зроблено дві спроби стандартизації мови xBase у 1987 – 1988 і 1992 роках, але вони не мали успіху. Тож у 80-х роках домінуючу роль на ринку СУБД для IBM PC мало сімейство СУВД xBase.

1985 року компанія Ansa Software випустила СУБД Paradox. Цей високоефективний продукт для створення реляційних баз даних був примітний своєю мовою QBE (Query By Example) і мовою розробки додатків. Він був популярний у кінці 80-х – початку 90-х років і конкурував із сімейством xBase.

Оптимізація. Реляційні системи базуються на високорівневому непроцедурному інтерфейсі, їхні мови записів декларативні. Через це в таких системах принципово важливим є питання оптимізації виконання запитів. У 70 – 80 роки минулого століття були проведені численні дослідження і опублікована величезна кількість статей із цього питання. Ми не зупиняємося в цій статті на даній проблемі і посилаємо читача до змістовного огляду С.Д.Кузнєцова [151].

Стандартизація. У травні 1979 року була створена робоча група із реляційних баз даних (RTG) ANSI/X3/SPARC DBS-SG під керівництвом Майкла Броуді (Michael L.Brodie) для проведення досліджень з обґрунтування можливості створення стандарту по реляційних базах даних. 1981 року ця група склала звіт [152], де підтверджувалась така необхідність. Для наступного сприяння в роботі зі створення такого стандарту було розроблено «Каталог функцій реляційних концепцій, мов і систем», який мав би допомогти виявити і встановити ті аспекти, як самої реляційної моделі, так і реляційних баз даних, які можуть розглядатися кандидатами для стандартизації. До початку 80-х років у зв'язку із широким розповсюдженням реляційних СУБД виникла необхідність аналізу можливості стандартизації мови для управління

реляційними базами даних і розробки такого стандарту, якщо це буде визнано доцільним.

Майкл Броуді

У зв'язку з цим 1982 року Американський національний інститут стандартів (American National Standards Institute – ANSI) створив комітет X3H2, перед яким було поставлене це завдання. Впродовж 11 років комітет очолював Дональд Р. Дойч (Donald R.Deutsch). Комітет мав розглянути різні реляційні мови, описані і реалізовані на той час. Однак, враховуючи широку розповсюдженість SQL у промислових СУБД і той факт, що тоді він фактично вже став стандартом, комітет зупинив свій вибір саме на цій мові. Узявши за основу її діалект, реалізований в СУБД DB2, комітет прагнув його узагальнити, враховуючи реалізовані в інших реляційних СУБД можливості. Після чотирьох років роботи, 1986 року запропонований комітетом варіант SQL, було офіційно затверджено як стандарт ANSI, а 1987 року він був узятий за стандарт Міжнародної організації стандартів (International Standards Organization – ISO). Згодом стандарт ANSI/ISO взяв уряд США за федеральний стандарт в галузі обробки інформації (Federal Information Processing Standard – FIPS). 1989 року стандарт був дещо змінений і дістав назву SQL – 89 (або SQL1).

Відтоді SQL було визнано єдиною мовою зовнішніх інтерфейсів реляційних баз даних. ANSI/ISO постійно працює над її удосконаленням і випуском нових версій. За 35 років було випущено 10 версій SQL (1986, 1989, 1999, 2003, 2006, 2008, 2011, 2016, 2019).

Дональд Р. Дойч

На завершення цього розділу зазначимо, що до початку 80-х років на ринку з'явилися дві промислові реляційні СУБД: Oracle і DB2. Згодом з'явилися Postgress, Informix тощо. Почалася ера реляційних СУБД, які й по сьогодні є найбільш популярними на ринку баз даних.

Управління транзакціями.

Важливою функцією використання БД є управління транзакціями. Транзакція (Transaction) – це логічна одиниця роботи, що являє собою групу послідовних операцій над даними БД, яка може бути або використана повністю й успішно, дотримуючись цілісності даних і незалежно від інших паралельно працюючих транзакцій, або не використана зовсім. Тоді вона не матиме ніякого ефекту. Поняття транзакції вперше було введене й обговорене Джимом Греєм (Jim Grey) і його колегами в працях [153, 155, 158].

Правила ACID. Одним із найпоширеніших наборів вимог до транзакцій і транзакційних систем є набір ACID (Atomicity, Consistency, Isolation, Durability).

- *Atomicity – атомарність.* Транзакція або виконується повністю, або не виконується взагалі. З точки зору зовніш-

нього сприйняття, вона не має ніяких проміжних станів.

- *Consistency – узгодженість.* Транзакція зберігає обмеження цілісності бази даних. По завершенні роботи транзакція залишає базу даних у цілісному стані.
- *Isolation – ізольованість.* Транзакція працює так, ніби не було ніяких одночасно працюючих транзакцій.
- *Durability –довготривалість.* По закінченні роботи всі зроблені транзакцією зміни зберігаються в базі даних на довготривалій основі.

Вимоги ACID були сформульовані переважно на початку 80-х років Джимом Греєм [153]. Водночас існують спеціальні системи з послабленими транзакцій ними властивостями [154].

Ізольованість транзакції – ситуація, в якій транзакція захищена (за ізольована) від дії інших одночасно з нею виконуваних транзакцій. Іншими словами, ізоляція гарантує, що проміжні стани транзакції є невидимими іншим одночасно працюючим транзакціям. Ступінь ізоляції транзакції визначається рівнями ізоляції. Аби досягти ізольованості транзакції, слід використовувати методи управління спільним виконанням транзакцій. План виконання набору транзакцій називається серіальним у разі, якщо результат спільного виконання транзакцій еквівалентний результату певного послідовного виконання цих таки транзакцій.

Серіалізація. Серіалізація транзакцій – це механізм такого спільного виконання транзакцій, коли результат еквівалентний результату певного послідовного виконання цих транзакцій. Забезпечення такого механізму є основною функцією управління транзакціями. Система, в якій підтримується серіалізація транзакцій, забезпечує реальну ізольованість користувачів.

Концепція серіалізації була сформульована і досліджена Греєм і його колегами в працях [155, 158]. Окрім цього, в праці [6] було визначено двофазний протокол блокування, а також досліджена техніка предикатного блокування. Питання межі дії блокування (гранулярності) обговорені в статтях [155, 156].

Моделі транзакцій. Варто зазначити, що всі моделі транзакцій визначалися, як правило, з урахуванням класів прикладних систем, де вони застосовуються [11]. Було запропоновано дві фундаментальні моделі транзакцій – модель сторінки і модель об′єкту. Перша з них – виконавча модель, а друга – концептуальна.

Модель сторінок (модель Read/Write). Базується на припущенні, що основні операції бази даних – це запис і читання сторінки, котрі передаються між зовнішньою і оперативною пам′яттю. Сторінкова модель транзакції бере свій початок у другій половині 70-х років зі статей Джима Грея [153, 157] і Капалі Есваран (Kapali Eswaran) [158]. Одночасно виникло споріднене поняття – атомарна дія [159, 160]. Концепція сторінкової моделі транзакції стала предметом інтенсивних теоретичних досліджень 80-х років [161 - 164] і дієва дотепер, хоча й набула кількох розширень і варіацій [165]. Із оглядом досліджень і розробок цієї моделі можна ознайомитися в працях [11, 166].

Усі наведені далі моделі належать до класу так званих моделей об′єктів.

Плоскі транзакції (Flat Transactions) мають єдиний рівень управляння для довільної кількості елементарних дій. Вони не мають внутрішньої структури. Плоскі транзакції – основні будівельні блоки для реалізації принципу атомарності. В плоских транзакціях атомарність і довготривалість підтримуються механізмом відновлення, який зазвичай забезпечується веденням журналів операцій оновлення, через що операції типу «відмінити», «повторити» можна виконувати за потреби. Ізольованість забезпечується механізмом управління паралелізмом (concurrency control), який реалізується з допомогою блокувань.

Огляд досліджень з управління паралелізмом наведено в праці [167]. Узгодженість забезпечується механізмом управління цілісністю. Було запропоновано два підходи до управління цілісністю в транзакціях: включення цього механізму в СУБД [168] і підтримання цілісності за рахунок зусиль розробників додатків [11].

Плоскі транзакції повністю дотримуються всіх принципів ACID і є цілком достатніми для багатьох традиційних додатків баз даних, у яких час виконання транзакції відносно нетривалий, кількість паралельних транзакцій досить невелика, і база даних не розподілена. Однак такі ACID-транзакції не взмозі підтримувати довгочасні транзакції та транзакції зі складною внутрішньою структурою і розподіленими базами даних.

Точки збереження. Це такі моменти в обчислювальному процесі, починаючи з яких можливий перезапуск обчислень у разі виникнення будь-яких проблем. Вони вперше були визначені 1976 року в SystemR [132]. У разі виникнення збою відбувається відкат до останньої збереженої точки з вивільненням усіх зроблених після цієї точки блокувань. Хоча механізм точок збереження широко використовується в плоских транзакціях, однак він набув нового звучання в розширених моделях транзакцій, які з′явилися у 80-х роках.

Усі наведені далі розширені моделі транзакцій призводять до ослаблення тих чи інших складових ACID.

Модель багатоланкових транзакцій (Chained Transactions) подібна до моделі плоскої транзакції з точки збереження, але вона не лише дає можливість позначити будь-яку точку для можливого повторного виконання, а й фіксацію тієї частини роботи, яка була виконана в момент досягнення цієї точки. Втім відкат може бути виконаний лише до останньої контрольної точки. В цьому підході було закладено ідею декомпозиції великих транзакцій на дрібніші послідовно виконувані субтранзакції, які відповідають інтервалам між точками. У разі збою поточної субтранзакції, попередня вже була зафіксована і її результати були збережені в базі даних, тому відкат проводиться до цієї точки збереження. Зазначимо, що в цій моделі атомарність і ізольованість не гарантуються для всієї транзакції. Згідно [11] ідея багатоланкової транзакції вперше реалізована в системі IMS компанії IBM.

Вкладені транзакції (Nested Transactions). Важливим кроком у розвитку базової моделі транзакції було розширення плоскої (однорівневої) моделі в багаторівневу структуру. Вкладена транзакція вперше була визначена 1981 року Моссом (Moss) [169], а потім [170]. Її концепція базувалася на понятті сфер контролю (spheres of control) [171]. Вкладена транзакція – це численні субтранзакції, що здатні містити інші субтранзакції, таким чином утворюючи транзакційне дерево. Дочірні транзакція запускається після батьківської, а батьківська транзакція завершується лише після завершення роботи всіх її дочірніх транзакцій. У разі аварійного завершення батьківської транзакції, всі її дочірні транакції також завершуються аварійно. У разі аварійного завершення дочірньої транзакції її батько може обрати альтернативний варіант (contingency subtransaction – транзакція на непередбачуваний випадок). Вкладені транзакції забезпечують повну ізоляцію на глобальному рівні. Для вкладених транзакцій послаблена здатність довготривалості ACID.

Модель вкладених транзакцій повністю відповідає активним базам даних, оскільки ієрархічна структура моделі дозволяє легко погоджувати зв′язок між основною трансакцією і запущеною з допомогою тригера. В статті [172] запропоновані наступні варіанти синхронізації запуску субтранзакції тригеру і щодо основної транзакції:

- негайний (immediate) - транзакція запускається одразу після настання події тригера;
- відкладається (deffered) - запуск субтранзакції відкладається до завершення основної транзакції;
- причинно-незалежна (causally independent) – субтранзакція тригера запускається як цілком самостійна транзакція;
- причинно-залежна (causally dependent) – субтранзакція тригера запускається як самостійна транзакція, однак її успішне завершення залежить від успішного завершення основної транзакції.

Відкриті вкладені транзакції (Open Nested Transactions) [173] послаблюють вимогу ізольованості через те, що результати зафіксованих субтранзакцій стають видимими іншим одночасно працю-

ючим вкладеним транзакціям. При цьому досягається високий рівень паралельності.

Багаторівневі транзакції є найзагальнішим варіантом вкладених транзакцій [173, 174]. Субтранзакції багаторівневої транзакції здатні провести фіксацію і звільнення своїх ресурсів до завершення роботи глобальної транзакції. Якщо глобальна транзакція завершується аварійно, то для підтримки атомарності слід зробити відкат субтранзакції запуском їх компенсуючи субтранзакцій. Однак все ж можливе порушення цілісності в зв′язку з тим, що певна інша транзакція мала доступ до результатів завершених субтранзакцій, які потім були відкатані компенсуючими субтранзакціями. Було запропоновано вирішення цієї ситуації. Зокрема, введенням горизонтальних компенсаторів [175]. У монографії [176] наводяться розпізнавальні ознаки багаторівневих і вкладених транзакцій.

Розподільні транзакції (Distributed Transactions). Це сукупність субтранзакцій, прив′язаних до локальних баз даних і загальної глобальної транзакції. У статті [177] подається огляд розподілених транзакцій, а також розглянута «модель базисної транзакції» (Base Transaction Model) і її розширення. 1996 року було запропоновано модель x/Open Distributed Transaction Processing (x/Open DTP) [178]. Ця модель є стандартом для протоколу двофазної фіксації (2PC – Two Phase Commit).

Гнучкі транзакції (Flexible Transactions) [179, 180] – були запропоновані для розподілених баз даних. У цьому випадку глобальна транзакція є набором субтранзакцій, кожна з яких здійснює доступ до даних на одному локальному вузлі. Модель гнучкої транзакції підтримує гнучке управління обчисленням шляхом специфікації залежностей двох типів між субтранзакціями: 1) залежності порядку обчислень між двома субтранзакціями; 2) залежності альтернатив між двома піднаборами субтранзакцій. Було розроблено кілька конкретних моделей гнучких транзакцій: Con Tracts, Flex Transaction, Split Transaction, S-transaction та інші [179 - 183]. Була також запропонована мова IPL [184] для специфікації гнучких транзакцій із атомарністю й ізольованістю, які визначаються користувачем.

Тривалі транзакції, компенсатори і модель Saga. Ідея компенсуючих транзакцій (Compensation Transactions) вперше була висловлена Греєм у праці [157], згодом вона була формалізована в [185, 186] і, зрештою, використана в моделі Saga [187]. Ця модель належить до тривалих транзакцій (Long – Running Transactions) [188]. Моделі розподільних транзакцій вдало справляються з короткочасними транзакціями, однак, водночас є неприйнятними для тривалих транзакцій. В моделі Saga пропонується розподіляти тривалі транзакції на коротші. Saga складається із сукупності впорядкованих ACID субтранзакцій і сукупності компенсуючих субтранзакцій, по одній на кожну з основних субтранзакцій. Координація всього процесу здійснюється за допомогою повідомлень і тимчасових поміток. Saga завершується успішно за умови успішної фіксації всіх субтранзакцій. Якщо ж котрась із субтранзакцій завершується аварійно, то всі попередньо завершені субтранзакції відкочуються виконанням так званих компенсуючи субтранзакцій. Saga послаблює вимоги до ізольованості і збільшує міжтранзакційний паралелізм. У праці [189] запропонована вдосконалена модель – вкладена Saga, яка дозволяє представляти лінійну структуру тривалих транзакцій у вигляді ієрархічної транзакційної структури.

Транзакції Split/Joint (розділити/об′єднати). Концепція роз′єднання/об′єднання транзакцій вперше була описана в [190], а відтак ретельно пропрацьовано в [191] для таких тривалих видів діяльності, як автоматизоване проєктування, інженерне проєктування, проєктування і розробка програмного забезпечення тощо. Операція Split розділяє транзакцію на дві серіалізовані транзакції, які фіксуються або завершуються аварійно незалежно одна від одної. Операція Joint об′єднує дві транзакції в одну. Split використовується, наприклад, щоб якнайшвидше зафіксувати результати роботи частини транзакцій, або щоб розподілити роботу між кількома виконавцями. Зі свого боку Joint рівно-

сильний передачі всієї роботи одному виконавцеві [191]. Згодом операції Split та Joint було включено до вкладених транзакцій для створення комбінованих моделей транзакцій [192].

Кооперативні транзакції були запропоновані в [193] для використання в системах, де яскраво виражена потреба у взаємодії між транзакціями, в кооперативному інтерактивному робочому середовищі. Фундаментальна проблема, пов′язана з кооперативними транзакціями – це відсутність для них чітких критеріїв узгоджуваності. Для кооперативних транзакцій було запропоновано їхню структуризацію у вигляді дерева, названого ієрархією кооперативних транзакцій (Cooperative Transaction Hierarchy). В окремому випадку ієрархія обмежена трьома рівнями: корінь, одна або більше транзакційних груп і кілька кооперативних транзакцій. Кооперативні транзакції утворюють листя ієрархії, яке об′єднується в групи. Члени групи транзакцій працюють разом, виконуючи певну логічну одиницю роботи, названу задачею, котра може бути розбита на підзадачі. Кожна кооперативна транзакція відповідальна за конкретну підзадачу. Оскільки вимога атомарності послаблена, кооперативна транзакція не зобов′язана зберігати глобальне непротиріччя бази даних, тобто зміни, зроблені кооперативною транзакцією, одразу стають видимими іншими кооперативними транзакціями цієї групи. Щоб результати були видимі поза групою, використовуються точки збереження. В ієрархії може існувати більше трьох рівнів, тобто допускається кілька рівнів вкладення груп. Кооперативні транзакції не обов′язково мають бути серіалізованими. Через свою інтерактивну природу, кооперативні транзакції тривають значно довше звичайних. Згодом, на відміну від традиційних транзакцій, кооперативні не обов′язково мають бути повністю ізольованими.

ACTA і її похідні. ACTA [192, 194, 195] – це мета модель, яка полегшує специфікацію, аналіз і синтез розширених моделей транзакції. Формалізм ACTA має за основу логіку першого порядку з відношенням перебування, яке дозволяє розробнику транзакцій специфікувати як високорівневі властивості (вимоги) моделі, так і низькорівневі аспекти поведінки в термінах аксіом. Окрім підтримки специфікації і аналізу існуючих моделей транзакцій, ACTA дає можливість специфікувати вимоги нових транзакцій них додатків та синтезувати моделі, що задовольняють ці вимоги. В роботі [42] автори запропонували спрощений спосіб розробки розширених транзакцій, названий ASSET. Він базується на транзактивних примітивах, запозичених у ASTA і може використовуватися на рівні програмування для специфікації спеціалізованих для конкретних додатків моделей транзакцій, які дозволяють підтримувати кооперацію і взаємодію. У 90-і роки велика увага приділялася транзакціям в системах реального часу [197, 198, 199] і в мобільних системах баз даних [200, 201].

Транзакції веб – сервісів. Із початком 2000-их років усе більша увага приділяється використанню транзакцій для слабопов′язаних веб-сервісів із метою забезпечення погоджуваності й надійності веб-сервісних додатків. Наразі розроблено три стандарти, що мають відношення до транзакцій веб-сервісів.

Business Transaction Protocol (BTP) [202, 203]. Перша версія розроблена 2004 року в OASIS. Вона стосується як веб-сервісів, так і довільних бізнес процесів. Це протокол, що базується на мові XML, для опису і управління складними багатокроковими B2B-транзакціями (business-to-business) в Інтернеті. Він дає можливість координувати транзакції між багатьма автономними сервісами, а використання XML робить його придатним для веб-сервісних архітектур [204].

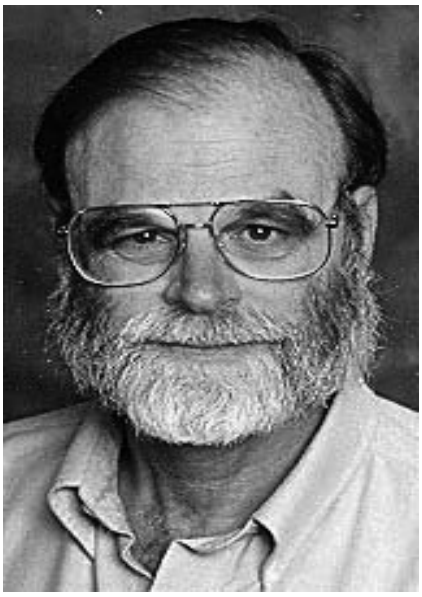
Web Services Transactions (WS – Tx) [205, 206]. Специфікація, схвалена 2007 року, складається з WS – Coordination (WS-C), WS-Atomic Transaction (WS – AT), WS-Business Activity (WS – BA) та розроблена в Microsoft, IBM і BEA. WS – Tx визначає механізми транзакційної інтероперабельності між веб-сервісами та забезпечує впровадження якісних транзакцій них сервісів до веб=сервісних додатків. WS – Tx визначає послідовність

повідомлень, що передаються сторонами – учасниками в короткострокових атомарних транзакціях (WS – AT) і (тривалих) бізнес – транзакціях (WS – BA). WS – C визначає координаційні протоколи повідомлень, якими обмінюються сторони – учасники транзакцій. WS – C підтримує різні координаційні моделі [2007].

WS Composite Application Framework (WS – CAF) [2008]. Стандарт розроблено в OASIS за участі компаній SUN, Oracle, Arijuna та інших. Мета стандарту – розробка інтероперабельних і простих у використанні складових вебсервісних додатків.

Було здійснено порівняльні дослідження наведених вище трьох стандартів, з результатами яких можна ознайомитися в статтях [209, 211].

1998 року Джеймс Ніколас Грей (James Nicholas Gray) був нагороджений премією Тьюрінга за основоположні ідеї в галузі баз даних, за дослідження з обробки транзакцій і технічне лідерство в реалізації систем.



Джеймс Ніколас Грей

(Далі буде)

References

1. Olle T. William. The CODASYL Approach to Data Base Management. Chichester, England: Wiley-Interscience; 1978: 287p.
2. Tsichritzis D.C., Lochovsky F.H., Data models, Prentice-Hall, Englewood Cliffs, N.J., 1982, 381 p.
3. Embley D., Thalheim B., editors. Handbook of conceptual modelling: its usage and its challenges. Springer; Berlin 2011
4. Date C.J. An Introduction to Database Systems, 8th Edition. Addison-Wesley Longman Publishing Co., Inc.75 Arlington Street, Suite 300 Boston, MA United States
5. Gallaire H., Minker J., eds. Logic and Databases. New York: Plenum. 1978.
6. Ullman J.D. Principles of Database and Knowledge-Based Systems.Maryland: Computer Sciences Press Inc., 1989
7. Maier D., Warren D.S. 1988. Computing with Logic: Logic Programming with Prolog. Benjamin-Cummings Publishing Co., Inc.Subs. of Addison-Wesley Longman Publ. Co390 Bridge Pkwy. Redwood City, CA United States. 535 p.
8. Ozkarahan E.A. Database Machines And Database Management. Prentice Hall, 1986, 636 p.
9. Kalinichenko L.A., Ryvkin V.M. Database and Knowledge base Machines (Rus). Moscow, Nauka, 1990, 296 p.
10. Özsu M.T., Valduriez P. Principles of Distributed Database Systems, Fourth Edition, Springer, 2020
11. Gray J., Reuter A. Transaction Processing: Concepts and Techniques. Morgan Kaufmann, San Francisco. 1993
12. Harrison G. Next Generation Databases: NoSQL, NewSQL and Big Data, Apress, 2015, 235 p.
13. Kogalovsky M. R. Encyclopedia of databases technologies (Rus). Moscow, Finance and Statistics,.2005. — 800 c.
14. Bachman Charles W. Integrated Data Store - The Information Processing Machine That We Need! General Electric Computer Users Symposium. Kiamesha Lake. New York May 17-18, 1962
15. IDS Reference Manual GE 625/635, GE Inform. Sys. Div., Pheonix, Ariz., CPB 1093B, Feb. 1968.
16. Bachman Charles W. “The Origin of the Integrated Data Store (IDS): The First Direct-Access DBMS,” IEEE Annals of the History of Computing, Vol. 31, Num. 4, Oct-Dec 2009, pp. 42-54.

17. History of IMS: Beginnings at NASA. - https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.imsintro.doc.intro/ip0ind0011003710.htm#ip0ind0011003710
18. Long R., Harrington M., Hain R., Nicholls G. IMS Primer. - http://www.redbooks.ibm.com/redbooks/pdfs/sg245352.pdf
19. Information Management System/360, Application Description Manual H20-0524-1. IBM Corp., White plains, N.Y., July 1968.
20. Nies T. Cincom Systems' Total. Annals of the History of Computing, IEEE. 2009, vol. 31, No 4, pp. 55-61.
21. Bachman Charles W. "The programmer as navigator". Communications of the ACM, November 1973, Vol. 16 No. 11, Pages 653-658 https://dl.acm.org/doi/10.1145/355611.362534
22. Haigh T. How Data Got its Base: Information Storage Software in the 1950s and 1960s // IEEE Annals of the History of Computing (Volume: 31, Issue: 4, Oct.-Dec. 2009) pp. 6-25
23. CODASYL: "Data Base Task Group Report", ACM (New York 1971).
24. GUIDE-SHARE: "Data Base Management System Requirements", SHARE Inc. (New York 1970)
25. CMSAG Joint Utilities Project: "Data Management System Requirements", CMSAG (Orlando, FL 1971)
26. Langefors B. Theoretical Analysis of Information Systems. 402 S. m. Fig. Lund/Kopenhagen/Oslo 1966. Akademisk Forlag/Universitetsforlaget
27. Langefors B. Information systems theory. Inf. Syst. 2(4): 207-219 (1977)
28. Langefors B. Infological models and information user views. Information Systems Volume 5, Issue 1, 1980, Pages 17-32
29. Sungren Bo. An Infological Approach to Data Bases. National Central Bureau of Statistics, Sweden, Stokholm, 1973. 294 p.
30. SPARC: "Outline for Preparation of Proposals for Standardization", Document SPARC/90, CBEMA (Washington, DC 1974).
31. ANSI/X3/SPARC, 'Study Group on Data Base Management Systems: Interim Report 75-02-08' // Newsletter ACM SIGMOD Record, FDT, Vol 7, No. 2, 1975. – P. 1-140
32. Tsichritzis D.C., Klug A. "The ANSI/X3/SPARC DBMS Framework". Report of the Study Group on a Database Management System". Information Systems, Vol. 3, No. 4, 1978.
33. CODASYL/Data Description Language Committee (DDLC), "June 73 Report". CODASYL Data Description Language Committee Journal of Development, June 1973
34. "CODASYL Data Description Language Committee Journal of Development", 1978.
35. Concepts and Terminology for the Conceptual Schema and the Information Base, van Griethauzen, J.J., Ed., ISO TC97/SC5/WG3, 1982, Publ. 695.
36) Falkenberg E,D. Structuring and Representation of Information at the Interface Between Data Base User and Data Base Management System. Diss. Univ. Stuttgart (1975).
37. Falkenberg E., Concepts of Modelling Information, Proc. of the IFIP Working Conf. on Modelling in Data Base Management Systems, Nijssen, G.M., Ed., North-Holland, 1976, p. 95-109.
38. Abrial Jean-Raymond, Data Semantics, In: J. W. Klimbie, K. L. Koffeman (eds.), Database Management, Proceedings IFIP TC2 Conference. Grgese, 1974., North-Holland Publishing Company, pp.1-60.
39. Bracchi G., Paolini P., Pelagatti G. "Binary Logical Associations in Data Modelling," in J. M. Nijssen (ed.), Modelling in Database Management Systems (Proc. IFIP TC2 Conference, Freudenstadt), North-Holland Publishing Company, Amsterdam, The Netherlands, 1976.
40. Durchholz R. and Richter G., "Concepts for data base management systems". In: Data Base Management, J. W. Klimbie and K. L. Koffeman, (eds.),
41. Senko, M.E., Conceptual Schemas, Abstract Data Structures, Enterprise Descriptions, In: International Computing Symposium, Liege, Belgium, 1977, North-Holland Publishing Company.
42. Senko M.E., Altman E.B., Astrahan MM., Fehder P.L. Data Structures and Accessing in Data-Base Systems. IBM System J., v. 12, no. 1 (1973).
43. Senko M.E., "The DDL in the Context of Multilevel Structured Description: DIAM II with FORAL". Proc. of the IFIP TC-2 Spe-

cial Working Conference on Data Base Description, pp.239-257, Jan. 1975
44. Smith J.M. and Smith D.C.P. Databas e Abstractions : Aggregation and Generalization. ACM Trans, on Database Syst, v. 2, no. 2, 1977, pp. 105133
45. Smith J.M. and Smith D.C.P. Databas e Abstractions: Aggregation. Comm. of the ACM, v. 20, no. 6, 1977, pp. 405-413
46. Hammer, M. and McLeod, D., Database Description with SDM: A Semantic Database Model, ACM Transactions on Database Systems,1981, Vol. 6, No. 3, pp. 351-386.
47. Abiteboul, S., Hull, R., IFO: A Formal Semantic Database Model, ACM Trans. Database Syst. 12, 4 (1987), 525-565.
48. Bachman, C. W., "Data Structure Diagrams", Data Base, 1969, No 1, 2, pp. 4-10.
49. Engles R.W. A Tutorial on Data-base Organization, Annual review in automatic programming, Vol 7. Part I, Pergamon Press, 1972, 93 p.
50. Chen P.P. The Entity-Relationship Model — Toward a Unified View of Data // ACM Transactions on Database Systems (TODS), 1976. — Vol. 1, No. 1. — P. 9–36.
51. Barker R. Case*Method: Entity Relationship Modelling Publisher: Addison-Wesley, 1990, 240 p.
52. Gogolla M. An extended entity-relationship model – fundamentals and pragmatics. LNCS, vol. 767. Berlin: Springer; 1994.
53. Hartmann S. Reasoning about participation constraints and Chen's constraints. In: The Fourteenth Australian Database Conference, Adelaide, Australia. Conferences in Research and Practice in Information Technology; 2003. p. 105–113.
54. Hohenstein U. Formale Semantik eines erweiterten Entity-Relationship-Modells. Stuttgart: Teubner; 1993.
55. Thalheim B. Entity-relationship modeling – foundations of database technology. Berlin: Springer; 2000.
56. Teorey, T.J., Yang, D. and Fry, J.P. A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model, ACM Computer Surveys, 1986, Vol.18, No. 2. pp. 197-222
57. Vincent S. Lai, Jean Pierre Kuilboer, Jan Lucille Guynes. Temporal databases: model design and commercialization prospects. ACM SIGMIS Database: the DATABASE for Advances in Information Systems, 1994, Vol. 25, No 3, pp. 6-18
58. Gregersen H., Jense C.S. Temporal Entity-Relationship Models—a Survey. IEEE Transactions on Knowledge and Data Engineering, 1999, Vol. 11, No. 3, pp. 464 - 497
59. Codd E.F. "A Relational Model of Data for Large Shared Data Banks," Communications of the ACM, Vol. 13, No. 6 (June 1970), pp 377-397
60. Codd E.F. "A data base sublanguage founded on the relational calculus," Proc. 1971 ACM-SIGFIDET Workshop on Data Description, Access, and Control, Nov. 1971. ACM. New York, 1971, DP. 35-68
61. Codd E. F. "Relational Completeness of Data Base Sublanguages" (presented at Courant Computer Science Symposia Series 6, "Data Base Systems." New York City, N.Y. May 24th-25th. 1971), IBM Research Report RJ987
62. Palermo F.P. "A data base search problem", Proceedings 4th Computer and Information Science Symposium (COINS IV), Miami Beach, Dec. 1972, Plenum Press, New York, 1972. pp. 67–101
63. Codd E.F. "Interactive Support for Nonprogrammers: The Relational and Network Approaches,"Proceedings of the ACM SIGMOD Workshop on Data Description, Access, and Control, Vol. II, Ann Arbor, Michigan, May 1974.
64. Codd E.F. Extending the database relational model to capture more meaning. ACM Trans. on Database Syst., vol. 4, No. 4, 1979, pp. 397-434
65. Codd E. F. "The Second and Third Normal Forms for the Relational Model", IBM technical memo (October 6th. 1970).
66. Codd E.F. "Further Normalization of the Database Relational Model", in Data Base Systems, Courant Inst. Comput.Sci. Symp. Series 6 (New York, 1971), Englewood Cliffs, N.J.: Prentice Hall, 1972, pp. 33-64.
67. Codd E.F. "Normalized Data Base Structure: A Brief Tutorial", Proc. 1971 ACM SIGFIDET Workshop on Data Description, Access. and Control. San Diego. Calif. 1971, p. 1-17
68. Date C. J. The database relational model : a retrospective review and analysisю - Ad-

dison-Wesley Educational Publishers Inc., 2000, 152 p.
69. Codd E.F. "Recent Investigations in Relational Database Systems," Information Processing 74, pp.1017–1021.
70. Heath I.J. Unacceptable File Operations in a Relational Data Base. Conference: Proceedings of 1971 ACM-SIGFIDET Workshop on Data Description, Access and Control, San Diego, California, November 11-12, 1971, pp. 19–33
71. Armstrong William Ward. "Dependency structures of data base relationships". In Jack L. Rosenfeld and Herbert Freeman, editors, Proceedings of IFIP Congress 74, pp.580-583, North Holland, 1974
72. Fagin R. Multivalued Dependencies and a New Normal Form for Relational Databases / R. Fagin // ACM Transactions on Database Systems. – 1977. – Vol. 2, № 1. – P. 262-278.
73. Beeri C., Fagin R., Howard J.H. A complete axiomatization for functional and multivalued dependencies in database relations. Proc. ACM SIGMOD Conf., D.C.P. Smith, Ed., Toronto, Canada, August 1977, pp. 47-61.
74. Zaniolo C. Analysis and design of relational schemata for database systems. Ph.D. Diss., Tech. Rep. UCLA-ENG-7669, U. of California, Los Angeles, Calif., July 1976.
75. Delobel C., Leonard M. The decomposition process in a relational model. Proc. Int. Workshop on Data Structure Models for Information Systems, Presses U. de Namur, Namur, Belgium, May 1974, pp. 57-80.
76. Rissanen J. Theory of relations for databases--A tutorial survey, in "Proc. 7th Sympos. on Math. Found. of Computer Science," 1978, pp. 537-551, Lecture Notes in Computer Science No. 64, Springer-Verlag, Berlin
77. Fagin R. Normal Forms and Relational Database Operators / R. Fagin // Proceedings of the ACM SIGMOD International Conference on Management of Data (Boston, Mass., May 30-June 1), ACM, New York, 1979, p. 153-160
78. Date Chris J. "On DK/NF normal form". - https://web.archive.org/web/20120406123712/http://www.dbdebunk.com/page/page/621935.htm
79. Buy B., Puzikova A. V. Some nonclassical normal forms in relational databases (Rus) // Bulletin of Taras Shevchenko National University of Kyiv. Series Physics & Mathematics, 2015, No 1, pp. 65-74
80. Ling T. W. An Improved Third Normal Form for Relational Databases / T. W. Ling, F. W. Tompa, T. Kameda // ACM Transactions on Database Systems. – 1981. – Vol. 6, № 2. – P. 329-346.
81. Zaniolo C. A New Normal Form for the Design of Relational Database Schemata / C. Zaniolo // ACM Transactions on Database Systems. – 1982. – Vol. 7, № 3. – P. 489-499
82. Normann R. Minimal lossless decompositions and some normal forms between 4NF and PJ/NF / R. Normann // Information Systems. – 1998. – Vol. 23, № 7. – P. 509-516.
83. Vincent M. W. A corrected 5NF definition for relational database design / M. W. Vincent // Theoretical Computer Science (TCS). – 1997. – Vol. 185, № 2. – P. 379-391.
84. Vincent M.W. Redundancy Elimination and a New Normal Form for Relational Database Design / M. W. Vincent // In Semantics in Databases (Libkin, L., Thalheim, B., eds.), vol. 1358 of LNCS. – 1998. – P. 247-264.
85. Darwen H. A Normal Form for Preventing Redundant Tuples in Relational Databases / H. Darwen, C. Date, R. Fagin // Proceedings of the 15th International Conference on Database Theory – ICDT'2012, March 26– 30, 2012, Berlin, Germany. – P. 114-126.
86. Fagin R. A Normal Form for Relational Databases That Is Based on Domains and Keys / R. Fagin // Communications of the ACM. – 1981. – Vol. 6. – P. 387-415.
87. Delobel C. Normalization and hierarchical dependencies in the relational data model. ACM TODS 1978, 3, 3, 201-222.
88. Pasichnik V.V., Stogniy A. A. Relational models of data bases (Rus). - M.: CNIIATOMINFORM, 1983, 268 p.
89. Casanova M.A. Inclusion dependencies and their interaction with functional dependencies / M. A. Casanova, R. Fagin, C. H. Papadimitriou // Journal of Computer and System Sciences. – 1984. – № 28. – P. 29-59.
90. Nicolas J.M. Mutual dependencies and same results on indecomposable relations / J. M. Nicolas // Proceedings of the fourth interna-

tional conference on Very Large Data Bases, 1978. – Vol. 4. – P. 360-367.
91. Ling T.W. Logical Database Design with Inclusion Dependencies / T. W. Ling, C. H. Goh // In Proceedings of the Eighth International Conference on Data Engineering, Tempe, Arizona, 1992. – P. 642-649.
92. Levene M. Justification for Inclusion Dependency Normal Form / M. Levene, M. W. Vincent // IEEE Transactions on Knowledge and Data Engineering, 2000. – Vol. 12, № 2. – P. 281-291.
93. Thalheim B. Bibliographie zur Theorie der Abhangigkeiten in relationalen Datenbanken, 1970-1984, TU Dresden 566/85, Dresden 1985.
94. Thalheim B. Dependencies in Relational Databases, 1991, Teubner-Texte zur Mathematik, 214 Pages
95. Chamberlin D.D. "Relational Data-Base Management Systems," Computing Surveys, Vol. 8, No. 1, p. 43-66, March 1976
96. Goldstein R.C., Strnad A.L. "The MACAIMS Data Management System," Proceedings of t he ACM-SIGFIDST Workshop on Data Description, Acess and Control, Nov. 1970. ACM, New York, 1970, pp. 201-229.
97. Notley M.G. "The Peterlee IS/1 system," IBM UK Scientific Centre Report UKSC-0018, March 1972.
98. Todd S.J.P. "Peterlee relational test vehicle PRTV, a technical overview," IBM Scientific Centre Report UKSC 0075, Peterlee, England, July 1975.
99. Whitney V.K.M. "RDMS: A Relational Data Management System," Proceedings of the Fourth International Svmposium on Computer and Information Sciences (COINS IV), Dec. 1972, Plenum Press, New York, 1972.
100. Pecherer R.M. " Efficient evaluation of expressions in a relational algebra," Proc. ACM Pacific 76 Regional Conf., April 1975, ACM, New York, 1975, pp. 44-49.
101. Gotlieb L.R. "Computing joins of relations, Proc. ACM-SIGMOD International Conference on Management of Data (San Jose, Calif., May 14-16, 1975), ACM. New York, 1975, pp. 55-63
102. Smith J.M., Chang P. "Optimizing the performance of a relational algebra data base interface," Comm. ACM 18, 10 (Oct. 1975), pp. 568-579.
103. Hall P. A. V. Optimisation of a single relational expression in a relational data base system, IBM Scientific Centre Renort UKSC 0076. Peterlee, England, July 1975.
104. Palermo F.P. An APL environment for testing relational operators and data base search algorithms. Proc. APL 75 Conf., June 1975, ACM. New York, 1975, pp. 249-256
105. Bui D.B., Skobelev V.G. Complexity of operations in database systems (a survey), Radioelectronic and computer systems, 2014, No 6(70). pp. 53-59
106. Held G.D., Stonebraker M.R., Wong E. "INGRES: a relational data base system," Proc. AFZPS h'ational Computer Conf., May 1975, Vol. 44, AFIPS Press, Montvale, N.J., 1975, pp 409-416.
107. Bracchi G., Fedeli A., Paolini P. A language for a relational data base management system. Proc. Sixth Annual Princeton Conf. on Information Science and Systems, March 1972, Princeton Univ., N.J., 1972. pp. 84-92.
108. Fehder P.L. The representation-independent language. Res. Rep. RJ 1121, IBM Research Laboratory, San Jose, Calif., Nov. 1972
109. McDonald N.,Stonebraker M. "CUPID — The Friendly Query Language," University of California, Berkeley, Technical Report No. UCB/ERL M487, October 1974. http://www2.eecs.berkeley.edu/Pubs/TechRpts/1974/ERL-m-487.pdf
110. McDonald N.,Stonebraker M Cupid--The friendly query language. Proc. ACM- Pacific-75, San Francisco, Calif., April 1975, pp. 127-131.
111. McDonald N. "Cupid: A Graphics Oriented Facility for Support of Non-Programmer Interactions with a Data Base," University of California, Berkeley, Technical Report No. UCB/ERL M563, November 1975.
112. McDonald N., Stonebraker M. "CUPID: the friendly query language," Proc. ACM Pacific 75 Regional Conf., Auril 1975. ACM. New York. 1975, pp, 127-131.
113. Zloof M.M."Query by example " RC4917, IBM T. J. Watson Research Center. Yorktown Heights, N. Y., July 1974.
114. Zloof M.M. "Query by Example," Proc. AFIPS National Computer Conf., May 1975, Vol. 44, AFIPS Press, Montvale, N.J., 1975, pp 431-438.

115. Zloof M.M. “Query by Example: the invocation and definition of tables and forms,” Proc. Internatl. Conf. on Very Large Data Bases, Sept. 1975, ACM, New York, 1975, pp. 1-24.
116. Zloof M.M. Query-by-Example: a data base language. IBM System J., 16:4, 1977, pp. 324–343
117. Thomas J. C., Gould J.D. “A psychological study of Query by Example,” Proc. AFIPS National Computer Conf., May 1975, Vol. 44, AFIPS Press, Montvale, N.J., p 439-445.
118. Zloof M.M. Office-by-Example: A business language that unifies data and word pro-cessing and electronic mail. IBM Systems Journal (Volume: 21, Issue: 3, 1982). Page(s): 272 - 304
119. Boyce R.F., Chamberlin D.D., King W.F., Hammer M.M. Specifying queries as relational expressions. Proc. ACM SIGPLAN/ SIGIR Interface Meeting, Gaithersburg, Md., Nov. 1973.
120. Boyce R.F., Chamberlin D.D., King W.F., Hammer M.M. Specifying queries as relational expressions: the SQUARE data sublanguage. Communications of the ACM, 1975, Volume 18, No 11, p. 621-628
121. Boyce, R.F., Chamberlin D.D. Using a structured English query language as a data definition facility. Res. Report RJ 1318, IBM Res. Lab., San Jose, Calif., Dec. 1973.
122. Chamberlin D D., Boyce R.F. SEQUEL: A structured English query language. SIGFIDET ‘74: Proceedings of the 1974 ACM SIGFIDET (now SIGMOD. workshop on Data description, access and control. May 1974 Pages 249–264.
123. Astrahan M.M., Chamberlin D.D. Implementation of a structured English query language. Communications of the ACM, 1975, Volume 18, No 10 pp. 580-588
124. Lorie R.A. XRM-an extended (n-ary. relational memory. Tech. Report G320-2096, IBM Scientific Center, Cambridge, Mass., Jan. 1974.
125. Astrahan M.M., Lorie R.A. “SEQUEL-XRM: a relational system,” Proc. ACM Pacific 76 Regional Conf., April 1975, ACM, New York, 1975, pp. 34-38.
126. Symonds A.J., Lorie, R. A. “A schema for describing a relational data base,” Proc. ACM-SIGFIDET Workshop on Data Description and Access, Nov. 1970, ACM, New York, 1970, pp. 230-245.
127. Lorie R.A., Symonds,A.J. “A relational access method for interactive applications,” Courant Computer Science Symposia, 6, Data Base Systems, Prentice-Hall, New York, 1971, pp 99-124.
128. Chamberlin D D., Astrahan M.M., Eswaran K.P., Griffiths P.P., Lorie R.A., Mehl J.W., Reisner Ph., Wade B.W. SEQUEL 2: A Unified Approach to Data Definition, Manipulation, and Control. IBM Journal of Research and Development 20(6): 560-575 (1976)
129. Carlson C.R., Kaplan R.S. A Generalized Access Path Model and Its Application to a Relational Data Base System. SIGMOD ‘76: Proceedings of the 1976 ACM SIGMOD international conference on Management of data. June 1976, Pages 143–154
130. Notley M, “Peterlee IS/1 System”, UKSC Report 18, 1972
131. Todd S. “The Peterlee Relational Test Vehicle - A System Overview”. IBM Systems Journal. 1976, 15 (4): 285–308.
132. Astrahan M.M., et al. System R: A relational approach to database management. ACM Trans. Database Syst. Vol. 1, No 2 (June 1976), 97-137
133. Chamberlin D.D. A summary of user experience with the SQL data sublanguage. Proc. Internat. Conf. Data Bases, Aberdeen, Scotland, July 1980, pp. 181-203
134. Chamberlin D.D., et al. Support for repetitive transactions and ad-hoc queries in System R. ACM Trans. Database Syst. Vol. 6, No 1 (March 1981), 70-94.
135. Chamberlin D.D., Gilbert, A.M., Yost, R.A. A history of System R and SQL/data system. VLDB ‘81: Proceedings of the seventh international conference on Very Large Data Bases - Volume 7, September 1981, pp. 456–464
136. McDonald N., Stonebraker M., Wong E. “Preliminary design of INGRES: Part I,” Electronics Research Lab. Report ERL-M435, Univ. of California, Berkeley, April 1974. http://www2.eecs.berkeley.edu/Pubs/TechRpts/1974/ERL-m-435.pdf
137. McDonald N., Stonebraker M., Wong E. “Preliminary design of INGRES: Part II,” Electronics Research Lab. Report ERL-

M436, Univ. of California, Berkeley, April 1974. http://www2.eecs.berkeley.edu/Pubs/TechRpts/1974/ERL-m-436.pdf

138. Held G., Stonebraker M. "Storage structures and access methods in the raational data base management system INGRES," Proc. ACM Pacific 75 Regional Conf., April 1975, ACM, New York, 1975, pp 26-33.
139. Wong E., Youssefi K. Decomposition--A strategy for query processing. ACM Trans. on Database Systems I, 3 (Sept. 1976), 223-241
140. Stonebraker M. "Implementation of integrity constraints and views by query modification," Proc. ACM-SIGMOD Conf. May 1975, ACM, New York, 1975, pp 65-78.
141. Stonebraker M., Wong E. Access control in a relational data base management system by query modification. Proc. 1974 ACM Nat. Conf., San Diego, Calif., Nov. 1974, pp. 180-187.
142. Stonebraker M. "High level integrity assurance in relational data base management systems," Electronics Research Lab. Report ERL-M473, Univ. of Calif. at Berkeley, August 1974.
143. Stonebraker M., Rubinstein P. The INGRES protection system. Proc. 1976 ACM National Conf., Houston, Tex., Oct. 1976
144. Stonebraker M., Held G., Wong E., Kreps P. "The Design and Implementation of INGRES". ACM Transactions on Database Systems. Vol.1, No 3. 1976 pp.189–222.
145. Stonebraker M., Rowe L. "The design of POSTGRES," in Proc. 1986 ACM-SIGMOD Conf., Washington, DC, June 1986.
146. Rowe L.A., Stonebraker M. "The POSTGRES data model," in Proc. 13th Intl. Conf. on Very Large Data Bases, P. M. Stocker, W. Kent, P. Hammersley, Eds., San Francisco, CA: Morgan Kaufmann Publishers Inc., 1987, pp. 83-96.
147. Stonebraker M. " The design of the POSTGRES storage system ", in Proc. 1987 VLDB Conf:, Brighton, England, Sept. 1987.
148. Stonebraker M., Hanson E., Hong C. H. "The design of the POSTGRES rules system", Proc. IEEE Conference on Data Engineering, Feb. 1987.
149. Stonebraker M., Rowe L.A., Hirohama M. The Implementation Of Postgres IEEE Transactions on Knowledge and Data Engineering, 1990, Vol. 2, No 1, pp. 125 - 142
150. ACM Turing Award Goes to Pioneer in Database Systems Architecture: MIT's Michael Stonebraker Brought Relational Database Systems from Concept to Commercial Success. - https://www.prweb.com/pdfdownload/12607207.pdf
151. Kuznetsov S.D. Methods for optimization of query execution in relational DBMS (Rus) // "Vychislitelnye nauki. Vol. 1 (Itogi nauki i tekhniki VINITI AN USSR" M.; VINITI AN USSR, 1989.- 76-153. - http://masters.donntu.org/2002/foreign/aswad/lib/mpbd.htm или http://citforum.ru/database/articles/art_26.shtml
152. Brodie M.L., Schmidt J.W. Final Report of the ANSI/X3/SPARC DBS-SG Relational Database Task Group. SIGMOD Record 12(4): i-62 (1982).

Литература по транзакциям

153. Gray J. The Transaction Concept: Virtues and Limitations. In: Proceedings of the 7th International Conference on Very Large Databases, 1981. pp. 144—154, IEEE, Cannes, France,
154. Advanced Transaction Models and Architectures. Sushil Jajodia and Larry Kerschberg (eds.) Springer Scienee+Business Media New York. 1997
155. Gray J., Lorie R., Putzulo G. "Granularity of Locks and Degrees of Consistency in a Shared Data Base," In Modelling in Data Base Management Systems. G.M. Ni]ssen, (ed.) North Holland Publishing Company, 1976, pp.365-394
156. Reis D.R., Stonebraker M. Effect of locking granularity in a database management systems. ACM Trans. on Database Syst., 2:3, 1977, pp. 233-246.
157. Gray J.N. Notes on data base operating systems. In: Bayer R., Graham R.M., Seegmüller G. (eds) Operating Systems. Lecture Notes in Computer Science, vol 60. Springer, Berlin, Heidelberg. 1978. p. 393–481.
158. Eswaran K.P, Gray J, Lorie R.A, Traiger I.L. The notions of consistency and predicate locks in a database system. Commun ACM. 1976;19(11):624–633.
159. Lampson B.W. Atomic transactions. In: Lampson B.W, Paul M, Siegert H.J, editors. Distributed systems – architecture and

implementation: an advanced course, LNCS, vol. 105. Berlin: Springer; 1981. p. 246–285.
160. Lomet D.B. Process structuring, synchronization, recovery using atomic actions. ACM SIGPLAN Not. 1977; 12(3):128–137.
161. Bernstein P.A, Shipman D.W, Wong W.S. Formal aspects of serializability in database concurrency control. IEEE Trans Software Eng. 1979;SE-5(3): 203–216.
162. Bernstein P.A, Hadzilacos V, Goodman N. Concurrency control and recovery in database systems. Reading: Addison-Wesley; 1987.
163. Papadimitriou C.H. The serializability of concurrent database updates. J ACM. 1979;26(4):631–653.
164. Papadimitriou C.H. The theory of database concurrency control. Rockville: Computer Science; 1986.
165. Weikum G, Vossen G. Transactional information systems – theory, algorithms, the practice of concurrency control and recovery. San Francisco: Morgan Kaufmann; 2002.
166. Shasha D, Bonnet P. Database tuning – principles, experiments, and troubleshooting techniques. San Francisco: Morgan Kaufmann; 2003.
167. Ramamritham, K., Chrysanthis, P. K., (1997). Advances in Concurrency Control and Transaction Processing. IEEE Computer Society Press, Los Alamitos, California.
168. Grefen P., Apers P. (1993). Integrity Control in Relational Database Systems - An Overview. Journal of Data & Knowledge Engineering (10)2: 187-223.
169. Moss J.E.B. Nested transactions: an approach to reliable distributed computing. Technical Report. PhD Thesis. UMI Order Number: TR-260: Massachusetts Institute of Technology; 1981. p. 178.
170. Been C, Bernstein P.A., Goodman N, Lai M.Y., Shasha D.E. A concurrency control theory for nested transactions. Proc. of Second ACM Symposium on Principles of Database Systems (PODS), 1983, pp. 45-62
171. Davies C.T. Data processing spheres of control. IBM Syst J. 1978;17(2):179–198.
172. Dayal U., Hsu M., Ladin R. A generalized transaction model for long-running activities and active databases. IEEE Data Engineering Bulletin, March 1991, vol. 14, No 1,. pp 4-8
173. Weikum G. and Schek H. Concepts and applications of multilevel transactions and open-nested transactions. In Elmagarmid A., editor. Database Transaction Models for Advanced Applications. Morgan Kaufmann Publishers. San Mateo. CA., 1992, pp. 515–553.
174. Weikum G. Principles and realization strategies of multilevel transaction management. ACM Transactions on Database Systems. 1991;16(1):132–180.
175. Krychniak P., Rusinkiewicz M., Chichocki A., Sheth A., Thomas G. Bounding the Efects of Compensation under Relaxed Multi-Level Serializability. Distributed and Parallel Database Systems, 1996, 4(4), pp. 355-374
176. Lewis, P. M., Bernstein A. J., Kifer M. (2002). Databases and Transaction Processing: An Application-Oriented Approach. Addison-Wesley, United States
177. Breitbart Y., Garcia-Molina H., Silberschatz A. Overview of multidatabase transaction management. VLDB Journal, 1992, vol. 1, No 2, pp. 181-240.
178. X/Open Company Ltd., (1996). Distributed Transaction Processing: Reference Model, version 3. X/Open Company Ltd., U.K.
179. Elmagarmid A.K., Leu Y., Litwin W.,Rusinkiewicz M. (1990) A Multidatabase Transaction Model for InterBase. In Proc. of the 16th. Intl. Conference on Very Large Data Bases, pp. 507-518, Brisbane. Australia
180. Zhang A., Nodine M., Bhargava B., Bukhres O. Ensuring Relaxed Atomicity for Flexible Transactions in Multidatabase Systems. In Proc/ 1994 SIGMOD International Conference on Management of Data, 1994, pp. 67-78
181. Zhang A, Nodine M, Bhargava B. Global scheduling for flexible transactions in heterogeneous distributed database systems. IEEE Trans Knowl Data Eng. 2001;13(3):439–450.
182. Wächter H, Reuter A. The ConTract model. In: Elmagarmid A.K., editor. Database transaction models for advanced applications. Los Altos: Morgan Kaufmann; 1992. pp 39-43

183. Veijalainen J., Eliassen F. The S—transaction Model. In: Elmagarmid A.K., editor. Database transaction models for advanced applications. Los Altos: Morgan Kaufmann, 1992, pp. 55-59
184. Chen J., Bukhres O., Elmagarmid A. K. (1993). IPL: A Multidatabase Transaction Specification Language. In Proc. of the 13th Intl. Conference on Distributed Computing Systems - ICDCS '93. 1993, pp. 439-448
185. Garcia-Molina H. Using Semantic Knowledge for Transaction Processing in a Distributed Database. ACM Transactions on Database Systems, 8(2):186-213, June 1983.
186. Korth H., Levy E., Silberschatz A. A Formal Approach to Recovery by Compensating Transactions. In Proceedings of the 16th International Conference on Very Large Data Bases, Brisbane, Australia, 1990, pp. 95–106
187. Garcia-Molina H., Salem K. Sagas. In Proc. of ACM SIGMOD International Conference on Management of Data, 1987, pp 249-259 San Francisco, CA.
188. Bancilhon F., Kim W., Korth H. A model of CAD Transactions. VLDB '85: Proceedings of the 11th international conference on Very Large Data Bases - Volume 11, 1985, pp. 25–33
189. Garcia-Molina. H., Salem K., Gawlick D., Klein J., Kleissner K., Modeling Long-Running Activities as Nested Sagas, IEEE Data Engineering Bulletin, 1991, 14(1) pp 14-18
190. Pu C., Kaiser G.E., Hutchinson N.C. Split-transactions for open-ended activities. In: Proceedings of the 14th International Conference on Very Large Data Bases; 1988. p. 26–37.
191. Kaiser G.E., Pu C. Dynamic restructuring of transactions. In: Elmagarmid AK, editor. Database transaction models for advanced applications. Burlington: Morgan Kaufmann Publishers; 1992. p. 265–295.
192. Chrysanthis P.K, Ramamritham K. Synthesis of extended transaction models using ACTA. ACM Trans. Database Syst. 1994;19(3):450–491.
193. Nodine M.H., Zdonik S.B. Cooperative transaction hierarchies: Transaction support for design applications. VLDB Journal, 1(1):41–80, 1992.
194. Chrysanthis P.K., Ramamritham, K., (1990). ACTA: A Framework for Specifying and Reasoning about Transaction Structure and Behavior. Proceedings of the ACM SIGMOD International Conference on Management of Data: 194-203.
195. Chrysanthis P.K., Ramamritham K. (1992). ACTA: The SAGA Continues. In Elmagarmid A., editor. Database Transaction Models for Advanced Applications. Morgan Kaufmann Publishers. San Mateo. CA., 1992, pp. 349-397
196. Biliris A., Dar S., Gehani N., Jagadish H., Ramamritham K. (1994). ASSET: A System for Supporting Extended Transactions. In Proc. of ACM SIGMOD Conference on Management of Data, pages 44-54, Minneapolis, M.N.
197. Abbott R., Garsia-Molina H. Scheduling real-time transactions: a performance evaluation. ACM Trans, on Database Syst, 17(3), September 1992, pp. 513-560
198. Agrawal D., El Abbadi A., Jeffers R. Using Delayed Commitment in Locking Protocols for Real-Time Databases. SIGMOD Conference 1992: 104-113
199. Hong D., Johnson T., Chakravarthy S. Real-Time Transaction Scheduling: A Cost Conscious Approach. SIGMOD Conference 1993: 197-206.
200. Alonso R., Korth H. Database System Issues in Nomadic Computing. SIGMOD Record, Vol. 22, No 2,1993, pp.388-392.
201. Imelinski T., Badrinath B.R. Data Management for Mobile Computing. SIGMOD Record, Vol. 22, No. 1, 1993
202. Ceponkus A., Dalal S., Fletcher T., Furniss P., Green A., Pope B. Business transaction protocol, Version 1.1, 2002
203. Business transaction protocol. - http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=businesstransaction [2004]
204. Stevens M., Mathew S., McGovern J., Tyagi S. Java Web Services Architecture. San Francisco: Morgan Kaufmann Publishers, 2003.
205. WSTx (Web Services Transactions). - https://searchapparchitecture.techtarget.com/definition/WSTx-Web-Services-Transactions
206. IBM, BEA Systems, Microsoft, Arjuna, Hitachi, IONA, \Web Services Transactions

specifations," IBM Developer Works, IBM, 2004.
207. Curbera F., Khalaf R., Mukhi N., Tai S.,Weerawarana S. The Next Step in Web Services," Communications of the ACM, October 2003, Vol. 46, No. 10, Pages 29-34
208. OASIS Web Services Composite Application Framework (WS-CAF), OASIS, 2006. - http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-caf
209. Little M., Freund Th. J.. A comparison of web services transaction protocols: A comparative analysis of WS-C/WS-Tx and OASIS BTP," IBM, 2003. Available: http://www-128.ibm.com/developerworks/webservices/library/ws-comproto/. [Accessed May 2008].
210. Kratz B., Protocols For Long Running Business Transactions. Technical Report 17, Infolab Technical Report Series, 2004, 48 p.
211. Jin T., Goschnick S. (2004) Utilizing Web Services in an Agent Based Transaction Model. In: Cavedon L., Maamar Z., Martin D., Benatallah B. (eds) Extending Web Services Technologies. Multiagent Systems, Artificial Societies, and Simulated Organizations (International Book Series), vol 13. Springer, Boston, MA. pp 273-291

Одержано 24.06.2021

Про автора:

Резніченко Валерий Анатолієвич,
кандидат фізико-математичних наук,
заступник завідувача відділом.

Кількість публікацій в українських виданнях – 61.
Кількість зарубіжнихих публікацій – 4.
Індекс Хірша – 12.
http://orcid.org/0000-0002-4451-8931.

Місце роботи автора:

Інститут програмних систем НАН України, 03187, м. Київ-187,
проспект Академіка Глушкова, 40.
Тел.: (044) 526 3559.
E-mail: reznich@isofts.kiev.ua