УДК 004.652.4

А.В.Бармак, М.Л.Яновский

ПОДХОД ДЛЯ МОДЕЛИРОВАНИЯ СТРУКТУРЫ ПРЕДМЕТНОЙ ОБЛАСТИ В ВИДЕ N-МЕРНОГО КУБА

ВВЕДЕНИЕ

Рост производительности компьютеров, объемов их оперативной и внешней памятей, пропускной способности внешних устройств и каналов связи качественно изменил ситуацию в вычислительной технике и сферах ее применения. Можно говорить об эпохе массовой компьютеризации. Основным предметом труда этой эпохи становится информация, а новым орудием труда – компьютеры. Наблюдается значительный разрыв между возможностями аппаратных средств компьютеров и применяемыми методами решения прикладных задач. Наиболее освоенные на сегодня методы основаны на хорошо формализованных алгоритмах, полученных в результате построения математических моделей предметных областей. Чаще всего это трудоемкие расчеты по известным формулам либо простые последовательности действий, приводящие после многократного применения к желаемому результату. Однако в практической деятельности многие актуальные задачи относятся к плохо формализованным, для которых неизвестны аналитические зависимости или цепочки действий, приводящее к результату без интеллектуального вмешательства человека. Ранее для решения таких задач просто не хватало ресурсов компьютера, поэтому было бессмысленно ставить саму проблему решения плохо формализованных задач.

Как правило, в плохо формализованных задачах имеется некоторый набор параметров, описывающий объекты предметной области. Вся информация о предметной области, которая может быть использована для решения плохо формализованной задачи, представлена некоторой совокупностью векторов этих параметров, представляющих подвергшиеся измерению объекты – т.е. можно говорить об n-мерных векторных пространствах. Хотя о

наборе параметров нельзя определенно сказать, что он полон, адекватен, и сами используемые значения параметров в совокупности неполны, противоречивы и искажены.

Методы решения плохо формализованных задач имеют дело с обработкой данных. Поэтому одним из вопросов является рассмотрение способов организации хранения и выборки данных о предметных областях в базах данных в зависимости от решаемой задачи. Любая информационная система оперирует той или иной частью реального мира – предметной областью. Предметная область рассматривается как некоторая совокупность реальных объектов (сущностей) и связей между ними. Каждый объект обладает определенным набором свойств (атрибутов). База данных это отражение предметной области, т.е. «материализация» в форме хранимой в памяти компьютера структурированной совокупности данных, характеризирующих состав объектов предметной области, их свойства и взаимосвязи.

Известно, что в зависимости от характера информационных ресурсов (информация фактического типа – характеристики объектов и связи, документальная информация - текстовая) различаются два класса систем хранения данных – документальные и фактографические. Далее рассматриваются фактографические системы. Они оперируют фактическими сведениями, представленными в виде специальным образом организованных совокупностей формализованных записей данных.

Среди фактографических систем важное место занимают два класса: системы операционной обработки данных и системы, ориентированные на анализ данных и поддержку принятия решений. Для обозначения систем операционной обработки используют термин OLTP (On-Line Transaction Processing – оперативная обработка транзакций или выполнение транзакций в реальном времени). Другой класс систем – системы поддержки принятия решений – аналитические системы – их обозначают OLAP (On-Line Analysis Processing – система оперативной аналитической обработки). Оба класса систем основаны на системах управления базами данных (СУБД), но типы выполняемых ими запросов сильно различаются. Принципиально отличаются и структуры баз данных.

Способ отображения сущностей, атрибутов и связей на структуры данных определяется моделью данных. На сегодняшний день, де факто, для OLTP-систем, самой распространенной моделью данных является реляционная. Соответствующая ей СУБД называется реляционной. В реляционных базах данных вся информация представляется в виде прямоугольных таблиц. Для манипулирования отношениями используют операции реляционной алгебры. При проектировании базы данных стараются сделать так, чтобы отображение объектов предметной области в структуры модели данных не противоречило семантике предметной области. Оно должно быть эффективным, т.е. – обеспечивать минимальное дублирование данных. Оптимальная структура базы данных получается методом нормализации отношений. Т.е. пошаговым процессом разложения (декомпозиции) исходных отношений на более простые. Для получения информации из базы данных направляются запросы к СУБД. Запросы формулируются на специальном «языке запросов». Фактическим стандартом такого языка для современных реляционных СУБД является SQL (Structured Query Language – структурный язык запросов).

Задачи, решаемые OLTP и OLAP системами, существенно различаются, поэтому и их базы данных тоже построены на разных принципах. Критерием эффективности для систем операционной обработки служит число транзакций, которое они способны выполнить за единицу времени [1]. Для аналитических систем важнее скорость выполнения сложных запросов и прозрачность структуры хранения информации для пользователей [2].

В начале 90-х годов прошлого века Билом Инмоном была предложена концепция хранилищ данных. Это концепция подготовки данных для последующего анализа. Одним из важных положений этой концепции было разделение наборов данных, используемых системами выполнения транзакций и аналитическими системами. В работе «Создание хранилища данных» («Building the Data Warehouse») Бил Инмон определил хранилище данных, как «предметно-ориентированный, интегрированный, неизменяемый и поддерживающий хронологию набор данных, предназначенный для обеспечения принятия управленческих

решений». Рассмотрим подробнее свойство «ориентация на предметную область»: 1) хранилище должно разрабатываться с учетом предметной области, а не приложений, оперирующих данными; 2) структура хранилища должна отражать представления аналитика об информации, с которой ему приходиться работать.

ПОСТАНОВКА ЗАДАЧИ

Основываясь на вышеизложенном, предлагается следующая ***постановка задачи***. Предложить единую, оптимальную, как с точки зрения систем операционной обработки, так и с точки зрения аналитических систем структуру хранения и выборки данных для плохо формализуемых задач. Структура хранения должна удовлетворять следующим основным требованиям:

1) оптимальность как с точки зрения обеспечения приемлемого времени отклика на аналитические запросы, так и приемлемого времени отклика на массовую обработку транзакций;
2) обеспечение защиты от несанкционированного доступа, от нарушения целостности, от аппаратных и программных сбоев;
3) первоначальное заполнение и последующее пополнение данных;
4) обеспечение удобства доступа пользователей к данным.

ОПИСАНИЕ ПОДХОДА

Существуют два подхода к построению хранилищ данных: подход, основанный на использовании многомерной модели базы данных (Multidimensional OLAP - MOLAP), и подход, использующий реляционную модель базы данных (Relational OLAP - ROLAP).

Информацию можно представлять в виде n-мерного гиперкуба. Представление данных в виде гиперкуба более наглядно, чем совокупность нормализованных таблиц, оно понятно не только администратору базы данных, но и пользователям. В гиперкубе каждое значение находится в строго определенной ячейке. На рис.1. каждое значение связано с точкой в 3-х мерном пространстве (N,S,T) с измерениями: N – название параметра; S – субъект; T – момент

времени. Число возможных параметров конечно, поэтому все возможные значения можно представить в виде гиперкуба.

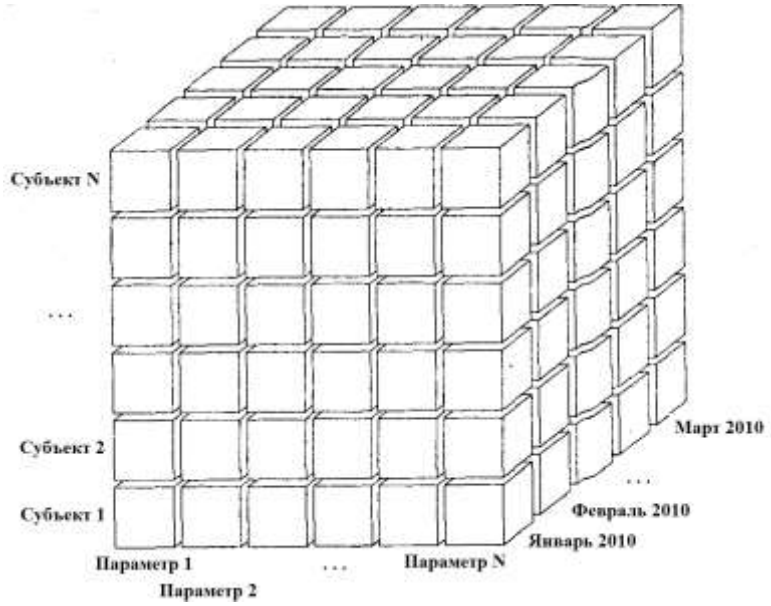


Рис. 1. Представление данных в виде гиперкуба

Основные понятия многомерной модели – ***измерение*** и ***значение*** (ячейка). Измерение – это множество, образующее одну из граней гиперкуба (аналог домена в реляционной модели). Измерения играют роль индексов, используемых для для идентификации конкретных значений в ячейках гиперкуба. Значения – это подвергаемые анализу количественные или качественные данные, которые находятся в ячейках гиперкуба.

У многомерных СУБД имеются серьезные недостатки, сдерживающие их применение. Многомерные СУБД неэффективно, по сравнению с реляционными, используют память. В многомерной СУБД заранее резервируется место для всех значений, даже если часть из них заведомо будет отсутствовать. Другой недостаток состоит в том, что выбор высокого уровня детализации при реализации гиперкуба может очень сильно увеличить размер многомерной СУБД.

Основой при построении хранилища данных может служить и традиционная реляционная модель данных. В этом случае гиперкуб эмулируется СУБД на логическом уровне. В отличие от многомерных реляционные СУБД способны хранить огромные объемы данных,

однако, в некоторых случаях, они проигрывают по скорости выполнения аналитических запросов.

При использовании реляционных СУБД для организации хранилища данные располагаются специальным образом. Чаще всего используется так называемая радиальная схема («звезда»(star)). В этой схеме используются два типа таблиц: таблица фактов (фактологическая таблица) и несколько справочных таблиц (таблицы измерений).

В таблице фактов обычно содержаться данные, наиболее интенсивно используемые для анализа. Запись в фактологической таблице соответствует ячейке гиперкуба. В справочной таблице перечислены возможные значения одного из измерений гиперкуба. Каждое измерение описывается своей собственной справочной таблицей. Фактологическая таблица индексируется по сложному ключу, скомпонованному из индивидуальных ключей справочных таблиц. На рис.2 приведена упрощенная схема для примера из рис.1.

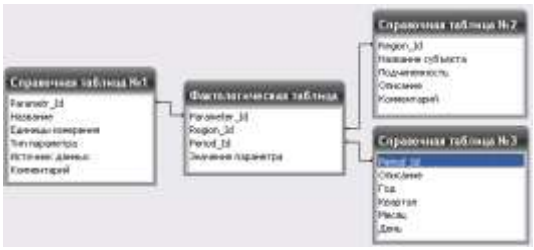


Рис. 2. Пример базы данных с радиально связанными таблицами

Рис. 3. Пример базы данных со схемой «снежинка»

В реальных системах количество строк в фактологической таблице может составлять десятки и сотни миллионов. Число справочных таблиц обычно не превышает двух десятков.

Если база данных включает большое количество измерений, можно использовать схему «снежинка» (snowflake). В той схеме атрибуты справочных таблиц могут быть детализированы в дополнительных справочных таблицах (см. рис.3).

Приведенный выше анализ показывает, что можно оперировать информационным пространством как n-мерным векторным пространством. Авторами сделана попытка объединить в одной структуре как OLTP так и OLAP подход.

Предлагается представлять информацию как многомерную структуру.

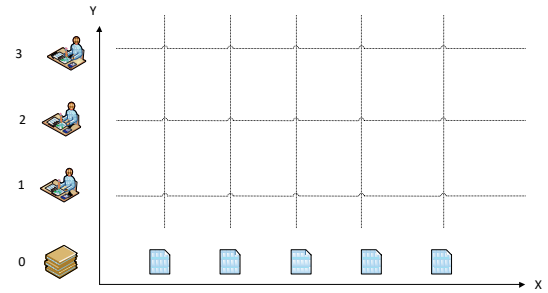


Рис.4.

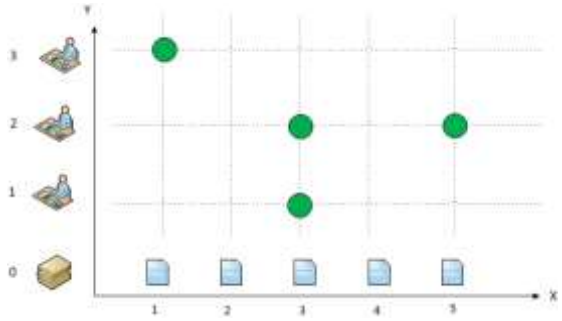


Рис.5

Компоненты информационного пространства разлагаются по оси X (рис.4). По оси Y на нулевом уровне будет исходный массив информации. Рассмотрим его как одномерный, укладывающийся в одну линию на ось X. Далее, по оси Y, над массивом первичной информации, идут пользователи. На рис. 5 видно, что пользователь 1 сформировал свою индивидуальную информационную среду из элементов исходного массива под номером 3. Пользователь 2 использует элементы 3 и 5, пользователь 3 – только элемент 1 (зеленая точка с координатами (1,3).

Пусть, на оси X расположены учебные предметы, в целом, без рассмотрения глав, страниц, иллюстраций и прочих составляющих. Тогда в базе данных это будет представлено простой таблицей (рис. 6)

X	Учебный предмет
1	Физика
2	Математика
3	История
4	Рисование
5	Танцы

Рис.6. Таблица [X] - расшифровка значений по оси X

Y	Наименование
0	Учебный материал
1	Иванов
2	Петров
3	Сидоров

Рис.7. Таблица [Y] – расшифровка значений по оси Y

Расшифровка по оси Y, без акцентирования внимания на том, кто такой Петров, где он учится и сколько ему лет представлена в другой простой таблице (рис.7). Для создания классической структуры реляционной базы данных понадобилось бы добавить таблицу [YX], связывающую учебные предметы с учениками. И тогда довольно простая структура базы данных содержала бы три таблицы (рис. 8). Но для создания модели многомерного информационного пространства результатом объединения [X] и [Y] будет не три таблицы, а одна таблица, назовем ее условно [Matrix] (рис.9).

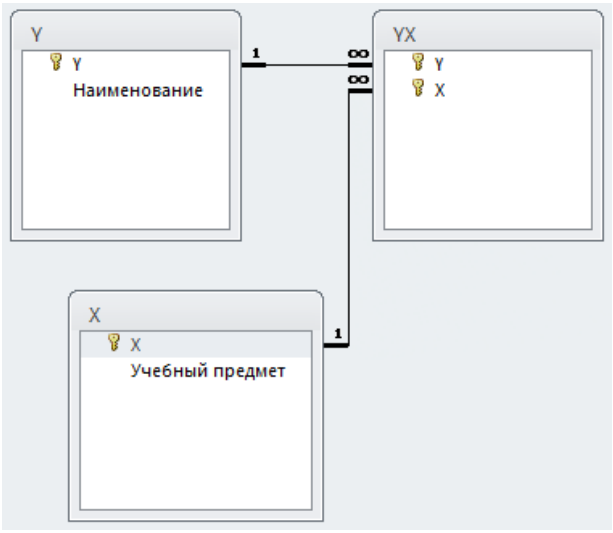


Рис.8. Классическая реляционная структура для связывания двух таблиц

Matrix

Y	X	Информация
0	1	Физика
0	2	Математика
0	3	История
0	4	Рисование
0	5	Танцы
1	0	Иванов
2	0	Петров
3	0	Сидоров
1	3	Иванов изучает Историю
2	3	Петров изучает Историю
2	5	Петров изучает Танцы
3	1	Сидоров изучает Физику

Рис.9 Таблица [Matrix]

Первые 5 строк в таблице [Matrix] – это измерение информации об учебных предметах. Следующие три строки описывают фамилии учащихся. Последние четыре строки – измерение информации об индивидуальных учебных планах – кто что изучает. В одном массиве представлена вся необходимая на данный момент информация. Каждый элемент информации имеет уникальные координаты, увязанные в общее пространство.

Расширим информационное пространство. Для большей детализации добавим координату Z. На срезе координаты Z=1 имеем имена. Если сделаем выборку при Z=3 – получим даты рождений. Таким образом, информация о дне рождения Петрова имеет уникальные координаты - (0,2,3).

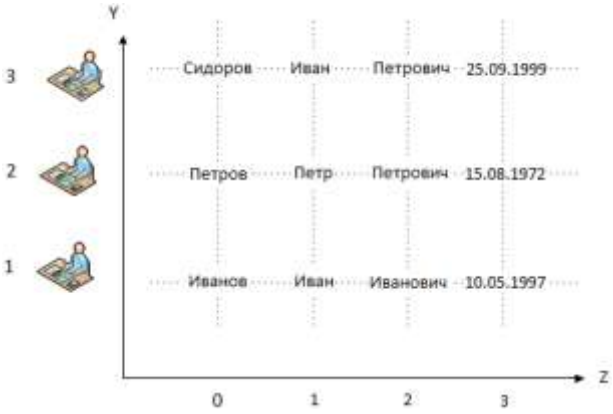


Рис.10.

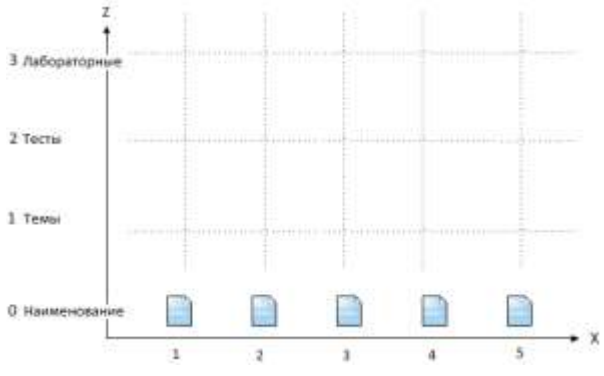


Рис.11

Добавив координату Z к учебному материалу, мы не получаем непосредственно информацию, а только опускаемся на один уровень абстракции ниже. В учебных материалах одной координатой Z не обойтись.

Надо заметить, что в отличии от традиционного применения для каждого уровня абстракции не всегда порождается отдельное измерение. Если структура данных не сложная и достаточно стабильная, то применяется смещение на одном из измерений. Например, объекты тем имеют диапазон на координате Z от 1 до 99, объекты тестов – от 100 до 199, лабораторные – 200-299. Таким образом, к примеру, авторы тем могут располагаться на оси Z на уровне Z=2, оглавление тем имеет Z=5. При этом авторы тестов будут иметь координату Z=101.

Объединив две трехмерные структуры в единое пространство координат, получим новую структуру, расшифровку которой можно проиллюстрировать следующей таблицей (Табл.1):

Таблица1

Расшифровка координат			
Y	**X**	**Z**	**Расшифровка**
0	n	0	Наименование учебного предмета
0	n	1	Автор учебного предмета
0	n	2	План
0	n	3	Тема
0	n	4	Тест
n	0	0	Фамилия студента
n	0	1	Имя
n	0	2	Отчество
n	n	0	Какой предмет изучает студент

В этом гиперкубе точки со следующими координатами представляют:

- (0,100,0) – наименование учебного предмета с измерением 100;
- (0,100,1) – автор учебного предмета с измерением 100;
- (200,0,0) – фамилия студента, имеющего личный номер в системе 200;
- (200,0,2) – отчество студента под номером 200
- (200,100,0) – информации о том, что студент 200 изучает учебный предмет 100

Рассмотрим реализацию предложенного подхода для моделирования учебного плана для образовательных учреждений. Для иллюстрации преимуществ покажем процесс моделирования как для классических реляционных таблиц так и для предложенного подхода.

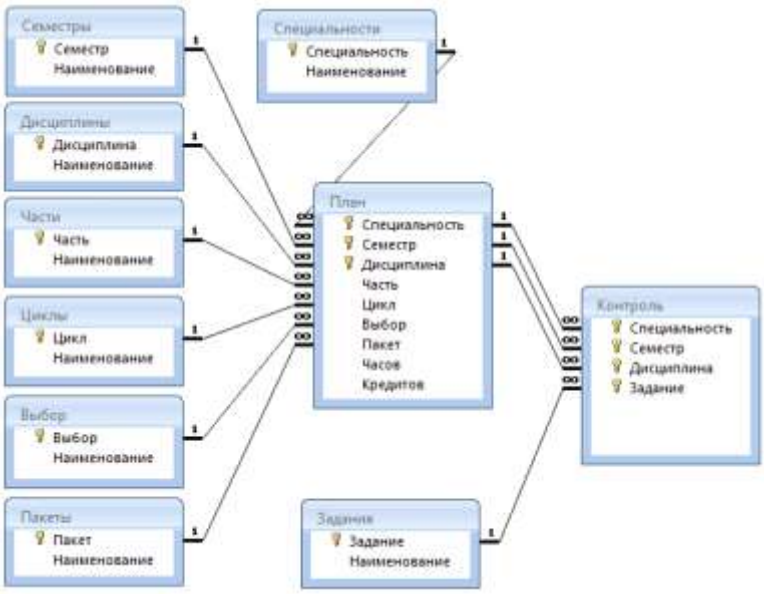


Рис. 12. Представление учебного плана в структуре данных на основе реляционных таблиц.

На рис. 12 приведена структура данных на основе реляционных таблиц. Достаточно не сложная задача породила весьма громоздкую структуру таблиц, ключей, связей. Но количество таблиц и связей - не главный недостаток наиболее распространенного сегодня подхода представления данных. На рис. 12, не видно модель учебного плана. Это, в лучшем случае, лингвистическая модель оперирующая весьма нечеткими критериями – словами.

Предлагается строить модель учебного процесса как разложение по измерениям предметной области, отраженной в учебном плане.

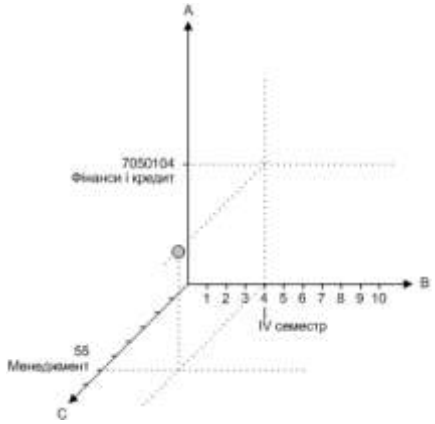


Рис.13. Привязка трех ключевых полей к координатам

Фінанси і кредит

III семестр

№ п/п	Назва дисципліни (модуль)	Всього годин	Годин			Самостійна робота	Кількість кредитів	Форма контролю
			лекції	практичні	лабораторні			
НОРМАТИВНА ЧАСТИНА								
7050104.3.738	Цикл гуманітарної підготовки							
1	738 Іноземна мова	72					2	ПМК

Рис.14. Элемент учебного плана с координатами (7050104.3.738)

Для описания всех ключевых полей учебного плана достаточно трех измерений. В словесном выражении пример на рис.13 звучит следующим образом: "В учебном плане по специальности 'Финансы и кредит' в IV семестре изучается предмет Менеджмент". В

математическом виде данная позиция (точка) плана будет иметь значение координат: (7050104, 4, 55).

Каждый элемент учебного плана имеет помимо ключевых полей еще и элементы, дополнительно описывающие свойства позиции. К таким элементам относится, к примеру: является ли позиция нормативной частью плана, к какому циклу она относится, какие формы контроля содержит и т.д.

На каждое дополнительное описание не отводится новое измерение - слишком громоздкая получилась бы структура, да и необходимости в таком решении нет. Для любого количества второстепенных описаний достаточно одной или двух координат. Как правило - достаточно одной, но в случае многовариантного описания одной и той же позиции - можно применить две координаты (рис.15).

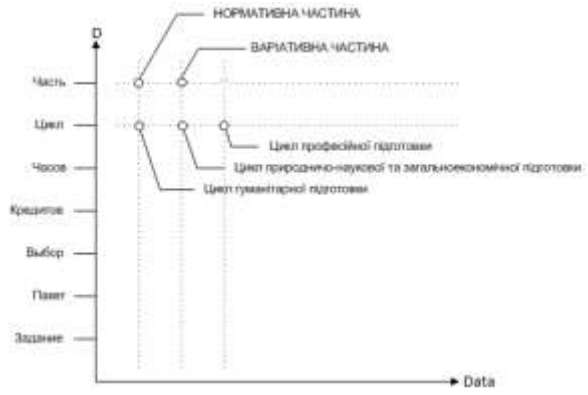


Рис.15. Двух координат достаточно для описаний по любому количеству характеристик

Рис.16. . Описывать значение может как одна ось, так и диапазон значений

Бывают случаи, когда недостаточно одного значения для описания некоторой характеристики. Например, характеристика "задание" может включать как одно задание (диплом), так и несколько (контрольная работа, зачет). В таком случае можно добавить координату, но можно сделать и проще - зарезервировать диапазон (рис.16).

Таким образом, на небольшой задаче учебного плана показаны практически все приемы организации данных.

Для представления структуры данных учебного плана по предложенному подходу понадобится только одна таблица (рис. 17).

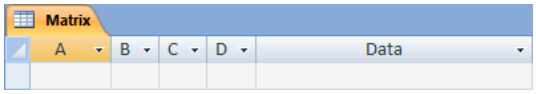
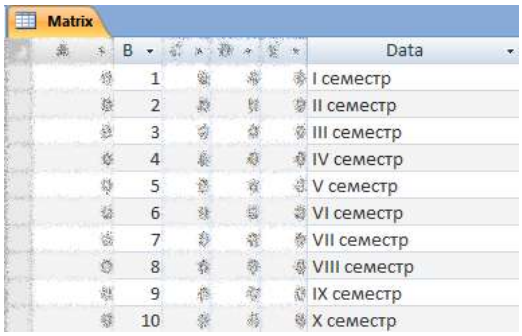


Рис.17. Таблица [Matrix] *Рис.18. Оцифровываем ось A*

Измерение A (рис. 18) содержит координаты наименования специальности. Измерение B (рис.19) содержит координаты названий семестров. Измерение C (рис.20) содержит координаты названий учебных предметов. Измерение D содержит дополнительные данные по основному объекту задачи: диапазоны для указания нормативных или вариативных частей (D=1), циклов (D=2), дисциплин по вибору (D=5) и т.п.



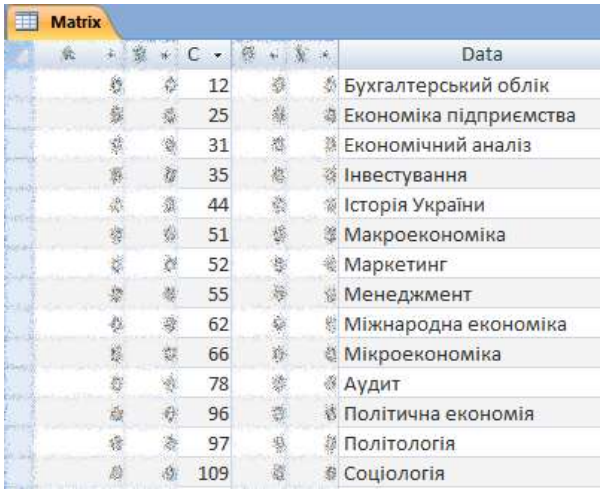


Рис.19. Таблица [Matrix] *Рис.20. Оцифровываем ось A*

На рис. 21 приведен учебный план, смоделированный при таком подходе.

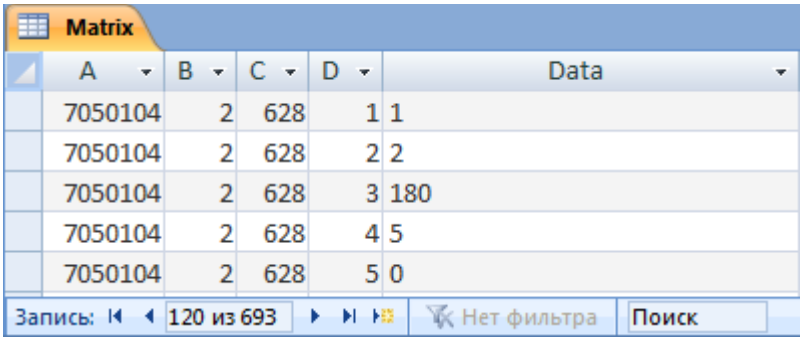


Рис. 21. Результат ввода учебного плана в систему координат A-D.

Предложенный подход моделирования позволяет организовывать схему представления предметной области задачи в виде n-мерного гиперкуба. Предложенная схема есть обобщением известных схем многомерных реляционных хранилищ данных («звезда» и «снежинка»). В тоже время, предложена схема пригодна для использования как в OLTP так и в OLAP-системах. Т.е.

возможно с помощью одного и того же программного решения работать как з задачами операционной обработки данных так и з задачами анализа данных.

С помощью предложенного подхода смоделирована и реализована задача учета дорожно-транспортных происшествий. Рассмотрим реализацию данной задачи, обращая внимание на моделирование структуры данных (рис.22).

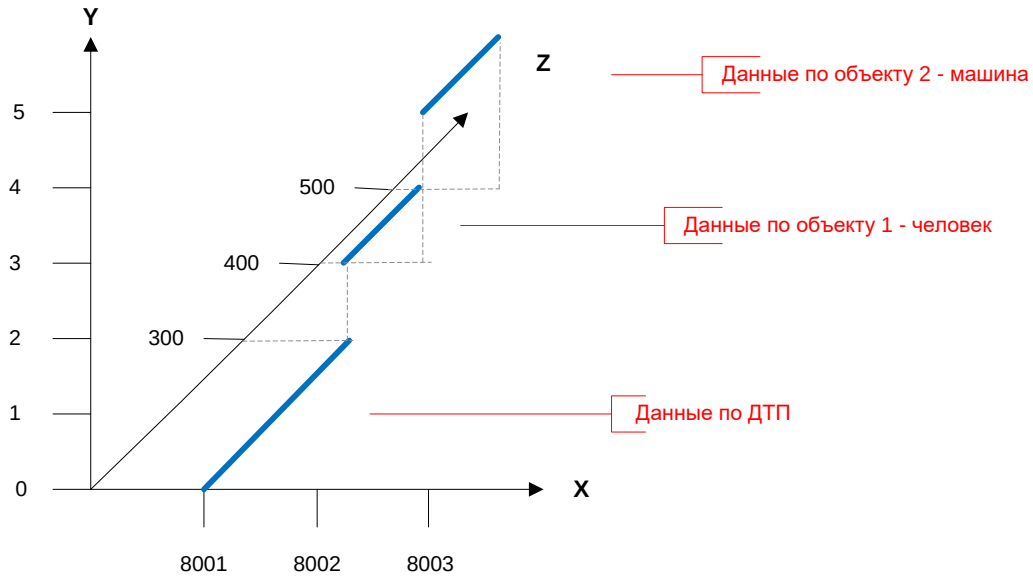


Рис. 22. Гиперкуб для задачи учета ДТП

Измерение X содержит номер ДТП - плоскость с данными по ДТП. Измерение Y - номер объекта ДТП, при Y = 0 - общие данные о самом ДТП. Измерение Z – диапазоны различных параметров. Параметры от 1 до 300 - это диапазон общих параметров по ДТП (состояние дороги, освещения и т.д.). Параметры от 400 до 499 - данные по человеку. Параметры от 500 до 599 - данные по транспорту.

Структура задачи учета ДТП с использованием классических реляционных таблиц (основные таблицы, баз справочников) представлена на рис. 23.

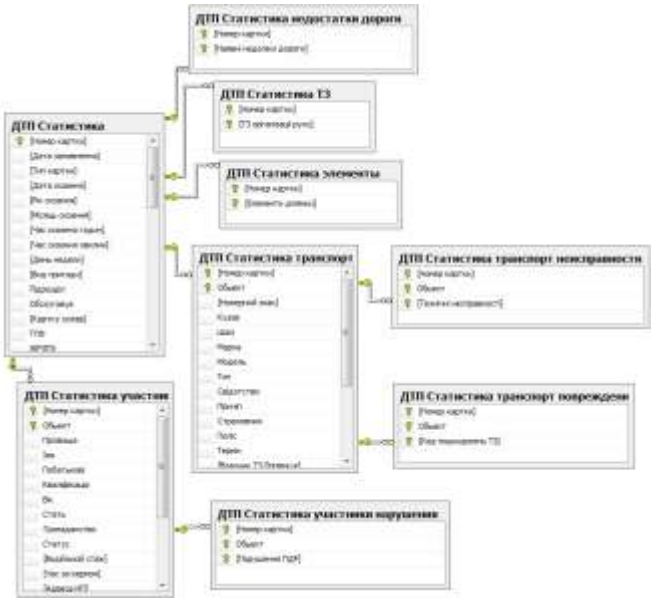


Рис. 23. Основные таблицы и связи для задачи учета ДТП

ДТП

Карточка	Обьект	Параметр	Значение
9001430	0	161	23000
9001430	0	160	23000
9001430	0	120	1
9001430	0	110	18.03.2010
9001430	0	100	9001430
9001429	0	172	EERT
9001429	0	161	23000
9001429	0	160	23000
9001429	0	130	22.01.2010
9001429	0	120	1
9001429	0	110	27.01.2010
9001429	0	100	9001429

Рис. 24. Многомерный куб задачи учета ДТП

На рис.24. видно эту же структуру (но уже всю, со справочниками) в виде матрицы. Поля X, Y, Z специально переименованы для простоты.

В задаче учета ДТП используется 88 различных параметров. При вводе ДТП могут фигурировать десятки участников ДТП и десятки транспортных средств, потому на практике для описания ситуации используется до 400 значений. Но, т.к. работа происходит с одной, довольно простой таблицей, форма ввода тоже получается простой.

Сама форма работает как чат(messenger). После ввода значения в поле, на событие on blur (сход с поля), данные отправляются на сервер, проверяются, если все в порядке - записываются в базу, и возвращаются в ячейку (т.е. то, что вы видите в ячейке секунду спустя, это не то, что вы ввели, а то, что записалось в базу). Если ошибка - в ячейку возвращается пусто и поступает красное сообщение об ошибке. Т.е. работа с базой идет постоянно, при вводе каждого значения.

Для передачи информации используется типовая посылка данных. Фактически, к каждому вводимому значению придаются 3 координаты (в какую карточку, номер объекта, параметр) и все это пишется в одну и ту же таблицу.

ВЫВОДЫ

Среди фактографических систем важное место занимают два класса: системы операционной обработки данных (OLTP-системы) и системы, ориентированные на анализ данных и поддержку принятия решений (OLAP-системы). Исторически, в OLTP системах используется классический реляционный поход для моделирования пространства данных. На практике это реляционные базы данных с огромным количеством связанных разными реляционными отношениями таблиц. Но сбор данных – не самоцель, и накопленные информационные массивы могут оказаться весьма полезными. OLTP-системы способны выполнять тривиальный анализ данных – вычислять максимальные, минимальные или средние значения атрибутов. Но из накопленных данных можно (и нужно) получать намного более глубокие сведения о функционировании организации, которая обслуживается информационной системой, так и о сфере ее деятельности. В информационных массивах можно выявлять скрытые закономерности и выводить из них правила, которым подчиняется предметная область информационной системы. Впоследствии эти правила можно использовать для стратегического планирования, принятия решений и прогнозирования их последствий.

Осознание пользы накапливаемой информации привело к появлению другого класса информационных систем – системы поддержки принятия решений (аналитические системы). Для того чтобы извлекать полезную информацию из данных, они должны быть организованы особым, отличным от принятого в OLTP-системах образом. Главным препятствием использования OLTP-систем для анализа является необходимость обработки больших информационных массивов. Чем выше степень нормализации базы данных и чем больше в ней таблиц, тем медленнее выполняется анализ. Происходит это прежде всего потому, что увеличивается число операций соединения отношений. В OLTP-системах нормализация таблиц базы данных позволяет устранить избыточность данных, уменьшив тем самым объем действий, необходимых при обновлении информации. А для анализа необходимо произвести обратный процесс.

Принципы, лежащие в основе OLAP-систем, не позволяют эффективно обрабатывать транзакции, поэтому данные, применяемые для анализа, стали выделять в отдельные базы данных, называемые хранилищами информации. Для построения хранилищ данных используют многомерные модели данных.

Предложенный в статье подход позволил объединить в одной структуре данные и дал возможность использовать их как для задач операционной обработки, так и для задач анализа. При такой структуре транзакции обладают основными свойствами: атомарности, согласованности, изолированности, долговечности. В то же время предложенная структура ориентирована на предметную область, что облегчает понимание информации конечными пользователями, и дает дополнительные возможности построения аналитических запросов.

Реализация предложенного похода для целого ряда задач (учет дорожно-транспортных происшествий для департамента ГАИ МВД Украины, унифицированная система дистанционного обучения на базе банка дистанционных курсов при МОН Украины, информационная система дистанционного обучения в Хмельницком национальном университете, и.т.д.) показала его эффективность и простоту использования. Наличие такого обобщения как n-мерное информационное пространство позволило создать универсальное программное обеспечение (engine) для работы с предложенной структурой данных.

1. Masaharu Murozumi, A Challenge To A High Transaction Volume Client/Server DB2 Data Shared OLTP System. IBM, 2000. Режим доступа: www.redbooks.ibm.com/redpapers/pdfs/redp0015.pdf

2. Going Real-Time for Data Warehousing and Operational BI. GoldenGate, 2009. Режим доступа: http://media.techtarget.com/Syndication/APP_DEVELOPMENT/GoldenGate_Software_GoingRealTime.pdf