

Московский государственный технический университет им. Н.Э. Баумана

На правах рукописи

Мин Тхет Тин

**МЕТОДИКА ФОРМИРОВАНИЯ РЕЛЯЦИОННЫХ ТАБЛИЦ
НА ОСНОВЕ ИНФОРМАЦИИ ТАБЛИЧНОГО ВИДА**

Специальность 05.13.11 – Математическое и программное обеспечение
вычислительных машин, комплексов и компьютерных сетей

ДИССЕРТАЦИЯ

на соискание ученой степени кандидата технических наук

Научный руководитель -
д.т.н., доцент, Брешенков А.В.

Москва 2015 г.

СОДЕРЖАНИЕ

СПИСОК СОКРАЩЕНИЙ	5
ВВЕДЕНИЕ	6
1. ИССЛЕДОВАНИЕ ЗАДАЧ ПОСТРОЕНИЯ МЕТОДИКИ ФОРМИРОВАНИЯ РЕЛЯЦИОННЫХ ТАБЛИЦ НА БАЗЕ ЗАПОЛНЕННЫХ НЕРЕЛЯЦИОННЫХ ТАБЛИЦ	20
1.1. Обзор существующих средств формирования реляционных таблиц в контексте решаемой проблемы.....	20
1.1.1. Основы современной методологии проектирования реляционных баз данных в контексте решаемой проблемы.....	20
1.1.2. Модель реляционных таблиц (РТ).....	22
1.1.3. Ключевые поля в реляционных таблицах.....	24
1.1.4. Нормализация и семантическое моделирование.....	26
1.2. Определение понятия расширенной информации табличного вида (ИТВР) и причины построения методики преобразования ИТВР в реляционное представление.....	28
1.2.1 Определение понятия ИТВР.....	28
1.2.2. Причины и проблемы построения методики автоматизированного преобразования ИТВР в расширенные РТ (РТР)	35
1.3 Анализ применимости современных теоретических и практических разработок.....	38
1.4. Постановка задачи разработки метода формирования РТР на основе использования ИТВР.....	39
Выводы по главе 1.....	43
2. МЕТОД ПРЕОБРАЗОВАНИЯ ЗАПОЛНЕННЫХ ИТВР В РТР	44
2.1. Модели объектов исследования.....	44
2.1.1. Модель РТР.....	44

2.1.2. Модель ИТВР.....	47
2.2. Задача приведения значений столбцов ИТВР к одному типу.....	52
2.2.1. Типы полей в реляционных таблицах.....	52
2.2.2. Преобразование значений атрибутов заполненных таблиц к одному типу.....	53
2.3. Исключение повторяющихся строк в ИТВР.....	55
2.3.1. Типы повторяющихся строк	55
2.3.2. Исключение повторений строк.....	59
2.4. Избавление от сложных заголовков и подзаголовков.....	60
2.4.1. Задача избавления от подзаголовков.....	60
2.4.2. Исключение внутренних подзаголовков.....	63
2.4.3. Способ избавления от сложных заголовков и подзаголовков.....	70
Выводы по главе 2.....	84
3. МЕТОД НАЗНАЧЕНИЯ КЛЮЧЕЙ В ИТВР.....	84
3.1. Задача формирования ключей в ИТВР.....	84
3.2. Алгоритмы назначения первичных ключей в ИТВР.....	87
3.2.1. Неформальные алгоритмы назначения первичных ключей в ИТВР.....	87
3.2.2. Формальные алгоритмы назначения первичных ключей в ИТВР.....	94
3.3. Алгоритмы назначения внешних ключей в ИТВР.....	101
3.3.1. Неформальные алгоритмы назначения внешних ключей в ИТВР.....	101
3.3.2. Формальные алгоритмы назначения внешних ключей в ИТВР..	106
Выводы по главе 3.....	108
4. МЕТОДИКА ФОРМИРОВАНИЯ РТР НА ОСНОВЕ ИСПОЛЬЗОВАНИЯ ЗАПОЛНЕННЫХ ИТВР.....	109
4.1. Формулировка проблемы формализации методики.....	109
4.2. Модель методики в операторной форме.....	112

4.3. Анализ методики для обнаружения дефектов ее представления.....	128
4.4. Анализ методики для обнаружения дефектов ее функционирования.....	143
Выводы по главе 4.....	147
ОСНОВНЫЕ РЕЗУЛЬТАТЫ РАБОТЫ.....	149
ЛИТЕРАТУРА.....	150
ПРИЛОЖЕНИЕ 1. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ МЕТОДИКИ ФОРМИРОВАНИЯ РТР НА ОСНОВЕ ИТВР.....	160
ПРИЛОЖЕНИЕ 2. АКТ ВНЕДРЕНИЯ И ИСПОЛЬЗОВАНИЯ РЕЗУЛЬТАТОВ ДИССЕРТАЦИОННОЙ РАБОТЫ	192

СПИСОК СОКРАЩЕНИЙ

БД – база данных

РБД – реляционная база данных

РМД - реляционная модель данных

СУБД – система управления БД

ИТВ – информация табличного вида

ИТВР - расширенная модель ИТВ

РТ – реляционные таблицы

РТР - расширенная модель РТ

ЭТ – электронные таблицы

ПК – первичный ключ

ПО – программное обеспечение

ОС – операционная система

ВВЕДЕНИЕ

В настоящее время трудно переоценить значение компьютерных информационных систем. А коль скоро базы данных (БД) являются ядром информационных систем, в полной мере это относится и к БД. Это детально и убедительно доказывается в соответствующей научно-популярной и технической литературе. В частности об этом говорят специалисты в области БД [1–23]. Более того, в паспорте специальности 05.13.11 (Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей) отмечается:

- необходимость разработки и исследования в области программных средств организации и управления обработкой данных и знаний;
- необходимость создания прикладного математического обеспечения, программных средств автоматизации разработки программ;
- актуальность разработки программных средств обработки данных и знаний в ВМ, ВК и КС;
- актуальность разработки методы проектирования систем управления базами данных (СУБД) и базами знаний (СУБЗ), в том числе распределенными СУБД и СУБЗ.

Даже из незначительной информации, которую можно получить из названия диссертации, можно сделать вывод о том, что она посвящена решению названных проблем. Действительно, речь идет о преобразовании информации в данные, кодировании информации в виде данных. Естественно, для этого потребуются модели информации табличного вида и модели таблиц данных. Кроме того необходима разработка математического, информационного, программного, методического и других видов обеспечений, ориентированных на преобразование информации в данные.

Собственно понятие информации – глобальное и охватывает все сферы человеческой деятельности от вербального общения между людьми до работы в Интернете [25]. Далеко не всякую информацию можно представить

в виде данных. Ведь данные – это информация, представленная в регламентированном виде. К сожалению, не всю информацию можно строго регламентировать. Поэтому в работе рассматривается информация табличного вида (ИТВ), модель которой предложена в работах [37, 53] и ее расширение – ИТВР. Представление такого рода информации близко к представлению данных в БД, и поэтому в принципе процесс преобразования ИТВ в формат БД можно формализовать. Но ИТВ по ряду признаков существенно отличается от данных. Суть проблемы в том и состоит, чтобы разработать способы, алгоритмы, методику и средства, которые позволят исключить эти отличия.

Для начала необходимо дать хотя бы неформальные определения ИТВ и представление данных в БД. ИТВ – это информация, которая воспринимается пользователем как двумерные таблицы.

По сути, это могут быть ведомости, прайс-листы, словари, списки и многое другое. Главная задача разработчиков такого рода таблиц – это обеспечение минимальной субъективной сложности восприятия информации.

В работах [26, 37, 53] введено и определено понятие ИТВ. Определены следующие свойства ИТВ. «

- Это информация, которая интуитивно воспринимается ее потребителями, как таблицы.
- В табличном представлении информации нередко отсутствуют разделители строк и разделители столбцов.
- Элементы данных нередко размещаются в нескольких строках.
- Типы элементов данных, соответствующих одному столбцу, могут различаться.
- Заголовки информации табличного вида могут включать в себя подзаголовки.

- Заголовки и/или подзаголовки одноименного столбца нередко размещаются в нескольких строках. ».

На момент времени определения ИТВ оно отражало большинство черт реальных ИТВ. С тех пор прошло около 10-и лет. Ситуация существенно изменилась. В частности, появились новые, инструментальные средства представления ИТВ, которые отражают все возрастающие потребности пользователей и прогресс в области технических и программных средств. Это новые версии: электронных таблиц, текстовых редакторов, текстовых процессоров, HTML средств, систем обработки статистической информации и др.

В связи с этим появилась возможность и потребность представления более "экзотических" ИТВ, которые наряду с перечисленными могут обладать следующими особенностями. В расширенном понятии ИТВ в ИТВР:

- допустимо отражение семантики данных посредством цвета, фона, шрифта и т.п.;
- допустимо повторное использование заголовков и подзаголовков (суть их определяется посредством места в таблице, цвета, фона, шрифта и т.п.);
- возможно использование значений любых доменов атрибутов в качестве подзаголовков;
- допустимо использование комбинированных подзаголовков;
- допустимо использование одноименных доменов;
- необязательно наличие первичных ключей;
- отсутствуют внешние ключи.

Таким образом понятие ИТВ расширено и для того чтобы это подчеркнуть в дальнейшем, при необходимости, будем использовать аббревиатуру ИТВР.

В данном случае никакой речи о регламентации информации не может быть и речи. Проблема и состоит в преобразовании нерегламентированной информации к регламентированному виду. А регламентированный вид – это формат БД. При преобразовании ИТВР необходимо учитывать необходимость обработки всех форм их представления, свойства которых перечислены выше.

По форме ИТВР может быть представлено на бумаге, в формате текстовых редакторов, в формате текстовых процессоров, в формате электронных таблиц и во многих других форматах. В связи с этим возникает вторая проблема – проблема преобразования форматов.

Естественно задаться вопросом, а нужны ли преобразования ИТВР в таблицы БД. Собственные исследования, работа с экспертами, участие в разработках показали, что, с одной стороны, БД исключительно редко создаются на пустом месте (чаще всего имеются значительные объемы информации вида ИТВР), а, с другой стороны, к настоящему времени накопилось множество ИТВР, которое просто необходимо обрабатывать средствами современных систем управления базами данных (СУБД).

И, к сожалению, в настоящее время нет теоретических и практических разработок, которые могли бы в полном объеме решить проблемы преобразования ИТВ в формат БД. Как сказал один из основоположников теории проектирования БД Дейт К. Дж.: “проектирование БД – это скорее искусство, чем наука” [12]. А преобразование ИТВ в формат БД – это важнейшая проблема, стоящая перед проектировщиками БД. Причем эта проблема еще менее исследованная область знаний по сравнению с традиционными методами проектирования БД.

Достаточно большой объем работы в области проектирования БД на основе использования ИТВ проделал Брешенков А.В. [4–7, 26–37]. Однако, несмотря на глубокую теоретическую и практическую проработку проблемы, в работах Брешенкова А.В. не рассматриваются некоторые задачи, которые

необходимо решать в процессе преобразования ИТВ в формат БД. В частности:

- рассмотрены не все возможные виды подзаголовков в ИТВ;
- не рассмотрены гибридные подзаголовки;
- в качестве атрибутов, которые входят в первичный ключ, анализировалось не более 2-х;
- связи между таблицами рассмотрены для ключевых полей, включающих только один атрибут;
- не проанализировано одно из требований минимальности первичного ключа: никакая часть первичного ключа не должна быть уникальной;
- не проведено детальное исследование по поводу выявления внешних ключей в ИТВ.

Все это, конечно, не умаляет научные и практические достоинства работ [4–7, 26–37]. Более того, эти работы послужили хорошей базой для дальнейших разработок и, в частности, разработок, выполненных в диссертации.

Для того чтобы сделать акцент на том, что в работе ИТВ представлено более детально, чем в [4–7, 26–37] введено понятие ИТВР - расширенная информация табличного вида. При этом под расширением понимается то, что наряду с известными характеристиками ИТВ учитываются и их дополнительные характеристики, которые обусловили необходимость решения задач перечисленных выше. Модели ИТВ и ИТВР рассматриваются во 2-й главе.

Теперь под ИТВ будем понимать объекты, обладающие свойствами и ИТВ и ИТВР. Там где это оправданно будет использоваться аббревиатура ИТВР.

Естественен вопрос – какую концептуальную модель БД использовать в качестве целевой при преобразовании ИТВР? К числу наиболее популярных концептуальных моделей БД относятся реляционные модели. Это связано с их наглядностью, простотой восприятия и реализации, наличием

формального аппарата для представления и обработки данных. Несмотря на применение предшествующих моделей данных (иерархической и сетевой) и, несмотря на бурное развитие новых моделей данных (постреляционной, многомерной и объектно-ориентированной), реляционные БД (РБД) в настоящее время занимают в мире лидирующее положение и подавляющее большинство существующих БД и СУБД построены в соответствии с реляционным подходом [38]. Как указывают даже сторонники новых концептуальных моделей данных: «Реляционные данные существуют уже около 30 лет. За это время вспыхивало несколько революций, которые должны были положить конец реляционным хранилищам. Конечно, ни одна из этих революций не состоялась, и ни одна из них ни на йоту не поколебала позиции реляционных БД» [44].

В работах [37, 53] предложена модель реляционных таблиц – РТ. Но она, как показано в диссертации, не удовлетворяет всем требованиям к реляционным таблицам. Поэтому в диссертации введено понятие и соответствующая расширенная модель реляционных таблиц – РТР. Ее основное отличие от РТ в том, что она отражает свойства ИТВР которые не допустимы в РТ.

За последние годы выполнен значительный объем научных исследований, посвященных проектированию РБД. В их числе работы Е. Ф. Кодда [19-23], К. Дж. Дейта [11-13, 39-44], Гэри и Джэймса Хансенов [45], Дж. Ульмана [47-48], Р. Р Чена. [177], Р. Стивенса, Р. Плю [154], Д. Энсора [112], Ю.В. Тихомировой[48-49], Ю.А. Григорьева, Г.И. Ревункова [50-51], Т.С. Карповой [53], А.В. Брешенкова [4-7, 26-37, 53], Баранчикова А.И. [101] и других.

Однако, несмотря на значительные успехи в области проектирования РБД большинство задач проектирования формализовать не удастся. Это, в частности, отмечает и Дейт К. Дж. [13]. Такое положение вещей связано в большинстве случаев с тем, что в процессе проектирования БД в основном используются не сами данные или

информация, преобразуемая в данные, а предполагаемые схемы отношений. В процессе эксплуатации БД может оказаться, что предположения ошибочны и схемы отношений сформированы ошибочно. «Действительно, не зная содержимого таблиц, а только отталкиваясь от их схемы отношений, далеко не всегда возможен правильный и оптимальный выбор ключевых полей, выявление функциональных зависимостей, решение вопросов нормализации, обоснованное формирование связей между таблицами. Ведь все эти вопросы решаются неформально на основании предполагаемого содержимого таблиц с данными, которых еще нет» [53]. В связи с этим БД приходится нередко перепроектировать, что связано с большими издержками различного характера.

Ситуация кардинально меняется, когда в качестве исходных сведений для проектирования используются существующая информация.

Ее анализ можно формализовать и в конечном итоге принять лучшие решения. В особенности это касается ИТВР.

С другой стороны, значительная часть информации, в том числе и информация табличного вида, находится вне баз данных и даже вне ЭВМ, хотя потребители этой информации, как правило, очень заинтересованы в возможности использования многочисленных возможностей БД [5,6].

Эти два положения и определяют актуальность разработки методов, алгоритмов, методики и средств формирования реляционных таблиц на основе существующих ИТВР.

Конечно, традиционная методология проектирования РБД является мощным, а на сегодня лучшим подходом к разработке реляционных БД, использующих в качестве концептуальной модели реляционную модель данных. Поэтому было бы просто неразумно в данной работе не использовать инструмент создания целостных, непротиворечивых и неизбыточных систем.

С другой стороны, неразумно не воспользоваться фактами повсеместного использования ИТВР и не решать задачи проектирования БД

на основе анализа имеющейся информации. В связи с этим и возникает проблема теоретических и практических разработок, ориентированных на разработку способов, алгоритмов, методики и средств формирования реляционных таблиц на основе информации табличного вида.

В работе [53] предложена методика автоматизированного формирования РТ на основе заполненных ИТВР. Однако в связи с введением в диссертации новых, концептуальных понятий и моделей ИТВР и РТР появилась необходимость в разработке нового метода преобразования ИТВР в РТР, нового метода назначения ключевых полей в ИТВР, новой методики автоматизированного формирования РТР на основе заполненных ИТВР, которые отражают концептуальные свойства современных ИТВ и современные потребности по поводу их преобразования в реляционное представление.

Проблема заключается в отсутствии полного комплекса способов, алгоритмов, методики и средств, ориентированных на преобразование ИТВР в расширенные реляционные таблицы (РТР).

В работе предлагаются методика преобразование ИТВР в РТР, в основе которого лежат модели ИТВР и РТР, методы и алгоритмы решения проектных задач преобразования, соответствующие лингвистические и программные средства.

Предмет исследования: модели, методы и методика проектирования реляционных таблиц на базе ИТВР, а также компоненты математического, лингвистического, информационного и программного обеспечений методики преобразования ИТВР в реляционные таблицы.

Целью работы является разработка в рамках методики теоретических и практических основ формирования таблиц РБД на базе ИТВР, улучшение качественных и количественных характеристик существующих средств и алгоритмов решения задач формирования РБД на основе ИТВР для:

- автоматизированного преобразования заполненных ИТВР, соответствующих реальным представлениям о информации табличного

вида, в РТР, соответствующих полным представлениям о реляционных таблицах;

- автоматизированного формирования связей между преобразованными таблицами ИТВР;

- автоматизированного назначения первичных ключей в ИТВР, в связи с их определением, как неотъемлемой компоненты РТР;

- автоматизированного назначения внешних ключей в ИТВР, в связи с их определением, как неотъемлемой компоненты РТР;

В соответствии с поставленной целью в диссертации решаются следующие задачи:

1. Обоснование необходимости и разработка расширенных моделей РТ и ИТВ – РТР и ИТВР, которые являются развитием и дополнением соответствующих моделей, предложенных в работах [37, 53], и отражают реальное положение вещей в представлении данных табличного вида.

2. Разработка метода автоматизированного преобразования заполненных ИТВР в РТР, в котором задействованы модели ИТВР и РТР. Метод должен обеспечить автоматизированное решение задач, решаемых ранее вручную. К этим задачам относятся: исключение внутренних подзаголовков в ИТВР и их преобразование в домены; преобразование доменов в ИТВР, в которых используются значения в качестве подзаголовков; исключение комбинированных подзаголовков в ИТВР и их преобразование в новые таблицы и домены; исключение пустых записей и доменов в ИТВР; исключение одноименных доменов в ИТВР.

3. Разработка метода автоматизированного назначения ключевых полей в заполненных ИТВР, в котором задействованы модели РТР и ИТВР. Метод должен обеспечить автоматизированное решение задач, решаемых ранее вручную. К этим задачам относятся: назначение первичных ключей, как неотъемлемой компоненты РТР; назначение внешних ключей, как неотъемлемой компоненты РТР.

4. Разработка методики автоматизированного формирования РТР на основе анализа заполненных ИТВР. В методике должны быть задействованы модели ИТВР и РТР, метод преобразования ИТВР в РТР и метод назначения ключевых полей в ИТВР, алгоритмы и процедуры, которые обеспечивают эффективное решение задач, решаемых ранее вручную.

5. Разработка формального описания методики автоматизированного формирования РТР на основе анализа заполненных ИТВР. Исключение, с помощью формального описания, принципиальных ошибок в описании методики и принципиальных ошибок в ее функционировании.

Под эффективностью здесь понимаем – решение задач бездефектного преобразования ИТВР в РТР с минимальной трудоемкостью и минимальными временными затратами.

При описании моделей ИТВР и РТР, а также при разработке методов и методики преобразования ИТВР в РТР использована реляционная алгебра, исчисление предикатов, теория множеств, теория алгоритмов, аппарат сетей Петри.

Структура диссертации соответствует списку сформулированных задач, включает в себя описание предложенных способов, средств и алгоритмов, которые задействованы в методике преобразования заполненных таблиц ИТВР в реляционные таблицы.

В первой главе выполнен аналитический обзор традиционного подхода формирования реляционных таблиц, сформулированы его достоинства и недостатки. Введено понятие расширенной информации табличного вида. Сформулированы мотивы и выполнен анализ проблем разработки методики преобразования ИТВР в РТР в рамках развития методологии проектирования РБД, предложенной в работах [37, 53]. Выполнена постановка задачи разработки методики автоматизированного преобразования ИТВР в РТР. Определен состав алгоритмов и средств,

разрабатываемых в рамках методики проектирования РТР на основе существующей информации табличного вида.

Во второй главе разработан метод преобразования таблиц ИТВР к реляционному виду. В рамках метода предложены алгоритмы преобразования значений атрибутов каждого домена к одному типу в ИТВР, исключения дублирования строк в ИТВР, избавления от сложных атрибутов в ИТВР, исключения подзаголовков в ИТВР, исключения гибридных подзаголовков в ИТВР, исключения дублирования имен заголовков и подзаголовков в ИТВР.

В третьей главе разработан метод назначения ключевых полей в заполненных реляционных таблицах, который обеспечивает формирование первичных и внешних ключей и рассматривает их в качестве неотъемлемой компоненты определения РТР.

В четвертой главе разработана методика формирования реляционных таблиц на основе информации табличного вида. В рамках методики задействованы модели РТР и ИТВР, методы и алгоритмы, разработанные в предыдущих главах диссертации. Разработано формальное описание методики. На его основе выявлены и исключены принципиальные ошибки в описании методики и принципиальные ошибки ее функционирования.

В заключении представлены основные результаты работы.

В приложениях приводятся программная реализация разработанной методики формирования реляционных таблиц на основе информации табличного вида, акт внедрения результатов диссертационной работы.

Научную новизну работы определяет концепция и теоретические основы формирования РТР на базе ИТВР, которые воплощены в методику проектирования РТР.

Основные научные результаты, выносимые на защиту:

- Разработана расширенная модель реляционных таблиц (РТР) которая обладает с точки зрения проектирования РБД концептуальными

особенностями. В частности, в модели отражены следующие ее свойства, не рассмотренные в существующих моделях реляционных таблиц: в РТР недопустимы внутренние заголовки; в РТР недопустимо использование значений доменов атрибутов в качестве подзаголовков; в РТР не должны использоваться комбинированные подзаголовки; в РТР недопустимо использование одноименных доменов; в РТР недопустимы пустые записи и домены; в РТР обязательно наличие первичных ключей; в РТР могут быть задействованы внешние ключи; РТР должна удовлетворять традиционным требованиям к реляционным таблицам.

- Разработана расширенная модель информации табличного вида (ИТВР), которая отражает представление таблиц в текстовых процессорах, HTML файлах, в текстовых файлах, в электронных таблицах и обладает с точки зрения проектирования РБД концептуальными особенностями. В частности, в модели учтены следующие свойства, которые не рассмотрены в существующих моделях ИТВ: в ИТВР допустимы внутренние подзаголовки; в ИТВР возможно использование значений доменов атрибутов в качестве подзаголовков, в ИТВР допустимо использование комбинированных подзаголовков; в ИТВР допустимы пустые записи и домены; в ИТВР допустимо использование одноименных доменов; в ИТВР необязательно наличие первичных ключей, в ИТВР отсутствуют внешние ключи; ИТВР отражает существующие требования к ИТВ.

- Разработан метод преобразования ИТВР в РТР, в котором задействованы предложенные модели ИТВР и РТР, спроектированные и реализованные алгоритмы и процедуры. Суть его заключается в автоматизированном преобразовании объектов, соответствующих предложенной модели ИТВР к объектам, соответствующим предложенной модели РТР. Метод позволяет в автоматизированном режиме решать задачи преобразования ИТВР в РТР, которые ранее решались вручную. В результате чего, как показали экспериментальные исследования, для ИТВР средней сложности (степень 20-30 и мощность 1000-10000) время преобразования

сократилось в десятки раз. Кроме того исключены ошибки преобразования, которые неизбежны при использовании традиционных способов.

- Разработан метод назначения ключевых полей в заполненных таблицах ИТВР, в котором задействованы предложенные модели ИТВР и РТР, спроектированные и реализованные алгоритмы и процедуры. Суть его заключается в автоматизированном назначении объектам, соответствующих модели ИТВР, первичных и внешних ключей в соответствии с моделью РТР. Метод позволяет в автоматизированном режиме решать задачи назначения ключевых полей, которые ранее решались вручную. В результате чего, как показали экспериментальные исследования, для ИТВР средней сложности время назначения ключевых полей сократилось в десятки раз. Кроме того исключены ошибки назначения ключевых полей, которые неизбежны при использовании традиционных способов.

- Разработана методика автоматизированного преобразования существующей информации табличного вида, которая соответствует модели ИТВР, в их реляционное представление, которое соответствует модели РТР. В методике задействованы модели ИТВР и РТР, разработанные методы, спроектированные и реализованные алгоритмы и процедуры. Методика позволяет решать в автоматизированном режиме задачи формирования РТР на основе использования заполненных ИТВР, которые ранее решались вручную. В результате чего, как показали экспериментальные исследования, для ИТВР средней сложности время преобразования сократилось в десятки раз. Кроме того исключены ошибки преобразования, которые неизбежны при использовании существующих методик.

Обоснованность основных научных положений, которые изложены в диссертации, обусловлена корректным использованием современного математического аппарата. Достоверность положений работы и выводов работы подтверждена результатами внедрения в учебный процесс МГТУ им. Н.Э. Баумана.

Научные и практические результаты, полученные в работе, доведены до использования. В частности - в учебном процессе. Они представляют интерес в области проектировании реляционных баз данных. Методика, методы и алгоритмы реализованные в работе могут быть использованы при решении задач проектирования РБД на основе использования заполненных ИТВР.

Содержание основных разделов работы изложено и получило одобрение:

- на Российских научно-технических конференциях и семинарах (2011 – 2014 г.г.);
- на заседании кафедры “Компьютерные системы и сети” МГТУ им. Н.Э. Баумана.

Научные положения, идеи и практические результаты исследований составляют оригинальное направление в области проектирования реляционных баз данных в различных областях человеческой деятельности.

По результатам выполненных исследований опубликовано 11 научных работ.

Диссертация состоит из введения, четырех глав и заключения, которые опубликованы на 159-и страницах машинописного текста, содержит 48 рисунков, 20 таблиц, список литературы из 101-го наименований и 2-х приложений.

1. ИССЛЕДОВАНИЕ ЗАДАЧ ПОСТРОЕНИЯ МЕТОДИКИ ФОРМИРОВАНИЯ РЕЛЯЦИОННЫХ ТАБЛИЦ НА БАЗЕ ЗАПОЛНЕННЫХ НЕРЕЛЯЦИОННЫХ ТАБЛИЦ

В первой главе выполнен краткий анализ подхода формирования реляционных таблиц в рамках современной методологии проектирования реляционных баз данных, сформулированы его достоинства и недостатки. Приведены реальные примеры ИТВ, которые определяют необходимость расширения существующей модели ИТВ. Введено новое понятие ИТВР и сформулированы мотивы разработки методики преобразования ИТВР в реляционное представление. Сформулированы мотивы и проблемы разработки методики формирования реляционных таблиц на основе использования ИТВР. Выполнен анализ применимости последних теоретических и практических разработок для решения проблемы преобразования ИТВР. Выполнена постановка задачи разработки способа формирования РТР на основе использования ИТВР. Выполнена постановка задачи разработки нового способа назначения ключевых полей в ИТВР.

1.1 Обзор существующих средств формирования реляционных таблиц в контексте решаемой проблемы

1.1.1 Основы современной методологии проектирования реляционных баз данных в контексте решаемой проблемы

Проектирование РБД в соответствии с традиционной методологией включает в себя 4 этапа [11–13]. Как правило, выделяют следующие основные этапы проектирования: формулировка и анализ требований, инфологическое проектирование, даталогическое проектирование и физическое проектирование.

Формулировка и анализ требований подразумевает определение целей разработки БД, выделение основных информационных потоков, экспертированием проекта и другими мероприятиями, которые реализуют детальный анализ рассматриваемой предметной области. Основная задача данного этапа - формулировка требований к создаваемой РБД. Этот этап заканчивается техническим заданием на проектируемую РБД [45-47].

При наличии информации табличного вида ТЗ в значительной степени уже сформулировано. Действительно, в общих чертах определены состав данных и их структура, известно смысловое назначение информации, имеются сведения о связях между сущностями. В связи с этим процесс формирования ТЗ при наличии ИТВР упрощается.

Инфологическое проектирование ориентировано на построение модели предметной области. В частности, выделяются сущности, формируются и оптимизируются локальные представления, назначаются первичные и внешние ключи, строятся диаграммы сущность – связь, таблицы приводятся к нормальным формам, формируются связи между таблицами. На этом этапе проектирования РБД разработчик абстрагируется от инструментальных средств их разработки. От таких как Oracle, Clarion, Access и других. В этом и состоит одно из основных достоинств теории проектирования РБД – разделение логического и физического уровней разработок [45-47] .

Именно на этом этапе проектирования РБД в наибольшей мере проявляется достоинство подхода формирования РТР на основе использования ИТВР. Оно обусловлено тем, что проектные задачи решаются не на основе анализа гипотетических схем отношений, а на основе анализа реальной информации, а это обеспечивает возможность формализации проектных процедур [53].

Даталогическое проектирование связано с построением модели предметной области в терминах языков инструментальных средств разработки БД. При этом в качестве модели данных используются сами

данные. Разработка датологической модели осуществляется на базе инфологической модели.

При проектировании РТР на основе существующих ИТВР инфологический и датологический этапы проектирования можно совместить.

Физическое проектирование ориентировано на осуществление связи между датологической моделью и средой хранения данных. Этот этап связан с выбором носителя данных, выбором внутренних форматов хранения данных, выбором методов доступа к данным, выбором методов сжатия данных. [1].

На этом этапе несущественно, использовалась ли ИТВР при проектировании РТР.

1.1.2 Модель реляционных таблиц (РТР)

РМД рассматривается практически во всех работах, посвященных проектированию РБД. В частности она обсуждается в работах [4–9], [11–13], [19–23]. Она является целевой моделью в контексте темы диссертации. В связи с этим ее необходимо обсудить. К числу удачных определений основной компоненты РМД можно отнести определение, представленное в работе [37]. По определению автора: « реляционная модель данных (РМД) некоторой предметной области представляет набор отношений, изменяющихся во времени ». Для однозначного понимания терминологии и сути данной работы оправдано привести таблицу элементов реляционной модели данных (таблица 1.1), которая представлена в работе [37]. «

Таблица 1.1

Элемент РТ	Форма его представления
Отношение	Таблица
Схема отношения	Заголовки столбцов таблицы
Кортеж	Запись таблицы
Сущность	Описание свойств объекта
Атрибут	Заголовок домена таблицы
Домен	Множество значений атрибута
Значение атрибута	Значения поля в строке
Первичный ключ	Уникальные атрибуты
Тип данных	Тип элементов таблицы

» [37].

Несмотря на это детальное представление элементов РМД в таблицу можно добавить описательный атрибут – свойства и ограничения атрибута, внешний ключ – атрибут, используемый для обеспечения уникальности записей и для связи между таблицами.

« Основным понятием РМД является отношение, которое представляет собой подмножество декартового произведения доменов D_1, D_2, \dots, D_k вида:

$$D = D_1 \times D_2 \times \dots \times D_k, \text{ где}$$

$$D_1 = (d_{11}, d_{12}, \dots, d_{1i}, \dots, d_{1m1})$$

$$D_2 = (d_{21}, d_{22}, \dots, d_{2i}, \dots, d_{2m2})$$

...

$$D_k = (d_{k1}, d_{k2}, \dots, d_{ki}, \dots, d_{kmk})$$

Домен – множество элементов, типы которых могут не совпадать.

Отношение R представляется следующим образом:

$$R \subseteq D = D_1 \times D_2 \times \dots \times D_k \text{ » [1].}$$

Следует отметить то, что в таблице 1.1 описаны скорее элементы РТ (составной части РМД), а не РМД. РМД отражает связи между таблицами. А

в рамках темы данной работы как раз и представляет основной интерес именно РТ.

« РТ соответствует отношению из k атрибутов, должна удовлетворять следующим свойствам:

- каждая строка представляет собой кортеж из k значений, принадлежащим k столбцам;
- каждый кортеж содержит точно одно значение (соответствующего типа) для каждого атрибута;
- порядок столбцов фиксирован $(1, 2, \dots, k)$;
- порядок строк произволен;
- любые две строки различаются хотя бы одним элементом;
- строки и столбцы могут обрабатываться в любой последовательности, определяемой применяемыми операциями обработки;
- атрибуты не должны дублироваться » [18-23].

Как правило, ИТВР, свойства которой рассмотрены во введении, не удовлетворяет большинству из перечисленных требований. В изменении такого положения вещей и состоит основная задача работы.

1.1.3. Ключевые поля в реляционных таблицах

В работах, посвященных теории проектирования РБД, в частности в работах [11-13], дается определение ключей, обосновывается их необходимость, формулируются требования к ним и определяются свойства внешних ключевых полей.

Ряд специалистов в области реляционных баз данных не включают в качестве требований к РТ наличие первичных ключей. Это оправданно лишь отчасти.

Действительно. Если эту проблему рассматривать в теоретическом аспекте, то отдельные первичные и внешние ключи могут быть назначены в процессе нормализации таблиц. Если эту проблему рассматривать в практическом аспекте, большинство инструментальных средств,

ориентированных на проектирование РБД, позволяют описывать реляционные таблицы без указания первичных и внешних ключей.

В связи с этим ключевые поля могут быть просто не назначены, а это недопустимо. Ведь одно из центральных требований к РТ звучит следующим образом: любые две строки таблицы должны различаться хотя бы одним элементом. А в этом и состоит одно из назначений первичных ключей.

К основным требованиям к первичным ключам относятся их уникальность и минимальность.

В соответствии с требованием уникальности первичного ключа ни одно значение, соответствующее атрибуту первичного ключа не должно повторяться. Если первичный ключ включает в себя несколько атрибутов, то это требование звучит следующим образом – ни один кортеж значений соответствующий всем атрибутам, входящим в первичный ключ, не должен дублироваться.

В соответствии с требованием минимальности первичного ключа, с одной стороны, суммарная длина атрибутов, входящих в первичный ключ, должна быть минимальной, а, с другой стороны, ни один атрибут, входящий в первичный ключ, не должен быть уникальным (в том смысле, что его значения могут повторяться). Второй критерий минимальности первичного ключа в работе [53], в которой обсуждается проблема назначения ключевых полей в ИТВ, не рассматривается.

При наличии только схемы отношения сложно выбрать атрибуты, удовлетворяющие данным условиям. При этом легко ошибиться, т.к. после заполнения таблицы реальная ситуация может отличаться от предполагаемой.

Используя ИТВР, разработчик РБД имеет возможность принимать решение на основе анализа существующих данных. Кроме того, процесс поиска уникальных и минимальных ключей можно формализовать.

Кроме первичных ключей в РБД широко используются внешние ключи. [15, 16]. Такие ключи используются для связи между таблицами.

Определяют внешние ключи следующим образом: если в отношении R1 значения какого-либо атрибута, не входящие в его первичный ключ, равны значениям первичного атрибута отношения R, то такой атрибут называют внешним ключом для отношения R. В данной работе реализован способ поиска такого рода атрибутов в ИТВР. Эта задача в работе для таблиц ИТВ [53] не решается.

Целостность данных в основном рассматривается в двух аспектах – целостность сущностей и целостность согласования. [13].

Требование целостности сущностей можно сформулировать так: сущности реального мира должны быть различимы. То есть ни одна запись в таблице не должна повторяться. В противном случае возникнет множество проблем в процессе обработки данных.

Целостность согласования: ссылаться можно только на те данные, которые существуют. Такую целостность еще называют ссылочной целостностью. Нарушение этого правила может привести к трудноразрешимым проблемам в процессе функционирования БД.

Обеспечение целостности данных связано с правильным использованием первичных и внешних ключей. В работах [59-61] достаточно полно рассмотрены вопросы обеспечения целостности данных при проектировании РБД на основе ИТВ.

1.1.4. Нормализация и семантическое моделирование

Нормализация отношений – аппарат ограничений на отношения. Он обеспечивает устранение избыточности РБД, обеспечивает непротиворечивость данных в РБД:

« Концепции этой методологии являются не чем иным как соображениями здравого смысла, записанными в формальном виде. Сутью теории является поиск и формирование этих принципов здравого смысла, что, конечно же, является весьма непростой задачей. Однако, если такая задача будет решена, найденные принципы могут быть положены в основу

решений автоматизации, т.е. можно будет написать программу, позволяющую выполнять проектирование с помощью компьютера » [13].

Процесс нормализации основывается на понятии нормальных форм. Отношение находится в нормальной форме, если оно удовлетворяет заданному набору условий. В таком случае процедуру нормализации можно описать как приведение данного набора отношений к более желательной форме. Процедура нормализации включает разбиение отношения, которое не удовлетворяет условиям нормальной формы [13]. Процесс декомпозиции является операцией проекции исходного отношения.

Важно отметить, что операции проекции имеют смысл, когда отношение имеет не только схему, но и тело. А при традиционном проектировании тело отношения может иметь лишь гипотетический характер.

В работах [4-7, 27-37, 53] предложены и реализованы идеи автоматизации, применение которых оправдано для решения задач формирования РТ на основе ИТВ. В этих работах, в том числе, на достаточно высоком уровне разработаны и методы нормализации таблиц ИТВ. В связи с этим вопросы нормализации в данной работе не рассматриваются.

Семантическое моделирование. Разработчики БД обычно обладают весьма ограниченными сведениями о семантике хранящихся в них данных. Однако в РМД присутствуют семантические (смысловые) аспекты. Их примерами являются домены, первичные и внешние ключи [13]. В связи с этим семантическое моделирование представляет интерес. Этим проблемам посвящены работы как зарубежных [11 – 13], так и российских ученых [54 – 57].

Это моделирование осуществляется на этапе инфологического проектирования РБД. При этом выполняются следующие шаги:

- выявляется множество смысловых понятий;
- выделяются сущности предметной области;
- выполняется классификация сущностей;

- определяются свойства сущностей;
- формируются связи между сущностями.

В настоящее время распространен метод семантического моделирования “сущность-связь” [58]. Связи в этой модели могут иметь 4-е типа: “один к одному”, “один ко многим”, “многие к одному” и “многие ко многим”. Одной из важных задач, возникающих при формировании семантических моделей, является назначение связей.

В работах [59-61] предложены эффективные методы формирования всех типов связей между РТ, которые формируются на основе ИТВ. В связи с этим вопросы формирования связей между РТ в данной работе не рассматриваются.

1.2. Определение понятия расширенной информации табличного вида (ИТВР) и причины построения методики преобразования ИТВР в реляционное представление

1.2.1. Определение понятия ИТВР

Во введении приведено неформальное определение ИТВР. В связи с тем, что это понятие является основополагающим в данной работе его необходимо обсудить более детально.

Под информацией табличного вида понимается информация, которая воспринимается ее потенциальными пользователями как таблицы. Данные – это информация, фиксированная в регламентированном виде [16, 52]. Под ИТВР в работе, также как и в работах [37, 53], понимается информация, определенная представления в форме таблиц только внешним видом. Поэтому ИТВР не являются данными в полном смысле этого слова. В ИТВР, как правило, отражена семантика данных. Семантика отражена в виде заголовков таблиц [53].

Использование ИТВР настолько распространено, что пришлось разработать специализированные инструментальные средства, которые

обеспечивают работу с информацией табличного вида – электронные таблицы.

Вид отображения ИТВР разнообразный: на бумаге, в виде файла текстового процессора, в виде электронных таблиц, в виде HTML текстов и др. Способы отображения ИТВР также различаются. Например, разделителями столбцов и строк могут быть специальные символы, неотображаемые символы, линии, знаки табуляции, пробелы [5].

Заголовки таблиц могут быть размещены в нескольких строках и использоваться как подзаголовки. Заголовки могут быть размещены внутри таблицы. В качестве месторасположения заголовков может быть использован первый столбец таблицы. В одном столбце таблицы может размещаться информация различного типа. ИТВР не является данными в строгом смысле этого слова. ИТВР не регламентированы. Это приводит к невозможности их непосредственного использования в РБД [37].

На рис. 1.1 приведен фрагмент реальной таблицы каталога выставки собак “Союза кинологических обществ России”, сформированной в текстовом редакторе.

ЙОРКШИРСКИЙ ТЕРЬЕР

КЛАСС ЩЕНКОВ САМЦЫ

- 001 **БЕНИ САНДР`С ВАЛЕНТАЙМ**, щ/к, р.01.09.04, сталь с золотом, вл. Смирнов В., м. Сантана Готсенд Ланд К
- 002 **ГАЙ ЮЛИЙ КРИСС**, щ/к, р.20.07.04, сталь с золотом, вл. Кузьмичев И.В., о. W.F`s Blue Krystal Cascade, м. Верджиния Ач Норд
- 003 **МАЙКЛ АЧ НОРД**, щ/к, р.21.07.04, сталь с золотом, вл. Белова Н.И., о. Ozmillion Emperor, м. ВНК Behold A Bit of Blue
- 004 **МАРК АЧ НОРД**, щ/к, р.21.07.04, сталь с золотом, вл. Токарева Н.Д., о. Ozmillion Emperor, м. ВНК Behold A Bit of Blue
- 005 **СТЕФАН GOLDEN LABEL**, щ/к, р.30.09.04, сталь с золотом, вл. Губанов А.П., о. Ozmillion Anticipation, м. Ридана

КЛАСС ЩЕНКОВ САМКИ

- 006 **БЕТТИ САНДР`С ВАЛЕНТАЙМ**, щ/к, р.01.09.04, сталь с золотом, вл. , м. Сантана Готсенд Ланд К
- 007 **БЭЛЛА**, щ/к, р.24.08.04, сталь с золотом, вл. Лаврова И.В., о. Ozmillion Emperor, м. Галатейя
- 008 **ГАБРИЛЬ КРИСС**, щ/к, р.20.07.04, сталь с золотом, вл. Кузьмичев И.В., о. W.F`s Blue Krystal Cascade, м. Верджиния Ач Норд
- 009 **ЕВА ДИ**, щ/к, р.26.11.04, сталь с золотом, вл. Волкова Г.В., о. Ozmillion Anticipation, м. Дениза Вивальди
- 010 **КАНА**, щ/к, р.06.07.04, сталь с золотом, вл. Ермолаева Л.С., о. Ozmillion Anticipation, м. Крошка Барби
- 011 **ЛИЗИ МАРДИ ГРАСС**, щ/к, р.27.11.04, сталь с золотом, вл. Неганова Н.А., о. Ozmillion Emperor, м. Лили Марди Грасс
- 012 **ЛОЛА МАРДИ ГРАСС**, щ/к, р.27.11.04, сталь с золотом, вл. Нагайцева М., о. Ozmillion Emperor, м. Лили Марди Грасс

013	ЛЯЛЯ МАРДИ ГРАСС , щ/к, р.27.11.04, сталь с золотом, вл Карасева С.Ю., о. Ozmilion Emperor, м. Лили Марди Грасс
014	САНТАНА GOLDEN LABEL , щ/к, р.30.09.04, сталь с золотом, вл. Рыбакова А.А., о. Ozmilion Anticipation, м. Ридана
015	СТЕФАНИ GOLDEN LABEL , щ/к, р.30.09.04, сталь с золотом, вл. Петрова Д.А., о. Ozmilion Anticipation, м. Ридана
016	ШАНЕЛЬ ЛАМБЕРГ , щ/к, р.22.08.04, сталь с золотом, вл. Кинах Н.М., о. Ozmilion Anticipation, м. Хиллари Ламберг
017	DELFINA OD KA MI , PKR.III-52031, р.08.07.04, сталь с золотом, вл. Ивонина Е.В., о. Orby De Penghibur, м. Giorgia Ad Operam

КЛАСС ЮНИОРОВ САМЦЫ

018	АЙРОН ВИГАС , щ/к, р.05.01.04, сталь с золотом, вл. Рюмина Е., о. Ozmilion Anticipation,
019	БЭКХЕМ ROYAL DAN , щ/к, р.07.06.04, сталь с золотом, вл. Азарова М., о. Ozmilion Anticipation, м. Аэлига Сандра

КЛАСС ЮНИОРОВ САМКИ

020	БРИТАНИКА ВИКТОРИЯ ROYAL DAN , RUS04119150, р.07.06.04, сталь с золотом, вл. Расторгуева Е., о. Ozmilion Anticipation, м. Аэлига Сандра
021	ЕВГЕНИЯ ИЗ ХРУСТАЛЬНОГО ДВОРЦА , щ/к, р.20.04.04, сталь с золотом, вл. Тусеева Л., о. Song Na Niepogode, м. Ахеhead King Texla
022	ЕЛЕТТА НАТАЛИ , щ/к, р.28.01.04, сталь с золотом, вл. Аршинов А., о. W.F's Blue Krystal Cascade, м. Ксения Натали
023	ЕЛИЗАВЕТА НАТАЛИ , щ/к, р.28.01.04, сталь с золотом, вл. Бистяйкина Н., о. Ozmilion Emperor, м. Елизавета

КЛАСС МОЛОДЫХ САМЦЫ

024	ВИКОНТ ЛУИ , щ/к, р.12.04.03, сталь с золотом, вл. Климкина Т.А., о. Луинни, м. Кэтрин Зета Джон
-----	---

Рис. 1.1. Фрагмент таблицы каталога выставки собак “Союза кинологических обществ России”.

В таблице нет разделителей заголовков и нет разделителей строк. Эти разделители воспринимаются человеком, но представляют малоразрешимую проблему для выделения записей с помощью компьютера. В таблице присутствуют внутренние подзаголовки, причем двух уровней – породы собак и их классы. Документ подготовлен в Word и включает в себя несколько сотен страниц. В Союзе кинологических обществ России появилась потребность создания БД, которая позволила бы хранить каталоги за прошедшие годы. Заполнение вручную таблиц БД информацией такого рода – задача нетривиальная.

Следует отметить, что приведенный реальный пример не соответствует описанию понятия ИТВ [53] – здесь используются внутренние заголовки и подзаголовки. Пример показывает, что расширение понятия ИТВ оправдано. Рассмотренные свойства ИТВ необходимо отразить в их модели.

Еще более неудачная ИТВ, с точки зрения ее преобразования в РТ, фрагмент реальной таблицы продаж оборудования, представленная на рис. 1.2

A20																				
A		B C D E F G					H I J K L M					N O P Q R S T								
Продажа модернизации - месяц							Продажа модернизации - УТД					Коммерческие предложения - месяц					K			
Region/branch		Объем		Единицы		Цели	Объем		Единицы		Цели	План		Факт		Цели	Пл			
		План	Факт	План	Факт	Объем 000 руб.	ед.	План	Факт	План	Факт	Объем 000 руб.	ед.	ед.	Объем 000 руб.	ед.	Объем 000 руб.	ед.	Объем 000 руб.	ед.
MSR		12500,0	52,6	66,7	1,0	14750,0	77,2	1375,0	52,6	6,0	1,0	3150,0	15,0	75,8	16666,7	35,0	8379,0	178,0	39166,7	62,5
PY-1		250,0	0,0	1,0	0,0	600,0	2,8	750,0	0,0	3,0	0,0	1800,0	8,5	11,4	2500,0	26,0	7309,0	27,3	6000,0	34,1
E. Кочуров		0,0		0,0		0,0	0,0	0,0		0,0		0,0	0,0	0,0			0,0	0,0	0,0	0,0
Д. Соловьев		83,3		0,3		158,3	0,8	250,0	0,0	1,0	0,0	475,0	2,3	3,8	833,3			7,2	1583,3	11,4
А. Трофимов		83,3		0,3		150,0	0,8	250,0	0,0	1,0	0,0	450,0	2,3	3,8	833,3	7,0	1973,0	6,8	1500,0	11,4
О. Вязкин (new)		41,7		0,2		145,8	0,7	125,0	0,0	0,5	0,0	437,5	2,0	1,9	416,7	11,0	3554,0	6,6	1458,3	5,7
А. Москвитин (new)		41,7		0,2		145,8	0,7	125,0	0,0	0,5	0,0	437,5	2,0	1,9	416,7	8,0	1782,0	6,6	1458,3	5,7
PY-2		208,3	52,6	1,0	1,0	450,0	2,2	625,0	52,6	3,0	1,0	1350,0	6,5	9,5	2083,3	9,0	1070,0	20,5	4500,0	28,4
А. Моисеев		0,0		0,0		0,0	0,0	0,0		0,0		0,0	0,0	0,0			0,0	0,0	0,0	0,0
С. Саникова		104,2		0,5		154,2	0,8	312,5	0,0	1,5	0,0	462,5	2,3	4,7	1041,7			7,0	1541,7	14,2
В. Левин		104,2	52,6	0,5	1,0	154,2	0,8	312,5	52,6	1,5	1,0	462,5	2,3	4,7	1041,7	9,0	1070,0	7,0	1541,7	14,2
Вязкин-1		0,0		0,0		141,7	0,7	0,0	0,0	0,0	0,0	425,0	2,0	0,0	0,0			6,4	1416,7	0,0
PY-3		250,0	0,0	1,0	0,0	600,0	2,8	0,0	0,0	0,0	0,0	0,0	0,0	11,4	2500,0	0,0	0,0	27,3	6000,0	0,0
А. Ямашкин		0,0		0,0		0,0	0,0	0,0		0,0		0,0	0,0	0,0			0,0	0,0	0,0	0,0

Рис 1.2. фрагмент таблицы продаж оборудования

В этой таблице задействованы заголовки и подзаголовки 3-х уровней, 1-й столбец используется в качестве двухуровневого подзаголовка.

Следует отметить, что приведенный реальный пример не соответствует описанию понятия ИТВ в [53] - здесь один из столбцов, (первый), используется для формирования заголовков. Пример показывает, что расширение понятия ИТВ оправдано.

Рассмотренные и другие свойства ИТВ необходимо отразить в их модели, для того чтобы ее использовать в качестве посылки, отображающей реальное положение дел, при разработке методов и методики автоматизированного преобразования ИТВ в РТ.

На рис 1.3 приведен результат импорта данной таблицы в формат СУБД Access. Как видно из рисунка получены маловразумительные данные, которые в таком состоянии не могут быть обработаны. Следует отметить, что в процессе импорта возникали ошибки, а мощность исходной таблицы несколько тысяч записей. Таблицы такого вида обуславливают дополнительные трудности преобразования ИТВ в РТ при использовании ручных способов или существующих инструментальных средств.

Код	Поле1	Продажи моде	Поле3	Поле4	Поле5	Поле6	Поле7	Поле8	Поле9	Поле10	Поле11	Продажи моде	П
1		План 2001 (тыс					Цели 2001 (тыс					План 1 кв.	
2	Region/branch				Другие				Другие			Объем	ед.
3		всего	коммерч.	муницип.	города	Ед.	всего	коммерч.	муницип.	города	Ед.	000 руб.	
4	MSR	150000,0	20000,0	130000,0	0,0	800,0	177000,0	47000,0	130000,0	0,0	926,0	37500,0	200,0
5	РУ-1	3000,0	3000,0	0,0	0,0	12,0	7200,0	7200,0	0,0	0,0	34,0	750,0	3,0
6	Б.Кочуров	0,0					0,0					0,0	0,0
7	Д.Сапожников	1000,0	1000,0			4,0	1900,0	1900,0			9,0	250,0	1,0
8	А. Трофимов	1000,0	1000,0			4,0	1800,0	1800,0			9,0	250,0	1,0
9	О.Базыкин (new	500,0	500,0			2,0	1750,0	1750,0			8,0	125,0	0,5
10	А.Москвитин (r	500,0	500,0			2,0	1750,0	1750,0			8,0	125,0	0,5
11	РУ-2	2500,0	2500,0	0,0	0,0	12,0	5400,0	5400,0	0,0	0,0	26,0	625,0	3,0
12	А.Моисеев	0,0					0,0					0,0	0,0
13	С.Санникова	1250,0	1250,0			6,0	1850,0	1850,0			9,0	312,5	1,5
14	В.Левин	1250,0	1250,0			6,0	1850,0	1850,0			9,0	312,5	1,5
15	Вакансии-1	0,0					1700,0	1700,0			8,0	0,0	0,0
16	РУ-3	3000,0	3000,0	0,0	0,0	12,0	7200,0	7200,0	0,0	0,0	34,0	750,0	3,0
17	А.Ямашкин	0,0					0,0					0,0	0,0
18	Ю.Фомин	1000,0	1000,0			4,0	1900,0	1900,0			10,0	250,0	1,0
19	Е.Зензина (new	750,0	750,0			3,0	1800,0	1800,0			8,0	187,5	0,8
20	М.Ануров (new	750,0	750,0			3,0	1750,0	1750,0			8,0	187,5	0,8
21	Б.Петричко (r	500,0	500,0			2,0	1750,0	1750,0			8,0	125,0	0,5
22	РУ-4	1500,0	1500,0	0,0	0,0	5,0	3600,0	3600,0	0,0	0,0	17,0	375,0	1,3
23	С.Мытарев	0,0					0,0					0,0	0,0

Рис. 1.3. Результат импорта в Access таблицы продаж оборудования

В качестве последнего, но далеко не единственного примера, приведем фрагмент реальной таблицы оборудования, сформированной в формате Excel (Рис.1.4).

10.01.01	ШК	Шварев С.	3	Люкс			2500\$	90%	сл.нед.
10.01.01	ШК	Аппатова И.	3	Бизнес	13VTR-М	507,00		90%	сл.нед.
10.01.01	ШК	Аппатова И.	3	Люкс	13VTR-М	549,00		90%	сл.нед.
10.01.01	РУ	Шварев С.	3	Бизнес		94,00		90%	сл.нед.
10.01.01	ШК	Шварев С.	3	Люкс			2500\$	90%	сл.нед.
11.01.01	РУ	Шварев С.	5	Люкс	13VTR-М	767,00		60%	06.02.2001
11.01.01	РУ	Козловская Л.	5	Люкс	13VTR-М	1 307,00		60%	06.02.2001
10.01.01	ШК	Аппатова И.	3	Люкс	13VTR-М	549,00		90%	сл.нед.
12.01.01	ШК	Аппатова И.	2	Стандарт	13VTR-М	287,00		90%	сл.нед.
16.01.01	ШК	Аппатова И.	5	Бизнес	13VTR-М	324,00		80%	06.02.2001
16.01.01	ШК	Козловская Л.	2	Люкс	13VTR-М	1 232,00		60%	06.02.2001
22.01.01	ШК	Аппатова И.	3	Люкс		137,00		70%	06.02.2001
22.01.01	ШК	Козловская Л.	2	Люкс		120,00		60%	06.02.2001
15.01.01	РУ-2	Левин В.	3	Люкс	13VTR	550,00			21.05.2001
25.01.01	РУ-2	Левин В.	4	Бизнес	13VTR	430,00			06.03.2001
26.01.01	РУ-2	Левин В.	2	Стандарт	13VTR	864,00			22.03.2001
08.01.01	РУ-3	Зензина Е.	3	Бизнес	13VTR	400,00			26.03.2001
15.01.01	РУ-3	Зензина Е.	6	Бизнес	13VTR	400,00		Сентябрь 01	30.03.2001
19.01.01	РУ-3	Зензина Е.	6	Стандарт	13VTR	350,00		Октябрь 01	30.03.2001
16.01.01	РУ-4	Родченков Н.	8	Бизнес	13VTR-М	270,00		на рассмотр.	16.04.2001

Рис.1.4. Фрагмент таблицы оборудование

Здесь последний и предпоследний столбцы представлены доменами, в которых внесены данные различных типов. Действительно, в предпоследнем столбце представлены данные процентного и символьного типов, в последнем столбце представлены данные типа дата и символьного типов.

После преобразования данной таблицы в Access (рис 1.5) казалось бы, что все произошло удачно. Однако предпоследний столбец сформировался

символьного типа, а запрашивается числовой. Последний столбец также сформировался символьного типа, а по логике вещей должен быть тип дата. Ведь сортировка по датам в данном случае невозможна! К тому же в процессе импорта возникло более 200 ошибок преобразования.

Код	Поле1	Поле2	Поле3	Поле4	Поле5	Поле6	Поле7	Поле8	Поле9	Поле10
1	10.01.2001	ШК	Шварев С.		3 Люкс			2500\$	90%	сл.нед.
2	10.01.2001	ШК	Алпатова И.		3 Бизнес	13VTR-M	507		90%	сл.нед.
3	10.01.2001	ШК	Алпатова И.		3 Люкс	13VTR-M	549		90%	сл.нед.
4	10.01.2001	РУ	Шварев С.		3 Бизнес		94		90%	сл.нед.
5	10.01.2001	ШК	Шварев С.		3 Люкс			2500\$	90%	сл.нед.
6	11.01.2001	РУ	Шварев С.		5 Люкс	13VTR-M	767		60%	06.02.2001
7	11.01.2001	РУ	Козловская Л.		5 Люкс	13VTR-M	1307		60%	06.02.2001
8	10.01.2001	ШК	Алпатова И.		3 Люкс	13VTR-M	549		90%	сл.нед.
9	12.01.2001	ШК	Алпатова И.		2 Стандарт	13VTR-M	287		90%	сл.нед.
10	16.01.2001	ШК	Алпатова И.		5 Бизнес	13VTR-M	324		80%	06.02.2001
11	16.01.2001	ШК	Козловская Л.		2 Люкс	13VTR-M	1232		60%	06.02.2001
12	22.01.2001	ШК	Алпатова И.		3 Люкс		137		70%	06.02.2001
13	22.01.2001	ШК	Козловская Л.		2 Люкс		120		60%	06.02.2001
14	15.01.2001	РУ-2	Левин В.		3 Люкс	13VTR	550			21.05.2001
15	25.01.2001	РУ-2	Левин В.		4 Бизнес	13VTR	430			06.03.2001
16	26.01.2001	РУ-2	Левин В.		2 Стандарт	13VTR	864			22.03.2001
17	08.01.2001	РУ-3	Зензина Е.		3 Бизнес	13VTR	400			26.03.2001
18	15.01.2001	РУ-3	Зензина Е.		6 Бизнес	13VTR	400		Сентябрь 01	30.03.2001
19	19.01.2001	РУ-3	Зензина Е.		6 Стандарт	13VTR	350		Октябрь 01	30.03.2001
20	16.01.2001	РУ-4	Родченков Н.		8 Бизнес	13VTR-M	270		на рассмтр.	16.04.2001

Рис 1.5. Результат импорта в Access таблицы оборудование

Перевод таблицы такого рода формат РБД с помощью средств импорта принципиально невозможен.

Способ приведения значений атрибутов каждого домена к одному типу в работах посвященных преобразованию ИТВ в РТ в полном объеме не реализован.

Рассмотренные и другие свойства ИТВ необходимо отразить в их модели.

Отчасти повторяясь следует сказать, что до настоящего времени под ИТВ подразумевались таблицы обладающие следующими свойствами:

- ИТВ воспринимаются их разработчиками и потребителями как двумерные таблицы;
- в ИТВ могут отсутствовать разделители строк и разделители столбцов;
- в ИТВ одноименные значения данных могут располагаться в нескольких строках;

- в ИТВ типы значений таблиц, соответствующих одному столбцу, могут различаться;

- в ИТВ возможны подзаголовки;

- в ИТВ заголовки и/или подзаголовки одноименного столбца могут размещаются в нескольких строках.

На основе приведенного выше и проведенного анализа современных, реальных ИТВ сделан вывод о том, что кроме перечисленных свойств для ИТВР характерны следующие свойства, которые необходимо учитывать при разработке и реализации методов и методики автоматизированного преобразования ИТВР в РТР:

- в ИТВР допустимы внутренние подзаголовки;

- в ИТВР допустимы многоуровневые подзаголовки;

- в ИТВР допустимо отражение семантики данных посредством цвета, фона, шрифта и т.п.;

- в ИТВР допустимо повторное использование заголовков и подзаголовков (суть их определяется посредством места в таблице, цвета, фона, шрифта и т.п.);

- в ИТВР возможно использование значений любых доменов атрибутов в качестве подзаголовков;

- в ИТВР допустимо использование комбинированных подзаголовков;

- в ИТВР допустимо использование одноименных доменов;

- в ИТВР необязательно наличие первичных ключей;

- в ИТВР могут отсутствовать заголовки столбцов;

- в ИТВР возможно дублирование строк.

Здесь понятие ИТВ расширено. Это понятие, как отмечено во введении, названо ИТВР. Оно включает в себя известные свойства ИТВ и свойства перечисленные выше.

В рамках метода преобразования ИТВР в реляционные таблицы, который обусловлен введением понятия ИТВР, необходима разработка соответствующих алгоритмов и средств преобразования таблиц ИТВР к виду РТР, приемлемому для их автоматизированной обработки. Эти алгоритмы позволят использовать преимущества наличия заполненных ИТВР, на основе анализа содержимого которых можно формализовать проектные решения, и свести к минимуму дефекты проектирования РБД.

1.2.2. Причины и проблемы построения методики автоматизированного преобразования ИТВР в расширенные РТ (РТР)

Немало ИТВР, как показано в предыдущем параграфе, представлено в текстовом формате или в формате текстовых процессоров или еще чаще в формате электронных таблиц. В связи с новыми задачами нередко потребности пользователей этой информации таковы, что им удобнее работать с ней, используя средства современных СУБД [15].

Значительная часть ИТВР хранится в электронных таблицах (ЭТ). Это связано с рядом достоинств ЭТ. В этой связи ЭТ популярны и они в первую очередь используются для представления и обработки ИТВР [53]. Зачастую в этом есть оправданная необходимость, но нередко для обработки ИТВР важна возможность использования РБД. Поэтому актуальна проблема переноса информации, содержащейся в ИТВР в РБД. Преимущества РБД перед ЭТ рассматриваются в работах [26, 37].

В частности, БД по сравнению с электронными таблицами обладают следующими достоинствами:

- БД позволяет контролировать правильность вводимых данных.
- В РБД проще, чем в ЭТ построить интерфейс пользователя.
- В РБД возможно создание связей между таблицами.
- В РБД обеспечивается контроль дублирования данных.

- РБД в отличие от ЭТ обычно ориентировано на работу с несколькими пользователями одновременно.
- РБД имеет развитую систему защиты от несанкционированного доступа.
- В РБД предусмотрены развитые средства администрирования.
- В РБД реализуются специальные средства защиты и восстановления данных.

В ЭТ имеются специфические особенности представления информации, которые недопустимы в БД. Так, например, в заголовках ЭТ могут встречаться символы недопустимые с точки зрения БД, заголовки и подзаголовки могут различаться только цветом или видом шрифта. Например, на рис. 1.6 приведен реальный фрагмент таблицы выполнения работ. В ней помимо сложных заголовков в именах заголовков используются символы ”.”, а это недопустимо во многих БД.

	A	B	C	D	E	F	G	H	I	J	K	L	M
4		Заказчик			Ген.подрядчик			Инвестор			Архитектор		
5	Заказчик	Ф.И.О.	Теле фон, факс	Дата посл. Переговор.	Ф.И.О.	Теле фон, факс	Дата посл. Переговор.	Ф.И.О.	Теле фон, факс	Дата посл. Переговор.	Ф.И.О.	Теле фон, факс	Дата посл. Переговор.
6	Хохтиф				Хохтиф			Конти					
7	Хохтиф				Хохтиф			Конти					
8	Хохтиф				Хохтиф			Конти					
9	Хохтиф												
10	Хохтиф												
11	Хохтиф												
12	Прогрес				Прогрес			Газпром					
13	Прогрес				Прогрес			Газпром					
14	а/п Домодедово				СУВАС								
15	а/п Домодедово				СУВАС								
16	КGM										КGM		
17	КGM										КGM		
18	КGM										КGM		
19	КGM										КGM		
20	ФЕКА												
21	ЕНКА Минфин												

Рис. 1.6. Фрагмент таблицы выполнения работ

При импорте ИТВР в РБД и в частности при импорте ЭТ, могут возникнуть ошибки различных типов. Исправление ошибок процесс трудоемкий и не всегда приводит к успеху.

Кроме того, после преобразования ИТВР в РТР возникает проблема назначения первичных и внешних ключей.

В связи вышесказанным можно сделать выводы:

- С точки зрения пользователей ИТВР мотивами преобразования ИТВР в РТР является потребность использования преимуществ БД.
- С точки зрения разработчика метода преобразования ИТВР в РТР мотивом его разработки являются перечисленные выше преимущества РТР перед ИТВР.

Резюмируя сказанное, можно сделать вывод о том, что проблема разработки методики преобразования ИТВР в РТР актуальна и представляют теоретический и практический интерес.

1.3. Анализ применимости современных теоретических и практических разработок

Анализ применимости современных теоретических разработок

Как отмечают специалисты в области проектирования и использования БД на протяжении всего существования реляционных БД они постоянно предлагали наилучшую смесь простоты, устойчивости, гибкости, производительности, масштабируемости и совместимости в сфере управления данными [44].

Это связано с наличием стройной методологии разработки реляционных БД [11, 17, 29, 43, 46, 47]. Руководствуясь ею, разработчик может успешно выполнить все этапы проектирования, включая постановку и анализ требований, концептуальное проектирование, даталогическое и физическое проектирование.

Методология позволяет решить проблемы нормализации таблиц БД, назначить ключевые поля, сформировать связи между таблицами.

Но даже основоположники реляционного подхода утверждают, что формализовать процесс проектирования РБД пока не удастся [13]. Это связано с тем, что все выкладки делаются на основе ручного анализа предполагаемых схем отношений. Но в этих схемах возможны ошибки, которые проявятся, может быть, только в процессе эксплуатации

спроектированной БД. Но ничего лучшего на сегодня нет. Реляционная методология проектирования БД является одной из лучших на сегодня методологий. И не использовать ее компоненты, по меньшей мере, не разумно.

Ситуация существенно меняется, когда при проектировании БД используются не гипотетические схемы отношений, а реальные, заполненные таблицы. Вот здесь и можно формализовать процесс проверки соответствия данных реляционным таблицам, процесс назначения ключевых полей, процесс нормализации таблиц, процесс формирования связей таблиц. Ведь все это можно сделать на основе автоматизированного анализа реальных данных (если, конечно, данные такого рода есть). В этом случае можно использовать традиционную методологию наиболее полно и эффективно. Ведь при решении всех вопросов разработчик отталкивается от реальных данных, и все рекомендации методологии проектирования РБД могут быть в полной мере формализованы.

Поэтому методика, разрабатываемый в диссертации, базируется на теории проектирования РБД.

Анализ применимости современных практических разработок

Во всех СУБД предусмотрены средства импорта данных. При этом данные могут быть представлены в различных форматах. В том числе и в форматах .txt и .xls. Но как показывает практика при импорте таблиц, которые хотя бы частично не удовлетворяют требованиям к РТР, возникает множество ошибок. И исправление этих ошибок очень трудоемко, а иногда и не представляется возможным.

Так, например, в системе Oracle используется специальная утилита для перемещения данных – SQL*Loader [66, 67]. При перемещении данных из другого приложения формируются файлы отвергнутых и некорректных записей. Некорректные записи – это записи, которые не могут быть включены в РБД по каким-либо причинам. Это говорит о том, что проблема преобразования ИТВР в РТР актуальна.

В Access есть развитые средства импорта данных. Но при импорте, казалось бы, самых “безобидных” таблиц, как правило, формируется файл ошибок, который зачастую включает в себя сотни записей.

Microsoft SQL Server [69, 70] также обеспечивают импорт данных. Но предварительно эти данные должны быть представлены в форме реляционных таблиц.

Таким образом, все современные СУБД можно использовать для импорта данных. Причем импортируемые данные могут быть представлены в различных форматах.

Однако использование средств импорта ограничено. Их оправдано использовать, когда таблицы имеют небольшую мощность и степень. Тогда необходимые исправления впоследствии можно внести вручную. Кроме того средства импорта оправдано использовать, когда импортируемые таблицы максимально приближены к реляционному представлению.

В качестве итога можно сделать вывод. При незначительных размерах таблиц (десятки полей и сотни записей) преобразования ИТВР в РТР можно выполнять “вручную” – импортировать данные, а затем ввести нераспознанные отложенные записи и выполнить корректировку импортированных записей. При больших размерах таблиц ИТВР (сотни полей и тысячи записей) целесообразно использовать автоматизированные средства. Это подтверждает опыт разработок и экспертные исследования.

1.4 Постановка задачи разработки метода формирования РТР на основе использования ИТВР

Состав способов, алгоритмов и средств, разрабатываемых в рамках методики

Сравнительный анализ неформального описания РТР и неформального описания ИТВР, выполненный выше, позволили определить состав методов, основных алгоритмов и средств, ориентированных на преобразование ИТВР в РТР.

В рамках метода преобразования ИТВР в РТР необходимо разработать следующие алгоритмы:

- исключения заголовков в содержимом таблиц;
- исключения подзаголовков в содержимом таблиц;
- исключения комбинированных подзаголовков;
- исключения пустых подзаголовков;
- избавления от вертикальных подзаголовков;
- приведения значений атрибутов таблиц к одному типу;
- исключения дублирования записей в таблицах;

При этом в отличие от алгоритмов, предложенных в работах [37, 53]:

- рассматриваются возможные сочетания подзаголовков (внешние, внутренние, вертикальные);
- рассматриваются пустые заголовки;
- расширяется понятие вертикальных подзаголовков;
- анализируется проблема исключения пустых записей;
- детально анализируется проблема приведения значений доменов к одному типу.

Кроме того, все алгоритмы, предложенные в работах [5, 37, 53], уточнены, дополнены и представлены более детально и наглядно.

В рамках метода назначения ключевых полей в преобразованных таблицах ИТВР необходимо разработать следующие алгоритмы:

- выявления столбцов с уникальными значениями его элементов;
- выявления сочетания столбцов с уникальными сочетаниями соответствующих им элементов;
- поиска минимальных первичных ключевых полей, которые включают в себя один атрибут;
- поиска минимальных первичных ключевых полей, которые включают в себя несколько атрибутов;

- выявления атрибутов, которые входят в первичный ключ и содержат уникальные значения;
- выявления внешних ключевых полей.

При этом в отличие от алгоритмов, рассмотренных в работах [5, 37, 53]:

- выявление первичных и внешних ключей включаются в этап преобразования ИТВР в РТР ;
- выполняется поиск минимальных первичных ключей, которые включают в себя несколько атрибутов, выполняется в соответствии со всеми требованиями к минимальности ключей;
- выполняется выявление внешних ключей в ИТВР.

Кроме того, алгоритмы, описанные в [5, 37, 53], модифицированы, развиты и дополнены.

С учетом сказанного выше разработана схема иерархии новых алгоритмов – рис. 1.7.

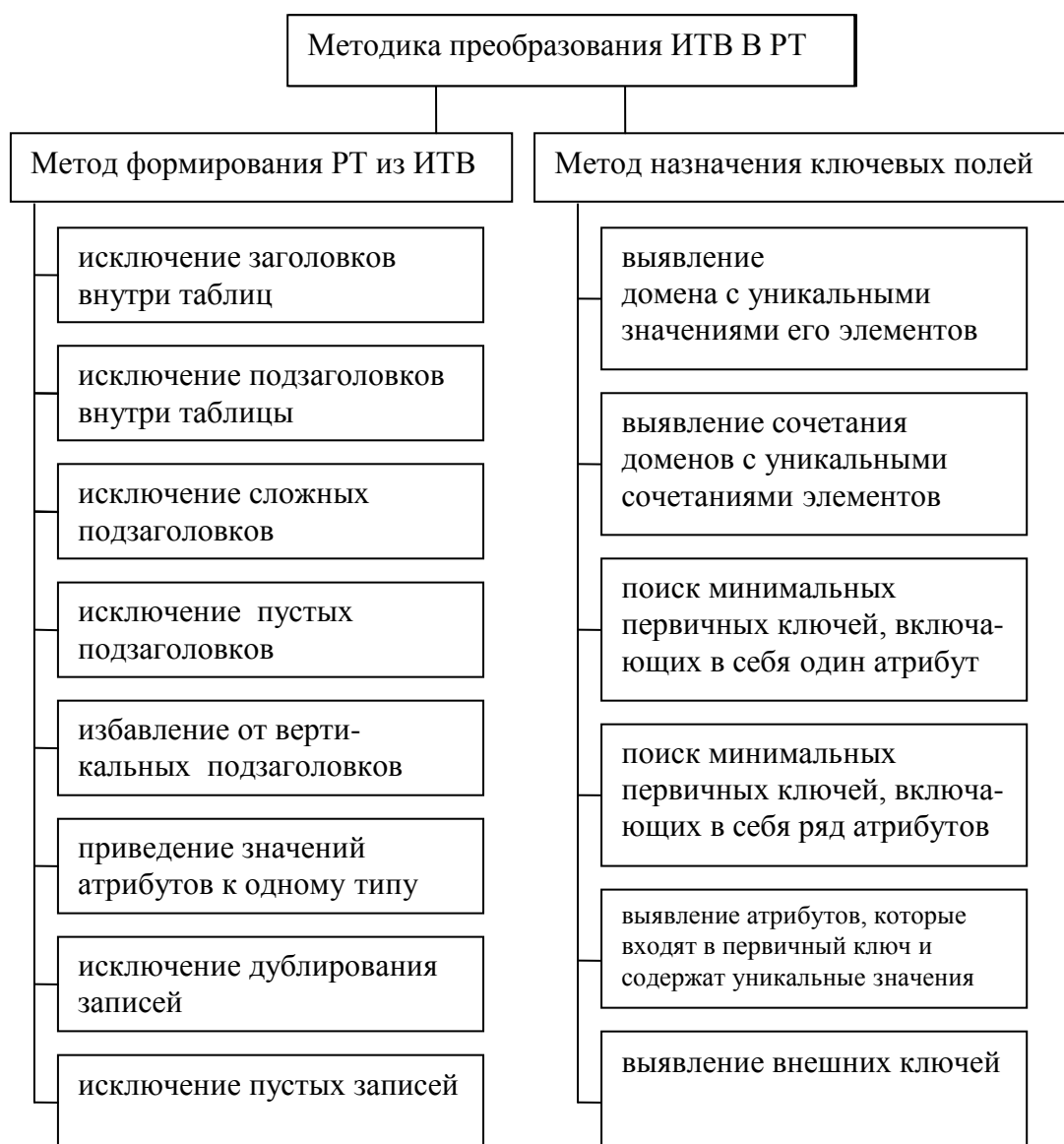


Рис. 1.7. Схема иерархии разрабатываемых алгоритмов

Нужно отметить, что разрабатываемые алгоритмы – человеко-машинные. То есть предполагается активное участие разработчика РБД в ходе преобразования ИТВР в РТР. Разработчику предоставляется возможность принимать решения по выбору последовательности действий и выбору вариантов проектных решений, предлагаемых подсистемой. Эта возможность в конечном итоге сведет к минимуму ошибки проектирования и позволит в полном объеме использовать опыт разработчика.

Проблемам автоматизированного проектирования посвящено ряд работ, в частности [70 – 73]. Следует учитывать общие закономерности

автоматизированного проектирования, которые изложены в соответствующей литературе.

Выводы по главе 1

1. Выполнен анализ задач построения методики формирования РТР на основе использования заполненных ИТВР. Он позволил оценить применимость классической методологии проектирования РБД и существующей методологии построения РБД на основе ИТВР для их решения.

2. Сформулировано 10, не рассмотренных ранее, особенностей ИТВ. Отражение этих особенностей в модели ИТВ позволит построить ее адекватную модель (ИТВР) и решать в автоматизированном режиме задачи преобразования ИТВР в РТР, которые до сих пор решались вручную.

3. Сформулирована проблема автоматизированного формирования РТР на основе ИТВР. Это позволило определить состав методов, алгоритмов и процедур, которые необходимо включить в разрабатываемую методику.

2. МЕТОД ПРЕОБРАЗОВАНИЯ ЗАПОЛНЕННЫХ ИТВР В РТР

Во второй главе разработаны: модели объектов исследования (РТР и ИТВР), которые отражают реальное положение дел в области представления табличных данных; метод преобразования таблиц ИТВР к реляционному виду. В рамках метода рассмотрены алгоритмы приведения атрибутов ИТВР к единому типу, исключения дублирования записей в ИТВР, избавления от сложных атрибутов и исключения простых и сложных подзаголовков.

2.1 Модели объектов исследования

2.1.1 Модель РТР

Во всех работах посвященных РБД сформулированы требования к РТ. Эти требования, как правило, представляют собой перечисление свойств, которыми должны обладать РТ. Требования не формализованы, да они и не могут быть формализованы, т.к. РБД проектируются на основе схем отношений. Например, схема отношения Абитуриент может иметь вид: Абитуриент = (Фамилия, Имя, Отчество, Дата рождения, Дата поступления, Адрес,..., Средний балл). Требования предъявляются к данным, а данных то еще нет. Данные появятся в процессе эксплуатации БД, и может оказаться, что сформулированные на начальных этапах проектирования требования к РТ не удовлетворяются. В частности, отталкиваясь от схемы отношения Студент, вполне можно предположить, что сочетание Фамилия, Имя, Отчество вполне подходит для первичного ключа. Однако, как показывает опыт в рамках даже одной организации, нередко встречаются полные тезки. А основным требованием к первичному ключу является его уникальность.

Нередко исходная информация присутствует в виде ИТВР. В этом случае ситуация существенно меняется в лучшую сторону. Ведь на основе реальных данных вполне возможно формализовать требования к ним.

В работе [53] описана формальная модель РТ. Основные ее черты следующие:

« Реляционная таблица (RT) представляется множеством

$$RT = \{Z, D\},$$

где Z – множество заголовков, D – множество данных.

$$z = \{z_1, \dots, z_i, \dots, z_n\}, \quad i = 1, n; \quad n \geq 1 \quad (1)$$

где n – степень множества заголовков.

Должно быть обеспечено условие $z_i \neq z_m, \quad i = 1, n; \quad m = 1, n; \quad i \neq m, \quad (2)$

где n – степень множества заголовков, т.е. недопустимо совпадение заголовков.

$$D = \{SD\}, \quad \text{где } SD \text{ – множество строк данных.} \quad (3)$$

$$SD = \{SD_1, \dots, SD_i, \dots, SD_n\}, \quad i = 1, n; \quad n \gg, \quad (4)$$

где n – мощность множества строк данных.

$$SD_i = \{ED_{i1}, \dots, ED_{ij}, \dots, ED_{ik}\}, \quad j = 1, k; \quad k \geq 1,$$

где k – степень множества i -ой строки данных, ED_{ij} – элемент данных.

Недопустима ситуация, когда внутри таблицы данных могут встретиться заголовки, т.е. должно выполняться условие:

$$SD_i \neq z_j, \quad i = 1, n; \quad n \gg 1; \quad j = 1, k; \quad k \geq 1, \quad (5)$$

где n – мощность множества строк данных, k – степень множества заголовков.

Для реляционных таблиц выполняется правило:

$$(\forall ED) (ED \in SD) (\exists z (z \in Z) (z \leftrightarrow ED)) \quad (6)$$

Т.е. каждому элементу данных соответствует только один заголовок.

$$(\forall ED) (ED \in SD) (\exists (TED) (ED \leftrightarrow TED)), \quad (7)$$

где $TED = string \vee integer \vee datetime \vee real \vee logical$.

Т.е. каждому элементу данных соответствует определенный тип данных.

В реляционных таблицах обязательно выполнение следующего требования:

$$\begin{aligned} TED_{11} =, \dots, = TED_{i1} =, \dots, = TED_{n1} \\ \dots \dots \dots \dots \dots \dots \dots \dots \dots \\ TED_{1j} =, \dots, = TED_{ij} =, \dots, = TED_{nj} \\ \dots \dots \dots \dots \dots \dots \dots \dots \dots \\ TED_{1k} =, \dots, = TED_{ik} =, \dots, = TED_{nk}, \quad i = 1, n; n \gg 1; j \\ = 1, k; k \geq 1, \end{aligned} \quad (8)$$

где n - мощность множества строк данных, k - степень множества i -ой строки данных; ED_{ij} - элемент данных. Другими словами, значения типов данных одного столбца должны совпадать.

$$\begin{aligned} \text{Недопустима ситуация, когда } SD_i = SD_j, \quad i = 1, n; j = \\ 1, n; i \neq j, \end{aligned} \quad (9)$$

где n - мощность множества данных. Т.е. невозможно полное совпадение строк данных » [53].

Однако в данном случае отражены не все свойства РТ, которые отвечают реальным требованиям к РБД и, которые в числе прочих предполагается взять за основу в данной работ. И соответственно эта модель не может в полной мере служить исходной формализацией для методов, разрабатываемых в диссертации.

В реляционной модели данных [53] не предусмотрено наличие первичных ключей. Расширение модели РТ (РТР) состоит в следующем.

$$\text{Пусть } SD_i = \{ED_{i1}, \dots, ED_{ij}, \dots, ED_{ik}\}, \quad j = 1, k; k \geq 1,$$

где SD_i - i -я строка данных множества, k - степень множества, ED_{ij} – элемент данных.

Должен быть найден такой заголовок Z_i ($Z_i \leftrightarrow ED_{ij}$), такой чтобы выполнялось выражение:

$$ED_{i1} \dots \neq \dots ED_{ij} \dots \neq \dots ED_{in} \quad , \text{ где } n - \text{ мощность множества} \quad (11)$$

В таблице должен присутствовать такой столбец, все значения которого бы отличались. Такой столбец можно использовать в качестве первичного ключа.

Кроме того, в качестве первичного ключа могут использоваться сочетания атрибутов таблицы.

В реляционной модели данных не должно быть пустых заголовков:

$$Z_i \neq \emptyset \quad (12)$$

В реляционной модели данных не должно быть пустых строк:

$$SD_i \neq \emptyset \quad (13)$$

В реляционной модели данных содержимое никакого столбца не могут использоваться как подзаголовки. (14)

В реляционной модели недопустимы внутренние заголовки.

В реляционной модели недопустимы пустые домены.

В реляционной недопустимо использование одноименных доменов.

В реляционной модели данных не должно быть сочетаний различного типа подзаголовков.

В связи с этим введем понятие РТР и соответствующее им описание модели РТР.

2.1.2 Модель ИТВР

Если в предыдущем параграфе рассматривалось целевая модель данных РТР, то далее речь пойдет об исходной модели, которую необходимо преобразовать в форму РТР – модель ИТВР.

Модель ИТВР в известной литературе не рассматривается. В работе [53] представлена неполная формализованная модель ИТВ, которая с модификациями может быть принята в качестве базовой для решения задач диссертации. Рассмотрим эту модель.

« Информация табличного вида (ИТВ) представляется множеством $DT = \{Z, D\}$, где Z – множество заголовков ИТВ, D – множество данных, соответствующих заголовкам . (15)

$$Z = \{Z_1, \dots, Z_2, \dots, Z_n\}, \quad i = 1, n; \quad n \geq 1, \quad (16)$$

где n – степень множества заголовков.

$$\text{Допустима ситуация, когда } Z_i = Z_m, \quad i = 1, n; \quad m = 1, n; \quad i \neq m, \quad (17)$$

где n – степень множества заголовков, т.е. возможно полное совпадение заголовков

В данных табличного вида возможны подзаголовки 1-го уровня, что формально выглядит следующим образом.

$$Z_i = \{PZ_{i1}, \dots, PZ_{ij}, \dots, PZ_{ik}\}, \quad j = 1, k; \quad k \geq 2, \quad (18)$$

где k - степень множества подзаголовков i -го заголовка.

$$Z_p = \{PZ_{p1}, \dots, PZ_{pt}, \dots, PZ_{pm}\}, \quad t=1, m; \quad m \geq 2 \quad (19)$$

где m – степень множества подзаголовков p -го заголовка.

Допустима ситуация, когда $PZ_{ij} = PZ_{pt}$. (20) » [53].

В ИТВ возможны подзаголовки 2-го уровня, что формально выглядит следующим образом.

$$PZ_{ij} = \{PPZ_{ij1}, \dots, PPZ_{ijm}, \dots, PPZ_{ijf}\}, \quad m = 1, f; \quad f \geq 2, \quad (21)$$

где f – степень множества подзаголовков 2-го уровня ij -го подзаголовка 1-го уровня.

$$PZ_{pt} = \{PPZ_{pt1}, \dots, PPZ_{ptr}, \dots, PPZ_{ptq}\}, \quad r = 1, q; \quad q \geq 2, \quad (22)$$

где q – степень множества подзаголовков 2-го уровня pt -го подзаголовка 1-го уровня

$$\text{Допустима ситуация, когда } PPZ_{ijm} = PPZ_{ptm}. \quad (23)$$

Теоретически в данных табличного вида может быть больше уровней подзаголовков.

На рис. 2.1 приведен реальный пример таблицы ИТВ с подзаголовками.

Регион	№ п/п	Дата	Город	Продавец	Заказчик	Сегмент	Количество	Тип оборудования							Тип конт.	Цена			
								лифтов	Купе кабины	Лебедка	С-ма	Двери	Вызывная и ризажн. аппа	Устр-ва безоп-ти		Другое	ракта (М, М+L, L)	в тыс. руб.	или \$
0	1	02.01.01	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17		
MSR	E2S68-MSR	10.01.01	ШК	Шкарев С.	Частное	лицо	2	1						фотореверс	M+L		1 750		
MSR	E2S69-MSR	10.01.01	ШК	Аплатова	Централ.те		3	1	Бизнес	13VTR-M	КЛМ-VF	Ств. ДШ	Бизнес	ВКЛ	M+L	507,00			
MSR	E2S70-MSR	10.01.01	ШК	Аплатова	Централ.те		3	1	Люкс	13VTR-M	КЛМ-VF	Ств. ДШ	Люкс	ВКЛ	M+L	549,00			
MSR	E2S71-MSR	10.01.01	ШК	Аплатова	Централ.те		3	1	Бизнес				Бизнес	НЕ ВКЛ	M+L	94,00			
MSR	E2S72-MSR	10.01.01	ШК	Аплатова	Централ.те		3	1	Люкс				Люкс	НЕ ВКЛ	M+L	149,00			
MSR	E2S73-MSR	11.01.01	ШК	Козловская	ОАО		5	2	Люкс	13VTR-M	КЛМ-1	Ств. ДШ	Люкс	ВКЛ	M+L	767,00			
MSR	E2S74-MSR	11.01.01	ШК	Козловская	ОАО		5	2	Люкс	13VTR-M	КЛМ-VF	Ств. ДШ	Люкс	ВКЛ	M+L	1 307,00			
MSR	E2S75-MSR	11.01.01	ШК	Шкарев С.	Госкомрезе		2	2	Бизнес			Ств. ДШ	Люкс	НЕ ВКЛ	M+L	388,00			
MSR	E2S76-MSR	12.01.01	ШК	Аплатова	Централ.те		2	1	Стандарт	13VTR-M	КЛМ-1	Ств. ДШ	Бизнес	ВКЛ	M+L	287,00			
MSR	E2S77-MSR	16.01.01	ШК	Аплатова	ГНП		5	1	Бизнес	13VTR-M	КЛМ-1	Ств. ДШ	Бизнес	ВКЛ	M+L	324,00			
MSR	E2S78-MSR	16.01.01	ШК	Козловская	Министер.к		2	3	Люкс	13VTR-M	MCS-220	Ств. ДШ	Люкс	ВКЛ	M+L	1 232,00			
MSR	E2S79-MSR	22.01.01	ШК	Аплатова	"Импексбан		3	1	Люкс				Люкс	НЕ ВКЛ	M+L	137,00			
MSR	E2S80-MSR	22.01.01	ШК	Козловская	Министер.ф		2	1	Люкс				Люкс	НЕ ВКЛ	M+L	120,00			
MSR	E2S81-MSR	16.01.01	РУ-2	Левин В.	"Рособорон		3	3	Люкс	13VTR	КЛМ-В		Люкс	ВКЛ	M+L	550,00			
MSR	E2S82-MSR	26.01.01	РУ-2	Левин В.	Г-ца		4	2	Бизнес	13VTR	КЛМ-1	Ств. ДШ	Бизнес	ВКЛ	M+L	430,00			
MSR	E2S83-MSR	26.01.01	РУ-2	Левин В.	МГУ ПС		2	2	Стандарт	13VTR	КЛМ-В	Ств. ДШ	Стандарт	ВКЛ	M+L	864,00			
MSR	E2S84-MSR	08.01.01	РУ-3	Зенкина Е.	ЗАО		3	1	Бизнес	13VTR	КЛМ	Пластик	Бизнес		нет	M+L	400,00		

Рис. 2.1. Пример таблицы ИТВ с подзаголовками

Здесь заголовок "Тип оборудования" включает в себя 7 подзаголовков, заголовок "Цена" включает в себя 2-а подзаголовка.

$$D = \{SD, Z\}, \text{ где } SD - \text{ множество строк данных.} \quad (24)$$

Это представление D допускает наличие нескольких заголовков и подзаголовков 1-го и 2-го уровней, расположенных в области данных.

В том числе допускается наличие заголовков и подзаголовков, расположенных до, после и между строк данных. На рис. 2.2 приведен реальный пример таблицы ИТВР с заголовками внутри области данных. Таковыми являются строки 1 и 9 и др. Кроме того, на рис. 2.2 представлен внутренний подзаголовок более высокого уровня - "Собрать свой компьютер".

Рис. 2.2. Пример таблицы ИТВР с заголовками внутри области данных

Этот пример не соответствует модели ИТВ [53] – в нем присутствуют внутренние заголовки и подзаголовки.

$$SD = \{SD_1, \dots, SD_i, \dots, SD_n\}, \quad i = 1, n; \quad n \gg 1, \quad (25)$$

где n - мощность множества строк данных.

$$SD_i = \{ED_{i1}, \dots, ED_{ij}, \dots, ED_{ik}\}, j = 1, k; k \geq 1, \quad (26)$$

где k – степень множества элементов данных i -ой строки данных;

ED_{ij} – элемент данных.

Для ИТВ должно выполняться следующее правило [53]:

$$\begin{aligned} & \ll (\forall ED) (ED \in SD) (\exists z (z \in Z) (z \leftrightarrow ED)) \vee (\exists (PZ)) (z \\ & \in PZ) (PZ \leftrightarrow ED) \vee (\exists (PPZ)) (PPZ \in PZ) (PPZ \leftrightarrow ED) \end{aligned} \quad (27)$$

Т.е. каждому элементу данных соответствует заголовок или подзаголовок 1-го или 2-го уровней и наоборот.

$$(\forall ED) (ED \in SD) (\exists TED (TED \in T(ED))), \quad (28)$$

где $TED = \text{string} \vee \text{integer} \vee \text{datetime}$, где $T(ED)$ – тип ED .

Но каждому элементу данных соответствует какой-либо тип данных.

В общем случае:

$$\begin{aligned} & TED_{11} \neq, \dots, \neq TED_{i1} \neq, \dots, \neq TED_{n1} \\ & \dots \dots \dots \dots \dots \dots \dots \dots \dots \\ & TED_{1j} \neq, \dots, \neq TED_{ij} \neq, \dots, \neq TED_{nj} \\ & \dots \dots \dots \dots \dots \dots \dots \dots \dots \\ & TED_{1k} \neq, \dots, \neq TED_{ik} \neq, \dots, \neq TED_{nk} \quad i = 1, n; n \gg 1; j = 1, k; k \\ & \geq 1, \quad (29) \end{aligned}$$

где n - мощность множества строк данных, k - степень множества элементов i -ой строки данных. Другими словами, значения типов данных одного столбца могут не совпадать.

Допустима ситуация, когда $SD_i = SD_j$, $i = 1, n$; $j = 1, n$; $i \neq j$, (30) где n – мощность множества данных, т.е. возможно полное совпадение строк данных » [53].

В модели ИТВР допускаются пустые заголовки:

$$Z_i = \emptyset \quad (31)$$

В модели ИТВР допускаются пустые строки:

$$SD_i = \emptyset \quad (32)$$

В ИТВР содержимые столбца могут использоваться как подзаголовки таблиц. (33)

В ИТВР могут быть сочетания различного типа подзаголовков (34)

В ИТВР допустимы внутренние заголовки.

В ИТВР допустимы пустые домены.

В ИТВР допустимо использование одноименных доменов.

Выше описана расширенная модель ИТВ - модель ИТВР.

Несмотря на некоторое сходство модели таблиц ИТВР и модели РТР, в них имеются существенные различия.

Сравним выражения для РТ и ИТВР.

1. Ограничения (17-24) для ИТВР противоречат ограничению (2) для РТР.
2. Ограничения (27-28) для ИТВР противоречат ограничению (5) для РТР.
3. Ограничения (31-33) для ИТВР противоречат ограничениям (12-14) для РТР.
4. Ограничение (29) для ИТВР противоречат ограничению (8) для РТР.
5. Ограничение (30) для ИТВР противоречат ограничению (9) для РТР.

Кроме того, в существующей модели РТ не предусмотрено наличие первичного ключа и отсутствуют требования к первичному ключу. А также не учтена возможность описания внешних ключей.

Одной из задач работы является: разработка метода автоматизированного преобразования ИТВР в РТР, который отражает введенные в работе модели РТР и ИТВР и позволит решать в полном объеме

задачи формирования РТР на основе заполненных ИТВР. Для ее решения необходима разработка средств, которые бы позволили преобразовать ИТВР в РТР. Эти средства должны обеспечить исключение отличий между объектами, которым соответствуют моделям ИТВР и РТР.

2.2 Задача приведения значений столбцов ИТВР к одному типу

2.2.1 Типы полей в реляционных таблицах

В РТР основные типы полей: числовой, текстовый, дата-время. В БД, если в каком-либо поле таблицы РТР вводятся новые значения, тип значения проверяется автоматически. В случае, если введенное значение не соответствует назначенному типу, выдается сообщение об ошибке. При заполнении таблиц ИТВР проверки типов вводимых значений не производятся. Поэтому в таблицах ИТВР в одном и том же домене могут храниться значения различных типов. Это недопустимо в РБД. Поэтому необходимы алгоритмы и средства преобразования элементов каждого столбца таблиц ИТВР к одному типу.

В качестве примера ИТВР с различными типами значений элементов в одноименных доменах представим таблицу (рис. 2.3).



	Имя	Зарплата	Возраст	День свадьбы	Должность
▶	Федоров	1700\$	34 года	09-03-1997	Инженер
	Алексей	нет работы	17 лет	неженатый	Студент
	Андрей	1100\$	45 лет	12-11-1995	Продавец
	Елена	1500\$	27 лет	23-04-2009	Преподаватель
	Ольга	нет работы	16 лет	незамужняя	Студентка
	Инна	2200\$	45 лет	02-10-1995	Врач
*					

Рис. 2.3. Таблица с различными типами значений атрибутов в одноименных столбцах

В ней в столбце "Зарботок" должны располагаться значения денежного типа, а в столбце "День свадьбы" – значения типа "Дата/Время". Но это не так. Есть смешение типов. Предлагается алгоритм назначения типов атрибутов заполненных таблиц.

Смысл его в следующем. В каждом столбце проверяются все значения, на соответствие проверяемому типу. Если значение атрибута не может быть отнесено к анализируемому типу, то во вспомогательную таблицу записывается это значение и его позиция. По завершению проверки выводятся записи вспомогательной таблицы. После анализа вспомогательной таблицы разработчик может принять решение по поводу назначения типа для каждого столбца ИТВР.

2.2.2. Преобразование значений атрибутов заполненных таблиц к одному типу

При преобразовании ИТВР в РТР значения атрибутов заполненных таблиц ИТВР в каждом столбце необходимо привести к одному типу.

Рассмотрим пример в Microsoft Access. В нем задана структура 2-х таблиц: ТаблицаТекст и ТаблицаДеньги. Каждая таблица имеет одно поле. В ТаблицеТекст поля типы – "Текстовый", в ТаблицеДеньги поля типы – "Денежный". В обеих таблицах имена полей одинаковые – "Деньги". ТаблицаДеньги не заполнена, а ТаблицаТекст заполнена данными на рис. 2.4.

Деньги
400\$
300\$
30\$
500\$
двести\$.
двадцатьРублей
▶ 15Рублей
*

Рис.2.4. Таблица Текст с данными

Можно добавить данные из ТаблицаТекст в ТаблицаДеньги. С этой целью сформирован запрос:

```
INSERT INTO ТаблицаДеньги
```

```
SELECT ТаблицаТекст.*
```

```
FROM ТаблицаТекст;
```

Здесь конструкция «INSERT INTO» ТаблицаДеньги указывает на данные, которые добавляются ко всем полям таблицы “ ТаблицаДеньги ”.

Конструкция “SELECT ТаблицаТекст.* FROM ТаблицаТекст;”

свидетельствует о том, что при добавлении используются данные из всех полей ”ТаблицаТекст ”.

После выполнения запроса сформируется сообщение (рис.2.5).

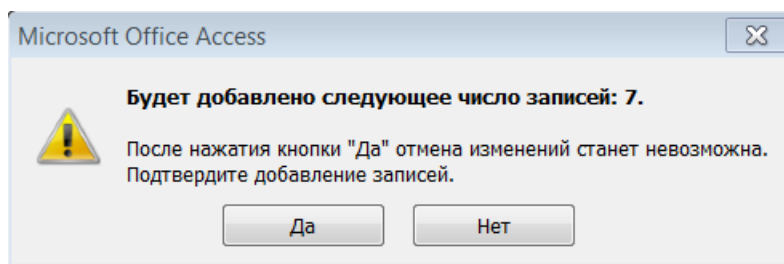


Рис.2.5. Первое сообщение, сформированное после выполнения.

Затем сформируется сообщение, представленное на рис.2.6.

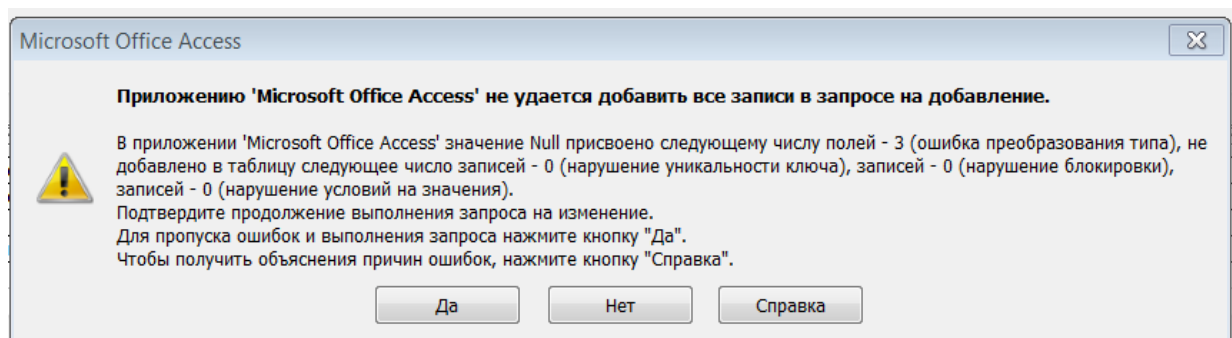
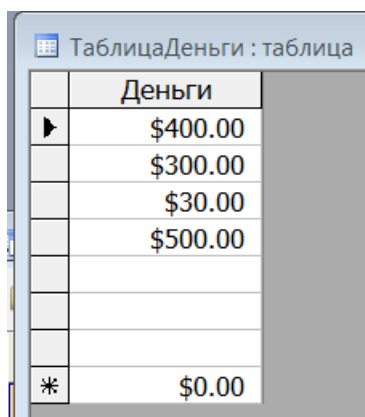


Рис.2.6. Второе сообщение, сформированное после выполнения запроса

В результате выполнения этого запроса ТаблицаДеньги примет следующий вид (рис. 2).



	Деньги
▶	\$400.00
	\$300.00
	\$30.00
	\$500.00
*	\$0.00

Рис. 2.7. Результат выполнения запроса на добавление

После сравнения рис. 2.4 и рис. 2.7 можно сделать вывод о том, что, несмотря на различные формы представления денег, большая часть значений успешно преобразовалась. Не преобразовались только последние три значения, которые явно на деньги не похожи, но в реальных таблицах такого рода ”деньги” зачастую имеют место.

В связи с этим необходимы средства, использование которых позволило бы преобразовывать элементы столбцов ИТВР к одному типу. Реализация этих средств разработана в приложении 1.

2.3 Исключение повторяющихся строк в ИТВР

В РТР недопустимо дублирование записей, а в ИТВР это возможно. В связи с этим необходимы средства исключения дублирования записей в ИТВР. Существующий способ [53] в полном объеме не реализован.

2.3.1 Типы повторяющихся строк

Перед разработкой средств исключения дублирования записей в ИТВР рассмотрим типы дублирования. Рассмотрим, как происходит дублирование записей в таблице, представленной на рис.2.8.

	№ Паспорт	Имя	факультет	№ Группа	Предмет	Оценка
	117449	Алексей	ИУ-6	122	БД	5
	224576	Андрей	ЯК-7	233	Алгебра	4
	115779	Анна	ИУ-1	126	КИС	5
	117449	Алексей	ИУ-6	122	БД	5
	114577	Марина	ИУ-3	223	РИС	4
*	0			0		0

Рис.2.8 . Таблица с дублированиями записей

Здесь повторяющиеся записи выделены. В РТР такая ситуация недопустима. Поэтому повторяющиеся записи необходимо оставить в единственном экземпляре.

Чтобы выявить повторяющиеся записи, можно сделать запрос на Microsoft Access, который имеет следующий вид.

```
SELECT First(ТаблицаДуп.[№ Паспорт]) AS [№ Паспорт поле],
First(ТаблицаДуп.Имя) AS [Имя поле], First(ТаблицаДуп.факультет) AS
[факультет поле], First(ТаблицаДуп.[№ Группа]) AS [№ Группа поле],
First(ТаблицаДуп.Предмет) AS [Предмет поле], First(ТаблицаДуп.Оценка) AS
[Оценка поле], Count(ТаблицаДуп.[№ Паспорт]) AS Повторы
```

```
FROM ТаблицаДуп
```

```
GROUP BY ТаблицаДуп.[№ Паспорт], ТаблицаДуп.Имя,
ТаблицаДуп.факультет, ТаблицаДуп.[№ Группа], ТаблицаДуп.Предмет,
ТаблицаДуп.Оценка
```

```
HAVING (((Count(ТаблицаДуп.[№ Паспорт]))>1) AND
((Count(ТаблицаДуп.Оценка))>1));
```


Хотя этот запрос внешне сложен, он прост по сути : из таблицы “ ТаблицаДур” выбираются значения номера паспорта (First(ТаблицаДур.[№ Паспорт])), и этим значениям присваивается имя (AS [№ Паспорт поле]). Выводимые данные группируются по этим значениям (конструкция GROUP BY). С помощью конструкции HAVING выводятся только те строки, в которых имеются повторения (((Count(ТаблицаДур.[№ Паспорт]))>1) AND ((Count(ТаблицаДур.Оценка))>1)).

Результат выполнения запроса на выборку повторяющихся записей представлен на рис.2.9. В отдельном столбце ”Повторы” выводятся количества повторений соответствующих записей в таблице.

	№ Паспорт поле	Имя поле	факультет поле	№ Группа поле	Предмет поле	Оценка поле	Повторы
▶	117449	Алексей	ИУ-6	122	БД	5	2

Рис.2.9. Результат выполнения запроса на выборку повторяющихся записей

В Microsoft Access такой запрос строится с помощью специального мастера ”Повторяющиеся записи”.

Иногда в базах данных происходит дублирование записей другого типа, как в таблице, представленной на рис.2.10.

	№ Паспорт	Имя	Факультет	№ Группа	Предмет	Оценка
▶	117449	Алексей	ИУ-6	122	БД	5
	224576	Андрей	ЯК-7	233	Алгебра	4
	115779	Анна	ИУ-1	126	КИС	5
	117450	Алексей	ИУ-6	222	БД	5
	114577	Марина	ИУ-3	223	РИС	4
*	0			0		0

Рис.2.10. Таблица с дублированием записей

На этом рисунке нетрудно увидеть, что не все записи дублируются. В таблице «ТаблицаДуп1» столбцы «Имя, Факультет, Предмет, Оценка» имеют дублированные записи, а столбец «№ паспорта» не дублируются. Если в столбце «№ паспорта» номер отличается, то такая запись не считается дублированной. Для этой таблицы запрос имеет вид:

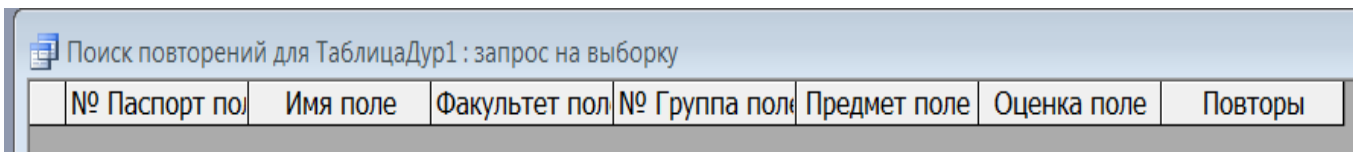
```
SELECT First(ТаблицаДуп1.[№ Паспорт]) AS [№ Паспорт поле],
First(ТаблицаДуп1.Имя) AS [Имя поле], First(ТаблицаДуп1.Факультет) AS
[Факультет поле], First(ТаблицаДуп1.[№ Группа]) AS [№ Группа поле],
First(ТаблицаДуп1.Предмет) AS [Предмет поле], First(ТаблицаДуп1.Оценка)
AS [Оценка поле], Count(ТаблицаДуп1.[№ Паспорт]) AS Повторы
```

```
FROM ТаблицаДуп1
```

```
GROUP BY ТаблицаДуп1.[№ Паспорт], ТаблицаДуп1.Имя,
ТаблицаДуп1.Факультет, ТаблицаДуп1.[№ Группа], ТаблицаДуп1.Предмет,
ТаблицаДуп1.Оценка
```

```
HAVING (((Count(ТаблицаДуп1.[№ Паспорт]))>1) AND
((Count(ТаблицаДуп1.Оценка))>1));
```

В результате выполнения запроса нет дублирования записей, что представлено на рис 2.11.



№ Паспорт поле	Имя поле	Факультет поле	№ Группа поле	Предмет поле	Оценка поле	Повторы

Рис.2.11. Результат выполнения запроса

2.3.2. Исключение повторений строк

Во всех инструментальных СУБД можно сформировать запрос, который формирует таблицу без повторяющихся записей. Он выглядит следующим образом:

```
SELECT DISTINCT ТаблицаДуп.*INTO Без Дуп
```

```
FROM ТаблицаДуп;
```

В этом запросе все поля ТаблицаДуп (ТаблицаДуп.*) добавляются к Таблице «Без Дуп» (INTO Без Дуп) из ТаблицаДуп (FROM ТаблицаДуп). При этом посредством конструкции "DISTINCT" в Таблицу «Без Дуп» добавляются только уникальные значения.

После выполнения запроса Таблица «Без Дуп» примет следующий вид (рис. 2.12).

	№ Паспорт	Имя	факультет	№ Группа	Предмет	Оценка
▶	114577	Марина	ИУ-3	223	РИС	4
	115779	Анна	ИУ-1	126	КИС	5
	117449	Алексей	ИУ-6	122	БД	5
	224576	Андрей	ЯК-7	233	Алгебра	4
*	0			0		0

Рис.2.12. Таблица без повторяющихся записей

В результате получается В таблице нет дублирования записей. Если не возможно переименование исходной таблицы, данный запрос тоже может быть использован, но при этом следует обязательно выполнить дополнительные действия. Они заключаются в удалении исходной таблицы и переименовании полученной таблицы.

2.4. Избавление от сложных заголовков и подзаголовков

В нереляционных таблицах могут встречаться следующие типы подзаголовков: внешние, внутренние и подзаголовки-столбцы. В реляционных таблицах подзаголовки недопустимы. Существующий способ исключения подзаголовков в ИТВ [53] не отражает всех типов подзаголовков и в полном объеме не реализован.

Рассмотрим примеры подзаголовков различных типов.

2.4.1. Задача избавления от подзаголовков

Внешние подзаголовки часто встречаются в справочниках. В качестве примера в таблице 2.1 приведем фрагмент каталога электрооборудования автомобилей (таблица 2.1).

Таблица 2.1

Указатели				Системы зажигания			
Габаритные указатели		Поворотные указатели		Контактные		Бесконтактные	
№	Марка	№	Марка	№	Марка	№	Марка

Как видно из структуры таблицы, она имеет заголовки трех уровней: основной заголовок, подзаголовок первого уровня и подзаголовки второго уровня. В реляционных таблицах допустимы заголовки только одного уровня. Для ИТВР в общем случае справедливы следующие утверждения:

$$T = \{Z_1, Z_2, \dots, Z_K, \dots, Z_N\}, \quad Z_K = \{P_1 Z_K, P_2 Z_K, \dots, P_L Z_K, \dots, P_F Z_K\}$$

$$P_L Z_K = \{P_{11} P_L Z_K, P_{21} P_L Z_K, \dots, P_{Q1} P_L Z_K, \dots, P_{PP} P_L Z_K\},$$

Где T – схема таблицы;

Z_K – схема K -того заголовка таблицы T ;

$P_L Z_K$ – L -ый подзаголовок первого уровня Z_K -го заголовка таблицы T ;

$P_Q P_L Z_K$ – Q -ый подзаголовок второго уровня.

Следует обратить внимание на то, что число уровней подзаголовков может быть и большим. Но, как показывает анализ доступных документов различных предметных областей, обычно задействуют два уровня подзаголовков. Это видно из примера, приведенного во фрагменте таблицы крепежных деталей (таблица 2.2).

Таблица 2.2

Болты		Шайбы		Винты				Штифты		Гайки		Шплинты
б-и	Цил	Плос	Гра	с	с	с	с	Прос	Кону	Осно	Коро	
гран	инд			че	по	по	ци					
ные	рич			че	лу	та	ли		раз			
	еск			ви	сф	йн	нд		ные			
	ие			чн	ер	ой	ри					
	с			ой	ич	го	че					
	вну			го	ес	ло	ск					
	тре			ло	ой	вк	ой					
	нним			вк	го	ой	го					
	б-й			ой	ло		ло					
	гран				вк		вк					
	ни				ой		ой					
	ком											

Для этой таблицы могут быть задействованы заголовки более высокого уровня. Их схема следующая:

Нормаль = {гайки, болты, винты, шайбы}.

Фиксирующие детали = {штифты, шплинты, клинья}.

В обозначениях столбцов типа PQPL3K заложены индексы, с помощью которых могут быть построены циклы, содержащиеся в алгоритмах исключения внешних заголовков. Так, например, для основного цикла могут быть задействованы индексы K и N, для 1-го внутреннего цикла индексы L и K, для 2-ого внутреннего индексы Q и P.

Чтобы исключить сложные заголовки и привести таблицу к реляционному виду, необходимо организовать сканирование подзаголовков самого нижнего уровня и для каждого подзаголовка осуществить сбор всей относящейся к нему информации из всех заголовков более высокого уровня. Затем собранную информацию необходимо использовать в качестве неделимого (атомарного) заголовка.

Например, 1-й полученный таким образом заголовок таблицы 2 будет выглядеть следующим образом: N шестигранные болты или болты шестигранные N. Второй заголовок может быть таким: болты шестигранные тип. Далеко не всегда заголовки, полученные таким образом, могут быть воспринимаемы потенциальным пользователем базы данных. Более того, их длина может превышать максимально допустимую длину атрибутов, используемых инструментальных средств. В связи с этим процедура формирования атомарных столбцов должна быть не автоматической, а автоматизированной. То есть пользователь для исключения внешних подзаголовков должен иметь возможность вмешаться в процесс формирования заголовков столбцов с целью присвоения атомарным столбцам приемлемых имен. Один из возможных алгоритмов исключения внешних подзаголовков приведен в работе [53].

Следует обратить внимание на то, что исходная информация может быть представлена различными способами: на бумаге, в виде текстовых файлов, в формате электронных таблиц и др. Удобнее всего для разработчика средств преобразования было бы использование в качестве исходного единый формат данных. Например, в качестве основного – использование формата Excel, а все другие представления данных преобразовывать в данный формат. Но, к сожалению, не всегда это возможно. Например, табличные данные, представленные на бумаге, далеко не всегда удается отсканировать в формат электронных таблиц.

Поэтому необходимо предусмотреть реализацию алгоритма не только для информации табличного вида, представленной в формате .xls, но и в формате .txt.

2.4.2. Исключение внутренних подзаголовков

Подзаголовки такого рода часто встречаются в прайс-листах, в номенклатуре товаров, то есть в тех случаях, когда структуры нескольких таблиц совпадают, но их данные относятся к различным группам. Группировка может осуществляться, например, по датам, категориям товаров, регионам. В качестве примера рассмотрим список проданных за день товаров магазина «Соки - воды» (таблица 2.3).

Таблица 2.3

Название	Объем	Цена	Количество
Газированная вода			
Тархун	1л	30р	7
Байкал	2л	60р	5
Колокольчик	1.5л	40р	10
Соки			
Вишневый	1л	45р	3
Ананасовый	1л	45р	15
Яблочный	1л	45р	8
Морсы			
Клюквенный	1.5л	60р	7
Малиновый	1.5л	60р	7
Рябиновый	1л	45р	7
...			

Как видно из примера, таблица легко воспринимается визуально. Однако в таком виде она неприменима в составе баз данных. Такая таблица с небольшим числом строк без особых усилий может быть преобразована в

реляционную таблицу вручную. Но в реальных таблицах может быть тысячи или более строк. В этом случае их преобразование невозможно. Необходимы специальные автоматизированные средства. В работе [53] вообще не идет речи о внутренних заголовках.

В случае если таблица 2.3 будет импортирована в какую-либо систему управления базами данных, она примет вид таблицы 2.4.

Таблица 2.4

Название	Объем	Цена	Количество
Газированная вода			
Тархун	1л	30р	7
Байкал	2л	60р	5
Колокольчик	1.5л	40р	10
Соки			
Вишневый	1л	45р	3
Ананасовый	1л	45р	15
Яблочный	1л	45р	8
Морсы			
Клюквенный	1.5л	60р	7
Малиновый	1.5л	60р	7
Рябиновый	1л	45р	7

Как видно из таблицы 2.4, смысл импортированной таблицы утрачен. В частности, из таблицы следует, что напиток Газированная вода не имеет объема, цены и не продавался. Хотя на самом деле все обстоит по-другому. Необходимо избавиться от противоречий такого рода. Для этого необходимо выполнить следующие действия:

- выявить внутренние подзаголовки;

- присвоить номер очередному напитку;
- сформировать новую таблицу напитков с соответствующими номерами;
- исключить записи с категориями напитков из исходной таблицы;
- сформировать реляционную таблицу или таблицы таким образом, чтобы сохранить смысл исходных таблиц.

Новый заполненный столбец с номерами напитков приведен в таблице 2.5.

Таблица 2.5

№	Название	Объем	Цена	Количество
1	Газированная вода			
1	Тархун	1л	30р	7
1	Байкал	2л	60р	5
1	Колокольчик	1.5л	40р	10
2	Соки			
2	Вишневый	1л	45р	3
2	Ананасовый	1л	45р	15
2	Яблочный	1л	45р	8
3	Морсы			
3	Клюквенный	1.5л	60р	7
3	Малиновый	1.5л	60р	7
3	Рябиновый	1л	45р	7

В сформированной таблице исключатся записи с напитками из исходной таблицы (табл. 2.6 и табл. 2.7) .

Таблица 2.6

№	Напиток
1	Газированная

	Вода
2	Соки
3	Морсы

Таблица 2.7

№	Название	Объем	Цена	Количество
1	Тархун	1л	30р	7
1	Байкал	2л	60р	5
1	Колокольчик	1.5л	40р	10
2	Вишневый	1л	45р	3
2	Ананасовый	1л	45р	15
2	Яблочный	1л	45р	8
3	Клюквенный	1.5л	60р	7
3	Малиновый	1.5л	60р	7
3	Рябиновый	1л	45р	7

Для таблиц этого вида можно строить запросы, на основе связи "один ко многим"

Предлагается машинный алгоритм преобразования.

Для описания алгоритм используется общий вид отношения, представленный в табл. 2.8.

Таблица 2.8

A_1	A_i	A_k
a_{11}	NULL	NULL
a_{21}	a_{2i}	a_{2k}

.....
a_{j1}	NULL	NULL
.....
a_{f1}	NULL	NULL
a_{m1}	a_{mi}	a_{mk}

Таблица такого рода – это таблица, в некоторых строках которой значение имеет только один атрибут, который является внутренним подзаголовком таблицы.

Неформальный алгоритм исключения внутренних подзаголовков содержит следующие действия.

П1: Каждая запись проверяется на наличие в ней только одного значения атрибута. Записи такого рода подсчитываются. Если таких записей несколько, то подзаголовки в отношении R присутствуют и переходят к следующему пункту (П2).

П2: К отношению R присоединится дополнительный атрибут KR с типом «числовой» (формируется R0).

CNT :=0;

П3: Сканируются все записи отношения R0. Если записи имеют только одно заполненное значение атрибута, то счетчик подзаголовков CNT увеличивается на 1.

Значению атрибута KR присваивается значение CNT.

П4: Создается новое отношение R2, включающее в себя 3-а атрибута FR и атрибут с подзаголовком.

Сканируются все записи отношений R0. Записи, которые имеют только одно (кроме ключевого) заполненное значение, атрибут перемещается в отношение R2.

На последней стадии алгоритма формируются отношение R2 (с подзаголовками исходной таблицы) и отношение R1 без подзаголовков.

Связи между таблицами объединяются ключевыми атрибутами KR, которые присутствуют в обоих отношениях.

Формализованный алгоритм исключения внутренних подзаголовков выглядит следующим образом.

```
CNT = 0
```

```
DO s = 1 TO m
```

```
  CNT1 = 0
```

```
  DO f = 1 TO k
```

```
    IF ask = NULL THEN CNT1 = CNT1 + 1
```

```
  NEXT f
```

```
  IF CNT1 = k-1 THEN CNT = CNT + 1
```

```
NEXT s
```

```
IF CNT < 2 THEN EXIT
```

Формирование двух отношений

```
R0 = R(A1, ..., Ai, ..., Ak) + R(KR)
```

```
CNT = 0
```

```
DO s = 1 TO m
```

```
  CNT1 = 0
```

```
  DO f = 1 TO k
```

```
    IF ask = NULL THEN CNT1 = CNT1 + 1
```

```
  NEXT f
```

```
  IF CNT1 = k-1 THEN
```

```
    CNT = CNT + 1
```

```
    C(R2,CNT,1) = CNT
```

```
    C(R2,CNT,2) = ask
```

```
    DELETE * FROM R0 WHERE (A1 = ask)
```

```
  ELSE
```

```
    C(R0,s,1) = CNT
```

```
  END IF
```

```
NEXT s
```

Здесь m – мощность R .

k – степень R .

Выражение $R0 = R(A_1, \dots, A_i, \dots, A_k) + R(KR)$ означает добавление к R атрибута с именем KR .

Выражение $C(R2_{CNT,1})$ означает значение элемента $R2$ в строке CNT и 1-ом столбце.

Выражение $C(R0_{s,1})$ означает значение элемента $R0$ в строке s и 1-ом столбце.

В таблице приведен пример с внутренними подзаголовками сложных атрибутов. Рассмотрим таблицу с использованием стандартных средств Microsoft Access, представленную на рис. 2.13.

	Название	Объем	Цена	Количество
	Газированная вода			
	Тархун	1 л	\$30.00	7
	Байкал	2 л	\$60.00	5
	Колокольчик	1.5 л	\$40.00	10
	Соки			
	Вишневый	1 л	\$45.00	3
	Ананасовый	1 л	\$45.00	15
	Яблочный	1 л	\$45.00	8
	Морсы			
	Клюквенный	1.5 л	\$60.00	7
	Малиновый	1.5 л	\$60.00	7
	Рябиновый	1 л	\$45.00	7
			\$0.00	0

Рис.2.13. Таблица с внутренними подзаголовками

Нетрудно заметить, что внутренние заголовки «Газированная Вода, Соки, Морсы» стоят в столбце с заголовком «Название».

В базах данных такая таблица невозможна в связи с наличием других таблиц и поэтому надо исключить такие внутренние заголовки. Для приведения таблицы к допустимому виду можно в режиме «Конструктора» определить новый столбец «Напиток», а потом в режиме «Просмотра» этот столбец

вручную заполнить и можно удалить внутренние подзаголовки. Результат такого преобразования приведен на рис. 2.14.

	Название	Объем	Цена	Количество	Напиток
▶	Тархун	1 л	\$30.00	7	Газированная вода
	Байкал	2 л	\$60.00	5	Газированная вода
	Колокольчик	1.5 л	\$40.00	10	Газированная вода
	Вишневый	1 л	\$45.00	3	Соки
	Ананасовый	1 л	\$45.00	15	Соки
	Яблочный	1 л	\$45.00	8	Соки
	Клюквенный	1.5 л	\$60.00	7	Морсы
	Малиновый	1.5 л	\$60.00	7	Морсы
	Рябиновый	1 л	\$45.00	7	Морсы
*			\$0.00	0	

Рис. 2.14. Преобразованная исходная таблица

Теперь таблица «ТаблицаВнутр» примет приемлемый вид. Реальные таблицы нередко включает тысячи и более записей. Тогда их преобразование трудоемко и может привести к ошибкам.

2.4.3. Способ избавления от сложных заголовков и подзаголовков

ИТВР часто представлена таким образом, что в области заголовков имеют место заголовки, которые включают в себя несколько позиций. Такого рода ситуации в работе [53] не рассматриваются. Например, какой-либо заголовок может быть представлен следующим образом: континент, часть света, страна. Таблица такого рода не является реляционной и преобразование ее к реляционному виду задача нетривиальная. Более того, проблема преобразования может существенно усложниться в связи с тем, что такие заголовки влекут за собой необходимость использования сложных заголовков. Например, таблица 2.9.

Таблица 2.9

Континент, часть света, страна		Количество крупных городов	Количество крупных рек
Европа			
Восток	Запад		

Россия		33	26
	Испания	8	14

И в первом и во втором случаях нарушена первая нормальная форма – атрибуты реляционных таблиц должны быть неделимы. Кроме того, такого рода подзаголовки могут встречаться внутри таблицы. На рисунке приведен пример такого рода таблицы (таблица 2.10).

Таблица 2.10

Континент, часть света, страна		Количество крупных городов	Количество крупных рек
Европа			
Восток	Запад		
Россия		33	26
	Испания	8	14
Азия			
Восток	Запад		
Мьянма		14	4
	Индия	26	8

Такую таблицу невозможно обрабатывать с помощью языка запросов.

В связи с этим эту таблицу оправданно представить в виде 2-х связанных реляционных таблиц: «Части света. Континенты» и «Страны». Но в этой таблице существуют сложные атрибуты, поэтому надо избавляться от сложных атрибутов, избавляться от подзаголовков, которые попали в значения атрибутов. В таблице 2.11 показаны эти изменения.

Таблица 2.11

Континент, часть света, страна		Количество крупных городов	Количество крупных рек
Восточная Европа	Западная Европа		
Россия		33	26

	Испания	8	14
Восточная Азия	Западная Азия		
Мьянма		14	4
	Индия	26	8

Предлагается следующая последовательность действий. Формируется новый столбец с номерами частей света, континентов. Сканируется преобразованная таблица, очередной части света, континенту присваивается номер и этот номер распространяется на страны. Но эта таблица не очень удачна для 2-х связанных реляционных таблиц: «Части света. Континенты» и «Страны». Поэтому предлагается следующий вариант – таблица 2.12.

Таблица 2.12

Континент, часть света, страна	Количество крупных городов	Количество крупных рек
Восточная Европа		
Россия	33	26
Западная Европа		
Испания	8	14
Восточная Азия		
Мьянма	14	4
Западная Азия		
Индия	26	8

Результат формирования нового столбца с номерами части света, континентов и заполнением столбца приведен в таблице 2.13.

Таблица 2.13

№	Континент, часть света, страна	Количество крупных городов	Количество крупных рек
1	Восточная		

	Европа		
1	Россия	33	26
2	Западная Европа		
2	Испания	8	14
3	Восточная Азия		
3	Мьянма	14	4
4	Западная Азия		
4	Индия	26	8

Результат формирования новой таблицы «Части света. Континенты» и исключения записей с «частью света, континентов» из исходной таблицы приведены соответственно в таблице 2.14. В таблице 2.15 - Страны.

Таблица 2.14

№	Континент, часть света
1	Восточная Европа
2	Западная Европа
3	Восточная Азия
4	Западная Азия

Таблица 2.15

№	страна	Количество крупных городов	Количество крупных рек
1	Россия	33	26
2	Испания	8	14
3	Мьянма	14	4
4	Индия	26	8

Для таблиц данного вида можно строить реляционные запросы.

Затем из записей с «частью света, континентов» формируется новая таблица, соответствующие записи из исходной преобразованной таблицы удаляются. После этих преобразований будут сформированы 2-е таблицы, связанные между собой связью типа 1: ∞.

Для рассмотренного примера описанные манипуляции вполне можно выполнить вручную. Но для реальных таблиц мощностью десятки тысячи записей это затруднительно. Поэтому необходима разработка алгоритма автоматизированного избавления от сложных атрибутов при преобразовании заполненных нереляционных таблиц к реляционному виду.

Предварительно представим (отношение) рассматриваемого типа в общем виде (таблица 2.16).

Таблица 2.16

A_1	A_2	A_i	A_k
a_{11}	NULL	NULL	NULL
a_{21}	a_{22}	NULL	NULL
a_{31}	NULL	a_{3i}	a_{3k}
NULL	a_{42}	a_{4i}	a_{4k}
a_{j1}	NULL	NULL	NULL
.....
a_{f1}	NULL	NULL	NULL
a_{m1}	a_{m2}	a_{mi}	a_{mk}

Особенность таблицы такого рода состоит в том, что в некоторых ее строках значение имеет один или два атрибута. Принимается, что такой атрибут является внутренним сложным подзаголовком таблицы.

Предлагается укрупненный алгоритм исключения сложных подзаголовков:

П1: Выполняется перебор всех записей отношения R. Каждая запись проверяется на наличие в ней только одного значения атрибута. Записи

такого рода подсчитываются. Если таких записей несколько, то подзаголовки в отношении R имеют место. Осуществляется переход к следующему пункту (П2). В противном работа алгоритма завершается.

П2: Избавление от сложных атрибутов:

r=1;

DO r=1 to m

DO i=1 to 2

WHILE $a_{1i} \neq a_{2i}$

$a_{1i} := \text{Concat}(a_{1i}, ', ', a_{2i})$

END;

END;

П3: Для 2-х связанных реляционных таблиц:

k=0;

i=1;

j=1;

r=1;

WHILE $a_{rj} \neq \text{END FILE}$

WHILE $a_{ij} \neq \text{Подзаголовок}$

IF k=0 THEN r=1

k=1;

$a_{1ij} := a_{ij}$;

i:=i+1;

END WHILE;

k:=0;

IF j=1 THEN j:=2;

ELSE j:=1;

END WHILE;

П4: К отношению R приписывается атрибут KR с типом «числовой».

CNT:=0;

П5: Выполняется перебор всех записей отношения R' . Если в строке имеется только одно значение атрибута, то счетчик подзаголовков CNT увеличивается на 1.

Значению атрибута KR присваивается значение CNT.

П6: Создается новое отношение R2, которое включает в себя 2-а атрибута NR и атрибут с подзаголовком.

Выполняется сканирование всех записей отношений R' . Записи, которые имеют только одно (кроме ключевого) заполненное значение атрибута перемещаются в отношение R2.

Результатом работы алгоритма является отношение R2 (с подзаголовками анализируемой таблицы) и отношение R1, в котором отсутствуют подзаголовки. Взаимосвязь между таблицами реализуется с помощью ключевых атрибутов KR.

Алгоритм исключения внутренних подзаголовков и избавления от сложных атрибутов при преобразовании нереляционных таблиц к реляционному виду выглядит следующим образом.

```

k=0;
j=1;
f=1;
DO f=1 to m
  i=1;
  DO i=1 to 2
    WHILE a1i <> a2i;
    a1i := Concat (a1i, ' ', a2i);
  END
WHILE afj <> END FILE
  WHILE aij <> Подзаголовок
    IF k=0 THEN f=1; k=1;
    a1ij := aij;
    i := i+1;
  
```

```

END WHILE;
k:=0;
IF j=1 THEN j:=2;
END WHILE;
CNT=0;
DO f=1 to m
  CNT1=0;
  DO f=1 to p
    IF afp=NULL THEN CNT1=CNT1+1;
  END f
  IF CNT1=p-1 THEN CNT=CNT+1;
END f
IF CNT < 2 THEN EXIT
FEM Формирование двух отношений
F2=F(A1,.....,Ai,.....,Ap)+F(KF)
CNT=0
DO f=1 to m
  CNT1=0;
  FOF f=1 to p
    IF afp=NULL THEN CNT1=CNT1+1;
  END f
  IF CNT1 =p-1 THEN
    CNT=CNT+1;
    Z(F2CNT,1)=CNT;
    Z(F2CNT,2)=afp;
    DELETE * FFOM F' WHERE (A1=afp);
  ELSE
    Z(Ff,1)=CNT;
  END IF
END f;

```

Здесь:

m – мощность отношения F;

P – степень отношения F;

END FILE - конец таблицы;

Подзаголовок – заголовок 2-го уровня;

F' = F + F(KF) - конкатенация атрибута F и имени KF;

Z(F_{2CNT,1}) - значение элемента F₂ в строке CNT и 1-м столбце;

Z(F'f,1) - значение элемента F' в строке f и 1-м столбце.

Часто оправдано использовать существующие инструментальные средства для избавления от сложных атрибутов и подзаголовков в заполненной таблице. Рассмотрим пример на рис. 2.15, где исходная таблица сформирована в Microsoft Excel.

	A	B	C	D
1	Континент, часть света, страна		Количество крупных городов	Количество крупных рек
2	Восточная Европа	Западная Европа		
3	Россия		33	26
4		Испания	8	14
5	Восточная Азия	Западная Азия		
6	Мьянма		14	4
7		Индия	26	8
8				
9				
10				

Рис.2.15. Исходная таблица с избавлением от сложных атрибутов и подзаголовков

В таблице имеется сложный атрибут – «Континент, часть света, страна». Выполним импорт этой таблицы в СУБД Microsoft Access. Для импорта ее используется меню «Файл/ Внешние данные/Импорт». В процессе выполнения шагов мастера указывается лист рабочей книги Microsoft Excel, назначается строка заголовки, имя создаваемой таблицы. Окна мастера на его очередных шагах имеют виды рисунки 2.16 - 2.18.

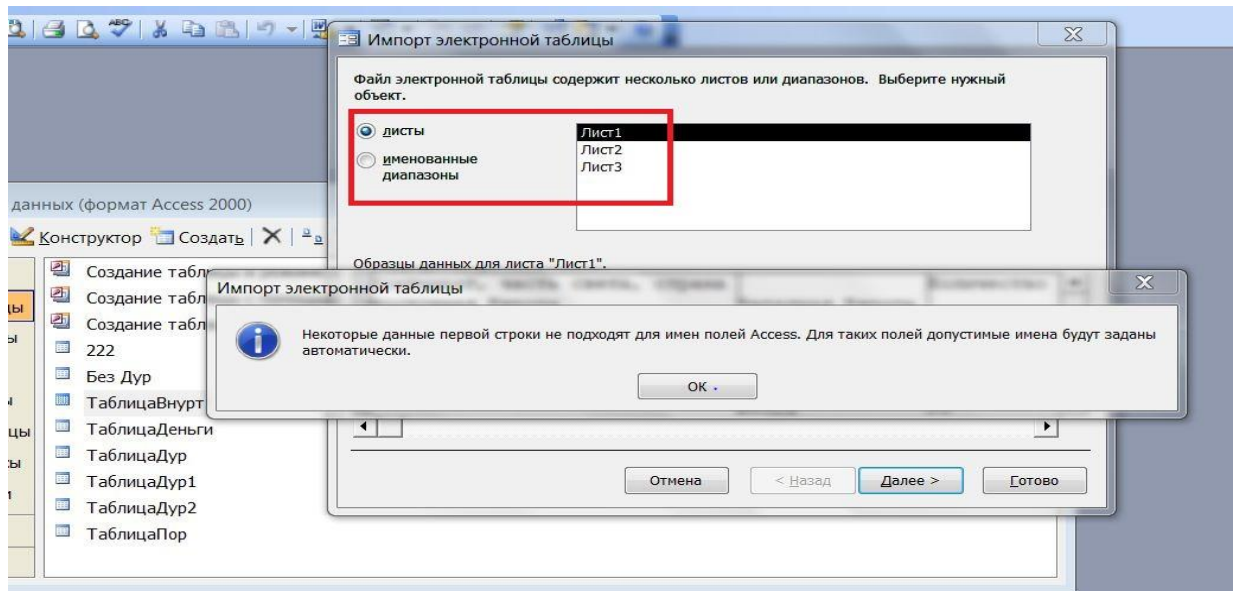


Рис. 2.16. Окно мастера импорта указания листа

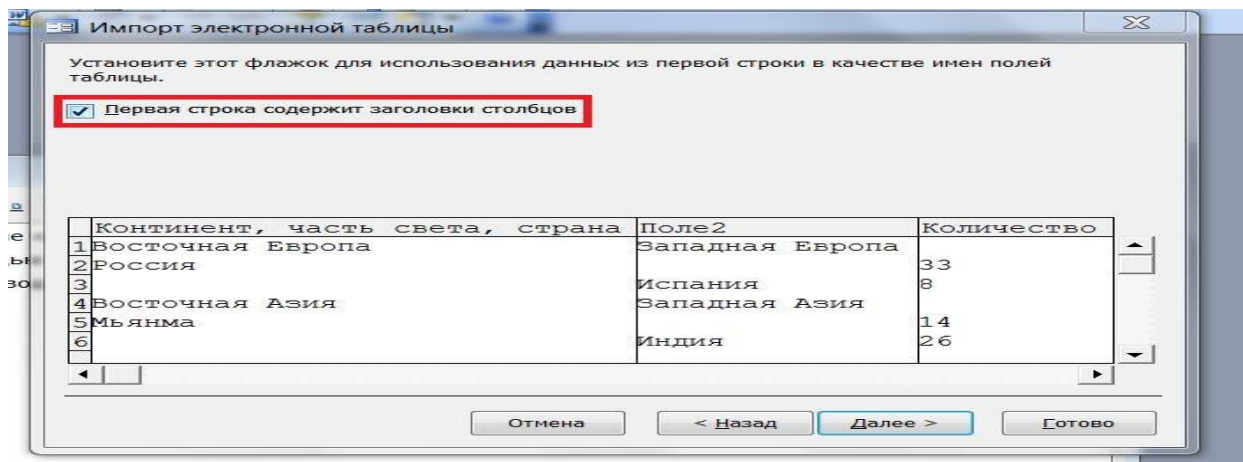


Рис. 2.17. Окно назначения строки заголовков

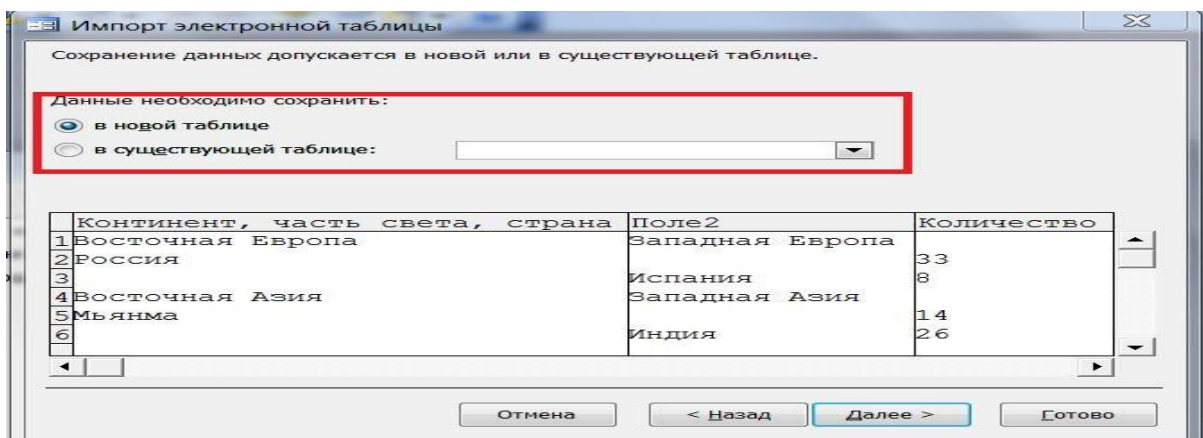


Рис. 2.18. Окно создания имени таблицы

В результате выполнения всех шагов мастера исходная таблица в формате Microsoft Access примет такой вид (рис.2.19).

	Континент, часть света, страна	Поле2	Количество крупных городов	Количество крупных рек
▶	Восточная Европа	Западная Евро		
	Россия		33	26
		Испания	8	14
	Восточная Азия	Западная Азия		
	Мьянма		14	4
		Индия	26	8
*				

Рис. 2.19. Исходная таблица в формате Microsoft Access

Как видно из рис. 2.19, в таблице нет заголовка второй строки, так как в формате Microsoft Access невозможно импортировать сложные атрибуты и подзаголовки. Такая таблица неприемлема для использования. Для избавления от сложных атрибутов и подзаголовков необходимо, в соответствии с алгоритмом, сформировать простые заголовки и избавиться от подзаголовков, которые попали в значения атрибутов.

Но эта таблица не очень удачна для связанных реляционных таблиц: «Части света. Континенты» и «Страны». Поэтому надо редактировать таблицу в режиме «Конструктора».

После выполнения необходимых действий в режиме «Конструктора» и в режиме «Просмотра» таблица примет такой вид (рис. 2.20):

	Континент, часть света, страна	Количество крупных городов	Количество крупных рек
	Восточная Европа		
	Россия	33	26
	Западная Европа		
	Испания	8	14
	Восточная Азия		
	Мьянма	14	4
	Западная Азия		
	Индия	26	8
▶		0	0

Рис. 2.20. Преобразованная таблица в формате Microsoft Access

Потом формируется новый столбец с номерами частей света, континентов с помощью «Конструктора» (рис. 2.21):

Континент, часть света, страна	Количество крупных городов	Количество крупных рек	№
Восточная Европа			1
Россия	33	26	1
Западная Европа			2
Испания	8	14	2
Восточная Азия			3
Мьянма	14	4	3
Западная Азия			4
Индия	26	8	4
	0	0	0

Рис. 2.21. Таблица, полученная после добавления нового столбца

Результат формирования новой таблицы «Части света. Континенты» и исключения записей с «частью света, континентов» из исходной таблицы приведены соответственно на рис. 2.22.

№	Страна	Количество крупных городов	Количество крупных рек
1	Россия	33	26
2	Испания	8	14
3	Мьянма	14	4
4	Индия	26	8
0		0	0

Континент, часть света, страна	№
Восточная Европа	1
Западная Европа	2
Восточная Азия	3
Западная Азия	4

Рис. 2.22. Раздельные таблицы «Континент, часть света» и «Страна».

Для таблиц данного вида можно строить реляционные запросы, чтобы связь между собой связью типа 1: 1. В формате Microsoft Access можно создать такой запрос с помощью меню Запросы/ Конструктор/ Создать. Потом разработчик получит схему данных, представлен на рис. 2.23.

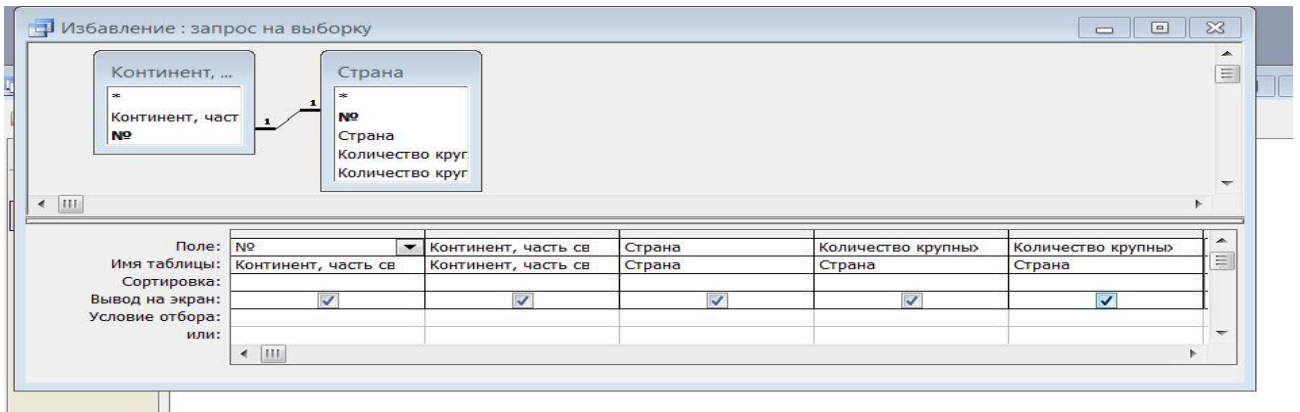


Рис. 2.23. Схема данных, полученная после создания запроса

Как видно из схемы данных, таблица «Континент, часть света, страна» и «Страна» связаны между собой связью типа 1: 1 использования ключевого поля «№». Этот запрос имеет следующий вид на Microsoft Access.

```
SELECT [Континент, часть света, страна].№, [Континент, часть света,
страна].[Континент, часть света, страна], Страна.Страна, Страна.[Количество
крупных городов], Страна.[Количество крупных рек]
```

```
FROM [Континент, часть света, страна] INNER JOIN Страна ON
[Континент, часть света, страна].№ = Страна.№;
```

Этот запрос выбирает все строки таблиц с помощью конструкции (SELECT). Если номер таблицы ([Континент, часть света, страна].№) равно номер таблицы (Страна.№), то эти таблицы связаны между собой внутренней связью с помощью конструкции (INNER JOIN).

Результат выполнения этого запроса представлен на рис. 2.24.

№	Континент, часть света, страна	Страна	Количество крупных городов	Количество крупных рек
1	Восточная Европа	Россия	33	26
2	Западная Европа	Испания	8	14
3	Восточная Азия	Мьянма	14	4
4	Западная Азия	Индия	26	8
*				

Рис. 2.24. Результат выполнения запроса

Как видно из рис.2.24 в результате выполнения запроса получена таблица без сложных атрибутов и подзаголовков

Суть метода преобразования заполненных нереляционных таблиц в реляционные таблицы состоит в преобразовании разработанной модели ИТВР в модель РТР в соответствии с предложенным комплектом способов, алгоритмов и методикой их использования.

Другими словами, способ состоит в контекстном анализе значений таблиц ИТВР, выявлении их несоответствий сформулированным требованиям к РТР и исключению этих несоответствий на основе применения предложенной формализации.

Выводы по главе 2.

1. Разработаны полные адекватные модели ИТВ и РТ (ИТВР и РТР). Эти модели отражают концептуальные особенности соответствующих объектов, которые ранее отражены не были.

2. Разработан метод преобразования объектов, соответствующих ИТВР в объекты, соответствующие РТР. Метод позволяет решать задачи преобразования не в ручном, а в автоматизированном режиме и сократить время преобразования в десятки раз.

3. Предложенный метод является одной из основных компонент методики формирования реляционных таблиц на основе использования существующих, заполненных нереляционных таблиц.

3. МЕТОД НАЗНАЧЕНИЯ КЛЮЧЕЙ В ИТВР

В третьей главе разработан метод назначения ключевых полей в ИТВР, который по сравнению с известным методом позволяет учитывать требования к введенным моделям ИТВР и РТР. В частности, в отличие от [53], в модели РТР предусмотрено наличие первичных и внешних ключей. Кроме того метод в полном объеме учитывает требование к минимальности ключей.

3.1 Задача формирования ключей в ИТВР

В работах [62] предложен способ назначения первичных ключей в ИТВ. Но в нем учтены не все особенности ключевых полей. В частности:

- рассматривается возможность включения в первичный ключ только 2-х атрибутов;
- не в полном объеме учитываются требования минимальности первичного ключа;
- не освещены вопросы назначения внешних ключей
- назначение ключевых полей не рассматривается как неотъемлемая задача преобразования ИТВ в РТ.

Как сформулировано в 1-й главе основными требованиями к первичным ключам являются уникальность и минимальность. Формализуем эти требования, а затем используем их в качестве целевых функций при разработке соответствующих алгоритмов.

Уникальность. Пусть имеется отношение R:

$R=(A_1, \dots, A_j, \dots, A_m, \dots, A_k), i = \overline{1, k}$, где k – степень отношения; A_i – атрибут отношения.

$A_i = \{e_{i_1}, \dots, e_{i_j}, \dots, e_{i_n}\} \quad j = \overline{1, n}$, где n – мощность отношения, e_{i_j} – j -й элемент атрибута A_i .

$A_m = \{e_{m_1}, \dots, e_{m_j}, \dots, e_{m_n}\} \quad j = \overline{1, n}$, где n – мощность отношения, e_{m_j} – j -й элемент атрибута A_m

Необходимо найти такие атрибуты A_i, \dots, A_m чтобы обеспечилась истинность выражения:

$$\text{concat}(e_{i_1}, \dots, e_{m_1}) \neq, \dots, \neq \text{concat}(e_{i_j}, \dots, e_{m_j}) \neq, \dots, \neq \text{concat}(e_{i_n}, \dots, e_{m_n}) \quad [62]. \quad (1)$$

Из выражения (1) следует, что нужно найти такое сочетание атрибутов, чтобы сцепление их значений было уникально. Причем:

- кортеж атрибутов может включать несколько атрибутов;
- число возможных сочетаний может быть большим, что зависит от степени отношения;
- может не найтись таких атрибутов, которые обеспечивают истинность выражения (1).

При назначении первичных ключей в рамках существующей методологии проектирования РБД исходят из визуального анализа предполагаемых схем отношений. Такой подход не всегда приводит к успеху. Ведь в схеме отношения могут быть погрешности.

При наличии ИТВР разработчик понимает смысловое назначение таблицы, а главное – имеет возможность анализа реальных данных. В этом случае процесс формирования первичного ключа формализуем.

Минимальность. Минимальность ключевого поля, в отличии от [62] рассматривается в рамках 2-х требований.

В первом случае экономиться память, которая необходима для хранения значений атрибутов, которые входят в первичный ключ. В связи с этим самая очевидная целевая функция – минимальное число атрибутов, входящих в первичный ключ:

$\min |A_1, \dots, A_i, \dots, A_m, \dots, A_k|$, где $i = \overline{1, k}$, где k – число атрибутов, входящих в первичный ключ; A_i – атрибут отношения, входящий в первичный ключ.

Строго говоря, более правильная целевая функция следующая:

$$\min(\text{Length}(A_i) + \dots + \text{Length}(A_k))$$

Минимальность ключей определяется не только количеством атрибутов, входящих в них, но и суммарной длиной этих атрибутов. А длина атрибутов в основном определяется их типом и свойствами.

При наличии только схемы отношения непросто выбрать атрибуты, которые составят первичные ключ. При недостаточном знании предметной области можно ошибиться, т.к. после заполнения таблицы реальная длина данных может быть не равна предполагаемой.

Используя ИТВР, разработчик РБД имеет возможность принимать решение на основе существующих данных. Кроме того, процесс поиска минимальных ключей можно формализовать.

Во втором случае под минимальностью первичного ключа подразумевается отсутствие в ключе атрибутов, значения которых уникальны [93]. Пусть первичный ключ K представлен множеством атрибутов:

$K = (A_1, \dots, A_i, \dots, A_j, \dots, A_k)$, $i = \overline{1, k}$, где k – число атрибутов, входящих в первичный ключ;

A_i – i -й атрибут отношения, входящий в первичный ключ.

$S_1 = (a_{1_1}, \dots, a_{1_i}, \dots, a_{1_j}, \dots, a_{1_k})$, где S_1 – часть 1-й записи отношения, соответствующая набору атрибутов, входящих в первичный ключ.

$S_n = (a_{n_1}, \dots, a_{n_i}, \dots, a_{n_j}, \dots, a_{n_k})$, где S_n – часть n -й записи отношения, соответствующая набору атрибутов, входящих в первичный ключ.

$S_m = (a_{m_1}, \dots, a_{m_i}, \dots, a_{m_j}, \dots, a_{m_k})$, где S_m – часть m -й, последней записи отношения, соответствующая набору атрибутов, входящих в первичный ключ.

m – мощность отношения;

a_{n_j} – значение j -го атрибута n -й записи.

Тогда, для ключевого поля, которое включает в себя несколько атрибутов должно выполняться условие:

$$\neg((a_{1_1} \neq, \dots, \neq a_{n_1} \neq \dots \neq a_{m_1}) \vee (a_{1_i} \neq, \dots, \neq a_{n_i} \neq \dots \neq a_{m_i}) \vee (a_{1_j} \neq, \dots, \neq a_{n_j} \neq \dots \neq a_{m_j}) \vee (a_{1_k} \neq, \dots, \neq a_{n_k} \neq \dots \neq a_{m_k}))$$

На основе сформулированных целевых функций ниже предлагаются неформальные и формальные алгоритмы назначения первичных ключей в заполненных таблицах.

3.2 Алгоритмы назначения первичных ключей в ИТВ

3.2.1 Неформальные алгоритмы назначения первичных ключей в ИТВР

Важно отметить, что в качестве ключей, а также в качестве атрибутов, которые входят в первичный ключ, не рассматриваются атрибуты, которые имеют тип логический, MEMO, LOB, BLOB, поле объекта OLE. В этой связи атрибуты данного типа необходимо исключить из рассмотрения.

П1. Поиск единственного атрибута, все значения которого уникальны.

$MKA = \emptyset$, где MKA – множество ключевых атрибутов, которые претендуют на роль первичного ключа.

Пусть имеется отношение R :

$R = (A_1, \dots, A_i, \dots, A_m, \dots, A_k)$, $i = \overline{1, k}$, где k – степень отношения; A_i – атрибут отношения.

$A_i = (e_{i_1} \dots e_{i_m})$, где e_{i_1} – 1-й элемент домена с атрибутом A_i , e_{i_m} – m -й элемент домена с атрибутом A_i .

Выполняется поиск такого атрибута A_i такого, что $e_{i_1} \neq \dots \neq e_{i_m}$.

В связи с этим перебираются все атрибуты отношения.

Если такой атрибут находится, то он запоминается: $MKA = MKA + A_i$

Если после перебора всех атрибутов $MKA = \emptyset$, то это означает, что назначить первичный ключ, включающий в себя один атрибут невозможно.

В этой связи разработчик должен принять одно из решений: назначить суррогатный ключ или сформировать первичный ключ на базе нескольких атрибутов. Если принято 2-е решение, то переход к П.2

Если $MKA \neq \emptyset$, то атрибуты с уникальными значениями нашлись и необходимо выбрать из них атрибут, удовлетворяющий требованиям минимальности. Так как на роль ключевого атрибута претендует только один из найденных атрибутов, то необходима проверка только первой части требования минимальности длины атрибута. В связи с этим необходимо найти в множестве MKA атрибут с минимальной длиной т.е. $\min(\text{Length}(A_i), \dots, \text{Length}(A_k))$, где $(A_i, \dots, A_k) \in MKA$. Таким образом, найден первичный ключ. STOP.

Несмотря на то, что найден атрибут минимальной длины, разработчику должен быть предоставлен весь список возможных ключевых полей, чтобы он имел возможность выбора ключевого атрибута, пусть и не минимального. Это может оказаться необходимо в связи с особенностями предметной области.

П2. Поиск атрибутов, конкатенация значений, которых минимальна.

Необходимо проанализировать все возможные сочетания атрибутов. Каждое сочетание проверить на уникальность конкатенации их значений.

Все сочетания с уникальными значениями необходимо сохранить, а затем для каждого сочетания измерить их суммарную длину. Для обеспечения возможности принятия волевого решения необходимо расположить эти сочетания в порядке возрастания. Более того, найденный ключ может не удовлетворять второму требованию минимальности, и придется выбирать альтернативный ключ.

Имеет смысл сначала выполнить проверку соответствия требований к первичному ключу 2-х атрибутов, т.е. необходимо проанализировать C_N^2 сочетаний, где N – степень отношения.

$MKA^2 = \emptyset$, где MKA^2 – множество пар атрибутов, которые претендуют на роль первичного ключа.

Пусть имеется отношение R:

$R=(A_1, \dots, A_i, \dots, A_m, \dots, A_k), i = \overline{1, k}$, где k – степень отношения; A_i – атрибут отношения.

$MPA = \emptyset$

Ищутся все возможные сочетания пар атрибутов и запоминаются в массив MPA :

$Cnt=0$

DO $i = 1$ to $k-1$

DO $j = i+1$ to k

$Cnt = Cnt + 1$

$S = Concat(A_i, A_j)$

$MPA(Cnt) = MPA + S$

END i

END j

Таким образом, в массиве MPA сформируются все возможные пары атрибутов, а в счетчике Cnt хранится их количество.

Проверяются все пары на уникальность.

$MUP = \emptyset$ /* Массив пар атрибутов, представляющих собой атрибуты, все соответствующие пары значений которых уникальны */

$Cnt1 = 0$

DO $i = 1$ to Cnt

$S = MPA(Cnt)$

/* По сути S представляет собой пару атрибутов (A_i, A_j)

$A_i = (e_{i_1} \dots e_{i_m})$, где e_{i_1} – 1-й элемент домена с атрибутом A_i , e_{i_m} – m -й элемент домена с атрибутом A_i .

$A_j = (e_{j_1} \dots e_{j_m})$, где e_{j_1} – 1-й элемент домена с атрибутом A_j , e_{j_m} – m -й элемент домена с атрибутом A_j , m – мощность отношения. */

DO $n = 1$ to m

Для каждой пары атрибутов (A_i, A_j) выполняется проверка условия $\text{Concat}(e_i, e_j) \neq \dots \neq \text{Concat}(e_j, e_i)$

END n

Если текущая пара атрибутов имеет все соответствующие пары значений уникальными, то эта пара добавляется к массиву пар с уникальными значениями:

$$\text{Cnt1} = \text{Cnt1} + 1$$

$$\text{MUP}(\text{Cnt1}) = S$$

END i

Если претенденты на ключевой атрибут найдены, т.е. $\text{MUP} \neq \emptyset$, то для проверки второго требования минимальности выполняется переход к ПЗ. Для обеспечения возможности принятия волевого решения необходимо расположить все найденные сочетания в порядке возрастания. Более того, найденный ключ может не удовлетворять второму требованию минимальности и придется выбирать альтернативный ключ.

Если не найдено таких двух атрибутов, которые удовлетворяют требованиям к первичному ключу, то разработчик может назначить суррогатный ключ или попытаться найти такие 3-и атрибута, которые удовлетворяют требованиям к первичному ключу, т.е. проанализировать C_N^3 сочетаний.

ПЗ. Поиск первичного ключа на основе 3-х атрибутов

$\text{MKA}^3 = \emptyset$, где MKA^3 – множество троек атрибутов, которые претендуют на роль первичного ключа.

Пусть имеется отношение R:

$R = (A_1, \dots, A_i, \dots, A_m, \dots, A_k)$, $i = \overline{1, k}$, где k – степень отношения; A_i – атрибут отношения.

$$\text{MPA} = \emptyset$$

Ищутся все возможные сочетания троек атрибутов и запоминаются в массив MPA :

```

Cnt=0
DO i = 1 to k-2
  DO j = i+1 to k
    DO r = j+1 to k
      Cnt = Cnt + 1
      S = Concat (Ai, Aj, Ar)
      MPA (Cnt) = MPA + S
    END r
  END j
END i
END j

```

Таким образом, в массиве MPA сформируются все возможные тройки атрибутов, а счетчике Cnt хранится их количество.

Проверяются все тройки на уникальность.

MUP = ∅ / Массив троек атрибутов, представляющих собой атрибуты, у которых соответствующие тройки значений которых уникальны */*

```

Cnt1 = 0
DO i = 1 to Cnt
  S = MPA(Cnt)
  /* По сути S представляет собой тройку атрибутов (Ai, Aj, Ar)
  Ai = (ei1...eim), где ei1 – 1-й элемент домена с атрибутом Ai, eim – m-й
  элемент домена с атрибутом Ai.
  Aj = (ej1...ejm), где ej1 – 1-й элемент домена с атрибутом Aj, erm – m-й
  элемент домена с атрибутом Aj, m – мощность отношения.
  Ar = (er1...erm), где er1 – 1-й элемент домена с атрибутом Ar, ejm – m-й
  элемент домена с атрибутом Ar, m – мощность отношения. */
  DO n = 1 to m

```

Для каждой тройки атрибутов (A_i, A_j, A_r) выполняется проверка условия $Concat(e_{i_1}, e_{r_1}, e_{j_1}) \neq \dots \neq Concat(e_{j_1}, e_{r_m}, e_{j_m})$

END n

Если текущая тройка атрибутов имеет все соответствующие тройки значений уникальными, то эта тройка добавляется к массиву троек с уникальными значениями:

$$Cnt1 = Cnt1 + 1$$

$$MUP(Cnt1) = S$$

END i

Если претенденты на ключевой атрибут найдены, т.е., $MUP \neq \emptyset$, то для проверки второго требования минимальности выполняется переход к ПЗ.

Для обеспечения возможности принятия решения разработчиком следует расположить все найденные сочетания в порядке возрастания. Не исключено, что найденный ключ может не удовлетворять второму требованию минимальности и придется использовать альтернативный ключ.

Аналогичный подход распространяется и на большее число атрибутов, но обычно в ключевом поле очень редко задействуют более 4-х атрибутов.

Складывается впечатление, что этот процесс займет много машинного времени, которое зависит от степени.

Из комбинаторики известно, что число возможных сочетаний $C_n^k = n!/(n-k)!/k!$, где n – число элементов множества, а k – количество проверяемых значений число возможных сочетаний $C_n^k = n!/(n-k)!/k!$, где n – число элементов множества, а k – количество проверяемых значений [63, 64]. Например, для множества из 4-х элементов (a, b, c, d) возможны следующие сочетания пар элементов (a, b), (a, c), (a, d), (b, c), (b, d), (c, d).

Подсчитаны числа возможных сочетаний для различных n и k , которые представляют наибольший интерес. Результаты сведены в таблицу 3.1.

Таблица 3.1

C_{10}^2	C_{10}^3	C_{10}^4	C_{20}^2	C_{20}^3	C_{20}^4	C_{30}^2	C_{30}^3	C_{30}^4
45	120	210	190	1140	4845	435	4060	27405

Степени отношений (10, 20, 30) наиболее близки к распространенным степеням в реальных БД, а число атрибутов (2, 3, 4) обычно достаточно для первичного ключа. Подсчитанное число сочетаний вполне может быть обработано на современных компьютерах без существенной потери машинного времени.

ПЗ. Поиск первичных ключей, которые входят в элементы массива, и которые имеют уникальные значения.

После выполнения П2 сформируется массив пар или массив троек атрибутов, представляющих собой атрибуты, у которых соответствующие тройки значений уникальны. Следует проверить все элементы массивов на предмет наличия атрибутов, которые входят в элементы массива и которые имеют уникальные значения. Если такие элементы (атрибуты) найдутся, то они не могут в соответствии со вторым правилом минимальности первичного ключа претендовать на составную часть первичного ключа.

Проверка массива пар атрибутов.

В этом случае $MUP = (P_1, \dots, P_i, \dots, P_n)$, где P_i – i -я пара атрибутов, n – число пар атрибутов с уникальными конкатенациями значений.

$$P_i = (A_{i1}, A_{i2})$$

$A_{i1} = (e_{i1_1}, \dots, e_{i1_m})$, где e_{i1_1} – 1-й элемент домена с атрибутом A_{i1} , m – мощность отношения.

$A_{i2} = (e_{i2_1}, \dots, e_{i2_m})$, где e_{i2_1} – 1-й элемент домена с атрибутом A_{i2} , m – мощность отношения.

При этом должно выполняться условие:

$$\neg ((e_{i1_1} \neq \dots \neq e_{i1_m}) \vee (e_{i2_1} \neq \dots \neq e_{i2_m}))$$

Подобные условия должны выполняться для всех элементов массива $MUP = (P_1, \dots, P_i, \dots, P_n)$, то есть для всех пар атрибутов, претендующих на ключевые.

Проверка массива троек атрибутов.

В этом случае $MUP = (P_1, \dots, P_i, \dots, P_n)$, где P_i – i -я тройка атрибутов, n – число пар атрибутов с уникальными конкатенациями значений.

$$P_i = (A_{i_1}, A_{i_2}, A_{i_3})$$

$A_{i_1} = (e_{i_1}, \dots, e_{i_1_m})$, где e_{i_1} – 1-й элемент домена с атрибутом A_{i_1} , m – мощность отношения.

$A_{i_2} = (e_{i_2}, \dots, e_{i_2_m})$, где e_{i_2} – 1-й элемент домена с атрибутом A_{i_2} , m – мощность отношения.

$A_{i_3} = (e_{i_3}, \dots, e_{i_3_m})$, где e_{i_3} – 1-й элемент домена с атрибутом A_{i_3} , m – мощность отношения.

При этом должно выполняться условие:

$$\neg ((e_{i_1} \neq \dots \neq e_{i_1_m}) \vee (e_{i_2} \neq \dots \neq e_{i_2_m}) \vee (e_{i_3} \neq \dots \neq e_{i_3_m}))$$

Подобные условия должны выполняться для всех элементов массива $MUP = (P_1, \dots, P_i, \dots, P_n)$, то есть для всех троек атрибутов, претендующих на ключевые.

Для четверок атрибутов, претендующих на роль первичного ключа, выполняются аналогичные пункты алгоритма. STOP.

В конечном итоге будет получен список пар или троек атрибутов, которые претендуют на первичный ключ. Из этого списка разработчик должен выбрать единственный кортеж, исходя из своих прикладных соображений. Список может состоять и из одного элемента. Если список окажется пустым, то назначается суррогатный ключ. Однако такая ситуация скорее всего будет отсечена на шаге поиска первичного ключа, включающего в себя один атрибут. Алгоритм детально излагается в авторской работе [93].

3.2.2. Формальные алгоритмы назначения первичных ключей в ИТВР

III. Поиск единственного атрибута, все значения которого уникальны.

$MKA = \emptyset$ __ Массив ключевых атрибутов

$Snt = 0$ __ Счетчик ключевых атрибутов

```

DO i = to k __ Перебор всех атрибутов отношения
  F = 1 __ Флажок - все элементы текущего атрибута уникальны
  DO j = 1 to m-1 __ Перебор всех элементов атрибута
    DO r = j + 1 to m __ Перебор всех элементов атрибута
      If  $e_{i_j} = e_{i_r}$  Then __ Сравнение каждой пары элементов
        Begin __ Если они равны
          F = 0 __ Флажок сбрасывается
          Exit r __ Завершается цикл по r
          Exit j __ Завершается цикл по j
        End
      END r
    END j
  If F = 1 Then __ Если флажок не сброшен
    Begin
      Cnt = Cnt + 1 __ В массив добавляется очередной
      MKA(Cnt) =  $A_i$  __ претендент на ключевой атрибут
    End
  END i
IF MKA =  $\emptyset$  Then
  Begin
    Print ("Претендентов на ключевой атрибут нет")
    Print ("Сформируйте суррогатный ключ или")
    Print ("перейдите к поиску ключа из нескольких атрибутов")
  Else
    Begin
      Print ("Претенденты на ключевые атрибуты: ", MKA)
    End
  /* Здесь k- число атрибутов отношения; m – мощность отношения. */
  /* Поиск атрибута с минимальной длиной */

```

```

/* Упорядочивание по возрастанию длины элементов массива МКА */
/* Используется известный алгоритм сортировки */
B = 0
DO i = 1 to Cnt-1
    DO j = i + 1 Cnt
        If Length( MKAj) < If Length( MKAi) Then
            Begin
                P = MKAj
                MKAi = MKAj
                MKAj = P
            End
        END j
    END i
Print ("Претенденты на ключевые атрибуты в порядке возрастания
длины : ")
Print (MKA)
STOP

П2. Поиск атрибутов, конкатенация значений, которых минимальна.
/* Ищутся все возможные сочетания пар атрибутов */
MPA = ∅ __ массив пар атрибутов
MUP = ∅ /* Массив пар атрибутов, представляющих собой
атрибуты, все соответствующие пары значений которых уникальны */
Cnt = 0 __ счетчик пар атрибутов
DO i = 1 to k-1 __ цикл по числу атрибутов
    DO j = i+1 to k
        Cnt = Cnt + 1
        S = Concat (Ai, Aj)
        MPA (Cnt) = MPA + S
    END j
END i

```


END j

Cnt1 = 0 __ *счетчик пар атрибутов с уникальными значениями*

DO i = to Cnt __ *Перебор всех возможных пар атрибутов*

F = 1 __ *Флажок – все элементы текущей пары*

__ *атрибутов уникальные*

DO j = 1 to m-1 __ *Перебор всех пар элементов атрибутов*

DO r = j + 1 to m __ *Перебор всех пар элементов атрибутов*

If e_j(MPA_i) = e_r(MPA_i) Then

/ Здесь e_j(MPA_i) j-я пара элементов, соответствующая i-му элементу массива пар атрибутов MPA */*

Begin __ *Если они равны*

F = 0 __ *Флажок сбрасывается*

Exit r __ *Завершается цикл по r*

Exit j __ *Завершается цикл по j*

End

END r

END j

If F = 1 Then __ *Если флажок не сброшен*

Begin

Cnt1 = Cnt1 + 1 __ *В массив добавляется очередной*

MUP(Cnt1) = MPA_i __ *претендент на пару атрибутов,*

__ *входящий в ключевой атрибут*

End

END i

IF MUP = ∅ Then

Begin

Print (“Претендентов на пару ключевых атрибутов нет”)

Print (“Сформируйте суррогатный ключ или”)

```

    Print (“перейдите к поиску ключа из трех атрибутов”)
Else
Begin
    Print (“Претенденты на ключевые атрибуты: ”, MUP)
End
/* m – мощность отношения. */

/* Поиск пары атрибута с минимальной длиной */
/* Упорядочивание по возрастанию длины элементов массива MUP */
B = 0
DO i = 1 to Cnt1-1
    DO j = i + 1 Cnt1
        If Length( MUPj) < If Length(MUPi) Then
            Begin
                P = MUPi
                MUPi = MUPj
                MUPj = P
            End
        END j
    END i
    Print (“Претенденты на пары ключевых атрибутов в порядке
возрастания длины: ”)
    Print (MUP)
STOP

П3. Поиск первичного ключа на основе 3-х атрибутов
/* Ищутся все возможные сочетания троек атрибутов */
MUT = ∅ /* Массив троек атрибутов, представляющих собой
атрибуты, все соответствующие тройки значений которых уникальны */
Cnt=0__ счетчик троек атрибутов
DO i = 1 to k-2__ цикл по числу атрибутов - 2

```

```

DO j = i+1 to k
  DO r = j+1 to k
    Cnt = Cnt + 1
    S = Concat (Ai, Aj, Ar)
    MUT (Cnt) = MUT + S
  END r
END i

```

```
END j
```

```
/* k – общее число атрибутов */
```

Cnt1 = 0 __ счетчик троек атрибутов с уникальными значениями

DO i = to Cnt __ Перебор всех возможных троек атрибутов

F = 1 __ Флажок - все элементы текущей тройки

__ атрибутов уникальные

DO j = 1 to m-1 __ Перебор всех троек элементов атрибутов

DO r = j + 1 to m __ Перебор всех троек элементов атрибутов

If e_j^{'''}(MUT_i) = e_r^{'''}(MUT_i) Then

/ Здесь e_j^{'''}(MUT_i) j-я тройка элементов, соответствующая*

*i-му элементу массива троек атрибутов MUT */*

Begin __ Если они равны

F = 0 __ Флажок сбрасывается

Exit r __ Завершается цикл по r

Exit j __ Завершается цикл по j

End

END r

END j

If F = 1 Then __ Если флажок не сброшен

Begin

Cnt1 = Cnt1 + 1 __ В массив добавляется очередной

$MUT(Cnt1) = MPA_i$ __ претендент на тройку атрибутов,
 __ входящий в ключевой атрибут

End

END i

IF $MUT = \emptyset$ *Then*

Begin

Print (“Претендентов на пару ключевых атрибутов нет”)

Print (“Сформируйте суррогатный ключ или”)

Print (“перейдите к поиску ключа из трех атрибутов”)

Else

Begin

Print (“Претенденты на ключевые атрибуты: ”, МКА)

End

/ m – мощность отношения. */*

/ Поиск тройки атрибутов с минимальной длиной */*

/ Упорядочивание по возрастанию длины элементов массива MUT */*

$B = 0$

DO $i = 1$ *to* $Cnt1 - 1$

DO $j = i + 1$ $Cnt1$

If $Length(MUT_j) < Length(MUT_i)$ *Then*

Begin

$P = MUT_i$

$MUT_i = MUT_j$

$MUT_j = P$

End

END j

END i

Print (“Претенденты на тройки ключевых атрибутов в порядке возрастания длины : ”)

Print (MUT)

STOP

Как нетрудно заметить, алгоритм поиска первичного ключа на основе 2-х атрибутов и алгоритм поиска первичного ключа на основе 3-х атрибутов имеют схожую структуру. В связи с этим алгоритмы, разработанные для большего числа атрибутов, в работе не приводятся.

3.3. Алгоритмы назначения внешних ключей в ИТВР

3.3.1. Неформальные алгоритмы назначения внешних ключей в ИТВР

В соответствии с [11] определение внешнего ключа звучит следующим образом. Пусть имеется отношение R2, значения атрибутов которого равны значениям атрибутов первичного ключа отношения R2. Тогда такие атрибуты являются внешним ключом по отношению к R1.

В соответствии с определением для нахождения внешних ключей необходимо проанализировать каждую таблицу, все возможные сочетания пар таблиц БД. Для каждой пары таблиц выполнить поиск неключевых атрибутов одной таблицы, значения которых были бы равны значениям ключевых атрибутов другой таблицы. Если таковые найдутся, то они и будут претендовать на внешний ключ.

Следует обратить внимание на то, что эти ключи ориентированы на формирование связи между таблицами типа “многие к одному”. Они и представляют наибольший интерес для БД. Это связано с тем, что связь типа “один к одному” по сути представляет собой связь между частями одной таблицы, а связь типа “многие ко многим” базируется на двух связях типа “многие к одному” между рассматриваемыми и какой-либо третьей таблицей.

Неформальный алгоритм.

П1. Осуществляется поиск всех возможных сочетаний пар таблиц анализируемой БД.

Пусть имеется табличное пространство T :

$T = (T_1, \dots, T_i, \dots, T_m, \dots, T_k)$, $i = \overline{1, k}$, где k – число прикладных таблиц БД.

$MPT = \emptyset$

Ищутся все возможные сочетания пар таблиц и запоминаются в массив

MPT

$Cnt = 0$

Do $i = 1$ to $k-1$

Do $j = i+1$ to k

$Cnt = Cnt + 1$

$S = Concat(T_i, T_j)$

$MPT(Cnt) = S$

END i

END j

Таким образом, в массиве MPT сформируются все возможные пары атрибутов, а счетчике Cnt хранится их количество.

П2. Для каждой пары таблиц из массива MPT выполняется поиск внешнего ключа.

Организуется цикл на Cnt итераций, в котором перебираются все пары таблиц.

Для каждой пары таблиц выполняются проверки. Причем рассматриваются два случая: когда в качестве таблицы с первичным ключом рассматривается 1-я таблица; когда в качестве таблицы с первичным ключом рассматривается 2-я таблица.

П2.1. Для ключа состоящего из одного атрибута.

$A_i = (e_{i_1} \dots e_{i_m})$, где A_i – атрибут, соответствующий первичному ключу 1-й таблицы, e_{i_1} – 1-й элемент домена с атрибутом A_i , e_{i_m} – m-й элемент домена с атрибутом A_i , m – мощность 1-й таблицы.

$A_j^2 = (e_{j_1}^2 \dots e_{j_f}^2)$, где A_j^2 – атрибут, соответствующий неключевому атрибуту 2-й таблицы, $e_{j_1}^2$ – 1-й элемент домена с атрибутом A_j^2 , $e_{j_f}^2$ – f-й элемент домена с атрибутом A_j^2 , f – мощность 2-й таблицы. Важно помнить, что число столбцов у 2-й таблицы, как правило, больше одного.

Следует отметить, что m и f в общем случае неравны. Ведь таблицы, как правило, имеют различные мощности.

Организуется цикл на g итераций, где g число столбцов 2-й таблицы

Сравниваются значения первичного ключа 1-й таблицы и значения текущего столбца 2-й таблицы:

Cnt1 = 0

Do t = 1 to m

Do p = 1 to f

If $e_{i_t} = e_{j_p}^2$ THEN Cnt1 = Cnt1 + 1

End p

End t

If Cnt1 \geq Min(m, f) Then PRINT (“столбец ”, g, “2-й таблицы претендует на внешний ключ”

Завершается цикл по g

Если n для одного столбца таблицы 2 не сформируется сообщение, то во 2-й таблице внешних ключей нет.

Здесь m – мощность 1-й таблицы,

f – мощность 2-й таблицы.

В качестве пояснения можно сказать, что истинность конструкции $Cnt1 \geq \text{Min}(m, f)$ в подавляющем случае подтверждает, что найден внешний ключ.

Действительно, для каждого значения одной таблицы нашлось равное ему значение другой таблицы.

П2.2. Для ключа, состоящего из двух атрибутов.

Принципиальное отличие от предыдущего алгоритма состоит в том, что во второй таблице рассматриваются все возможные сочетания пар атрибутов 2-й таблицы.

В связи с этим необходимо предварительно сформировать массив всех возможных пар атрибутов 2-й таблицы.

```

MPA = ∅
Cnt1 = 0
Do i = 1 to f-1
    Do j = 1 to f
        S = Concat(Ai, Aj)
        Cnt1 = Cnt1 + 1
        MPA(Cnt1) = s
    END j
END i

```

Здесь MPA – массив пар атрибутов 2-й таблицы,

f – степень 2-й таблицы,

A_i, A_j – два атрибута 2-й таблицы.

Организуется цикл на Cnt1 итераций, где Cnt1 – число сочетаний пар атрибутов 2-й таблицы, которые содержатся в массиве MPA, z – параметр цикла

Сравниваются все пары значений, соответствующих первичному ключу 1-й таблицы и все пары значений для каждого сочетания атрибутов 2-й таблицы:

```

Cnt1 = 0
Do t = 1 to m
    Do p = 1 to f

```


If Concat(e_{i_1}, e_{k_1}) = e_p (MPA_z) THEN Cnt1 = Cnt1 + 1

End p

End t

If Cnt ≥ Max(m, f) Then PRINT (“столбцы”, MPA_z, “2-й таблицы претендует на внешний ключ”

Завершается цикл по Cnt1

Если ни для одного сочетания таблицы 2 не сформируется сообщение, то во 2-й таблице внешних ключей нет.

Здесь

$A_i, A_k = ((e_{i_1}, e_{k_1}) \dots (e_{i_m}, e_{k_m}))$, где A_i – 1-й атрибут, соответствующий первичному ключу 1-й таблицы, A_k – k-й атрибут, соответствующий первичному ключу 1-й таблицы, e_{i_1} – 1-й элемент домена с атрибутом A_i , e_{k_1} – 1-й элемент домена с атрибутом A_k .

m – мощность 1-й таблицы,

f – мощность 2-й таблицы,

e_{i_1}, e_{k_1} – текущие t-е 2-а элемента первичного ключа 1-й таблицы,

e_p (MPA_z) – текущие p-е 2-а элемента 2-й таблицы, при этом номера сочетаний определяются z-м элементом массива MPA (массива, в котором содержатся все пары атрибутов 2-й таблицы).

П2.3. и П2.4. Для ключа состоящего из трех атрибутов и для ключа состоящего из четырех атрибутов выполняются в основном на базе конструкций **П2.1. и П2.2.** Но из соображений их значительного объема не приводятся.

Завершается цикл на Cnt итераций, в котором перебираются все пары таблиц

3.3.2 Формальные алгоритмы назначения внешних ключей в ИТВР

III. Осуществляется поиск всех возможных сочетаний пар таблиц анализируемой БД.

/ $T=(T_1, \dots, T_i, \dots, T_m, \dots, T_k)$, $i = \overline{1, k}$, где k – число прикладных таблиц БД */*

MPT = ∅ __ Массив всех пар таблиц БД

/ Ищутся все возможные сочетания пар таблиц */*

Cnt=0

Do i = 1 to k-1 __ k – число таблиц БД

Do j = i+1 to k

Cnt = Cnt + 1 __ счетчик числа пар таблиц

S = Concat (T_i, T_j)

MPT (Cnt) = S

END i

END j

II. Для каждой пары таблиц из массива *MPT* выполняется поиск внешнего ключа.

/ Перебираются все возможные пары таблиц, сформированные ранее, перебираются все столбцы 1-й таблицы текущей пары, перебираются все столбцы 2-й таблицы текущей пары, каждый элемент первого текущего столбца сравнивается с каждым элементом второго текущего столбца, если элементы равны, то они подсчитываются. Если совпавших элементов оказалось больше, чем наибольшая мощность пары таблиц, то столбцы претендуют на ключевые */*

F = 0 __ флажок наличия связанных ключей

Cnt1 = 0 __ счетчик совпадений

Do i = 1 to Cnt __ – число пар таблиц БД

```

Do l = 1 to C1__ – степень 1-й таблицы из i-й пары
  Do m = 1 to C2__ – степень 2-й таблицы из i-й пары
    Do j = 1 to M1__ – мощность 1-й таблицы из i-й пары
      Do k = 1 to M2__ – мощность 2-й таблицы из i-й пары
        If  $e_{i_j} = e_{m_k}$  Then Coun1 = Coun1 + 1
      END k
    END j
  END m
  If Coun1 >= Max(M1, M2) Then
    Begin
      Print (“Для пары таблиц ”, i, “столбцы”, l, “ и ”, m)
      Print (“ претендуют на первичный и внешние ключи”)
      Cnt1 = 0
      F = 1
    End
  END m
END l
END i
If F = 0 Then Print (“ внешних ключей не обнаружено не для одной из
таблиц”)

```

Следует обратить внимание на то, что атрибуты, предложенные алгоритмом в качестве связанных атрибутов, могут не удовлетворять разработчика, поэтому необходимо предоставить ему возможность окончательного решения.

Суть метода назначения ключевых полей в ИТВР состоит в преобразовании модели ИТВР в модель РТР в соответствии с требованиями к ключевым полям, со схемой взаимодействия разработчика и подсистемы и предложенным комплексом алгоритмов.

Другими словами, метод состоит в анализе значений таблиц ИТВР, выявлении несоответствий требованиям к ключевым полям и исключению этих несоответствий на основе применения предложенной формализации.

Выводы по главе 3

1. Сформулирована задача назначения ключевых полей в заполненных ИТВР. Эта задача для ИТВР в полном объеме ранее сформулирована не была.

2. Разработан метод назначения первичных и внешних ключевых полей в ИТВР. Метод позволяет решать задачи назначения ключевых полей не в ручном, а в автоматизированном режиме и сократить время назначения в десятки раз.

3. Предложенный метод является ключевой компонентой разрабатываемой методики.

4. МЕТОДИКА ФОРМИРОВАНИЯ РТР НА ОСНОВЕ ИСПОЛЬЗОВАНИЯ ЗАПОЛНЕННЫХ ИТВР

В четвертой главе решена задача формализации методики формирования реляционных таблиц на основе информации табличного вида, которая отражает реальное положение дел в области представления данных табличного вида. В основу методики положены разработанные модели ИТВР и РТР, метод преобразования ИТВР в РТР, метод формирования ключевых полей на основе анализа ИТВР, которые разработаны в предыдущих главах.

Разработана модель методики в операторной форме. Разработана модель методики с использованием аппарата сетей Петри. Проведено исследование методики с использованием аппарата сетей Петри. В результате исследований и модификаций модели исключены концептуальные ошибки в описании методики и ее функционировании.

4.1. Формулировка проблемы формализации методики

Целью работы является разработка методики формирования реляционных таблиц на основе существующей табличной информации. При этом необходимо описать совокупность приемов, способов для достижения поставленной цели [92]. Понятие способ в энциклопедиях и словарях определяется зачастую через понятие метод и наоборот. В любом случае в развернутых определениях этих понятий обычно используют такие термины как модели, преобразования, совокупность приемов, задачи, система действий, достижение цели. Все это отражено в предыдущих главах.

Комплект задач и состав методов их решения определен в 1-й главе. Метод решения задач формирования реляционных таблиц на основе информации табличного вида разработан во 2-й главе. Метод назначения ключевых полей разработан в 3-й главе. Эти методы тесно связаны между собой и ориентированы на достижение одной цели. Предложенные методы

включает в себя совокупность разработанных алгоритмов, которые взаимосвязаны и взаимозависимы и выполняются в логической последовательности. Взаимосвязанные методы и алгоритмы, выполняемые в логической последовательности, составляют методику.

Методика в общем и конкретном случаях – это не только совокупность методов выполнения действий. Большинству из методов соответствуют частные приемы использования их компонент.

Так, например, метод преобразования нереляционных таблиц ИТВР в реляционные таблицы включает в себя приемы исключения заголовков внутри таблиц, исключения сложных подзаголовков, исключения пустых подзаголовков, избавления от заголовков в первом столбце, приведения значений атрибутов к одному типу, исключения дублирования записей, исключения пустых записей.

Метод назначения ключевых полей включает в себя приемы выявления столбца с уникальными значениями его элементов, выявления сочетания столбцов с уникальными сочетаниями соответствующих им элементов, поиска минимальных первичных ключей, включающих в себя один атрибут, поиска минимальных первичных ключей, включающих в себя несколько атрибутов, выявления атрибутов, которые входят в первичный ключ и содержат уникальные значения, выявления внешних ключей.

В качестве основных мероприятий, которые необходимо выполнить в процессе разработки и анализа модели процесса преобразования информации табличного вида в реляционные таблицы обозначены следующие мероприятия:

- определить составляющие методики (модели, алгоритмы, методы);
- определить последовательность выполнения шагов использования моделей и человеко-машинных процедур
- выбрать средства формализации для описания и исследования методики;

- реализовать построение модели методики на основе использования выбранных средств формализации;
- выявить и исключить принципиальные ошибки в методике;
- провести анализ динамических свойств методики;
- полученную модель методики использовать в качестве исходной формализации для реализации человеко-машинных процедур преобразования ИТВР в РТР, а также в качестве руководства разработчика баз данных [37].

В разрабатываемой методике предполагается активное участие разработчика. Это связано с тем, что процедуры методики в процессе их выполнения должны предлагать промежуточные проектные решения, а пользователь должен принимать окончательные решения. Человеко-машинным процедурам посвящены, в частности, работы [6, 74, 75]. При разработке методики автор руководствовался основополагающими идеями, сформулированными в этих работах.

Реализована следующая процедура по построению модели методики проектирования РТР на основе существующих заполненных ИТВР.

На первоначальной фазе разработки методики в соответствии с описанием процессов совместной работы решающих систем [75] и на основе отличий ИТВР и РТР отображаются шаги формирования РТР на основе заполненных ИТВР, отображаются их взаимосвязи. Это описание формируется для обозначения основных человеко-машинных процедур и их взаимосвязи. Описание строится в виде операторной модели. Такой выбор обусловлен тем, что операторное описание включает в себя операторы, состояния и связи между ними. А это очень близко к описанию состава и взаимосвязи программных процедур, недаром команды многих языков программирования называются операторами. Таким образом, операторная модель является одной из лучших формализованных технических заданий на разработку и реализацию автоматизированной подсистемы. Важно отметить, что в отличие от обычных программных подсистем, автоматизированные

подсистемы кроме начального и конечного состояния имеют промежуточные состояния. В них пользователь оценивает ход работы подсистемы и принимает решение о дальнейшем направлении проектных работ. В связи с этим, тем более оправдано использование операторной модели, которая интуитивно понятна и может быть использована в качестве формализованного руководства пользователя.

На следующей фазе разработки методики на основе использования аппарата сетей Петри строится ее сетевая модель. Выбор этого аппарата обусловлен тем, что операторная модель при всех ее достоинствах не позволяет выполнять исследований модели процесса, не обеспечивает выявление и исключение в ней принципиальных ошибок. Кроме того, выбор обусловлен авторитетным мнением крупного специалиста в области автоматизированных систем И.П. Норенкова [74, 75]. Он считает, что для исследования автоматизированных процессов наиболее приемлемы системы массового обслуживания и сети Петри.

Заключительная фаза разработки методики сводится к выявлению концептуальных ошибок в сетевой модели. Это позволит избавиться от внесения этих ошибок в процедуры подсистемы преобразования в ходе их реализации.

4.2. Модель методики в операторной форме

В описанных мероприятиях за основу взят подход, который предложен в работах руководителя диссертации Брешенкова А.В. Он изложен в работах [6, 7, 32, 37, 53, 71, 76].

Такой выбор обусловлен тем, что диссертация является развитием, продолжением и дополнением работ моего руководителя. И просто нелогично и неоправданно выбирать математический аппарат для описания и исследования методики, который отличен от аппарата использованного в базовой работе. Ведь тогда исследования, разработки и реализации не будут органично вписываться в существующие исследования, разработки и

реализации, что существенно снизит их теоретическую и практическую ценность. Подход - это скорее стратегия, чем решение проблемы. Все решения в работе принимаются самостоятельно, исходя из соображений обеспечения эффективности использования способов, разработанных в первых главах диссертации. В технической интерпретации подход наиболее близок к техническому заданию, а не к его реализации.

Следует отметить, что в подходах есть и различия. В частности в [37, 53] разработано формализованное описание методологии, выявлены ее компоненты и связи между ними, а затем разработаны эти компоненты. В рассматриваемом случае сначала разработаны новые компоненты методики (их состав уже известен), а затем разработано формализованное описание процесса их использования.

Укрупненная модель процесса формирования РТР на основе использования заполненных ИТВР представлена на рисунке 4.1. Использование в модели изображения человека объясняется тем, что разрабатывается модель человеко-машинного процесса.

Важно отметить, что, несмотря на некоторое внешнее сходство представленной модели с моделью предложенной в [53], есть принципиальное отличие - целевое состояние не БД, а РТР. Кроме того, модели ИТВ и РТ во 2-й главе диссертации существенно расширены и мероприятия по преобразованию ИТВР в РТР оригинальны и носят более развернутый характер (главы 2, 3).



Рис. 4.1. Укрупненная модель процесса формирования РТР на основе использования заполненных ИТВР

Активное и определяющее участие разработчика в процессе преобразования ИТВР в РТР позволяет в полной мере задействовать его творческий потенциал. В этой связи стоит задача разработки следующих средств:

- средств, которые позволят активно участвовать разработчику в процессе преобразования ИТВР в РТР;
- средств, которые позволят использовать существующие системы проектирования БД [37].

Введем оператор преобразования ИТВР в РТР. Результатом выполнения этого оператора является объект, соответствующий предложенной ранее модели РТР. Операндами оператора *ОПП*, как видно из рисунка, являются модель ИТВР и модель РТР.

Следует отметить, что обозначения операторов и состояний здесь и в дальнейшем приняты отличные от близких по смыслу обозначений операторов и состояний, использованных в работах [37, 53]. Это сделано намеренно. С одной стороны для того, чтобы избежать путаницы в моделях. А с другой стороны, для того чтобы сделать акцент на то, что операторы и состояния отражают различные по сути объекты операторной модели.

Укрупненная операторная модель формирования РТР на основе использования заполненных ИТВР приведена на рисунке 4.2.

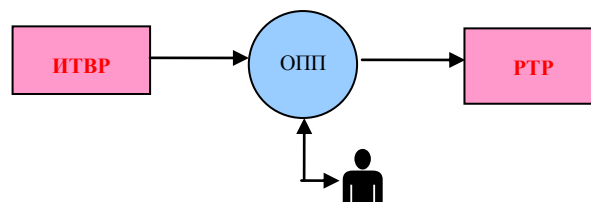


Рис. 4.2. Укрупненная операторная модель формирования РТР на основе использования заполненных ИТВР

Таким образом: $РТР = ОПП(ИТВР)$.

Разработчик РТР должен принимать решение по результатам выполнения *ОПП*. В связи с этим необходима система оценок результатов

выполнения *ОПП*. Назовем ее *СООП*. Для реализации указаний разработчика необходимы соответствующие средства. Оператор, соответствующий этим средствам, назовем *ОПРУ*.

В свете высказанных соображений укрупненная операторная модель формирования РТР на основе использования заполненных ИТВР примет вид (рис. 4.3).

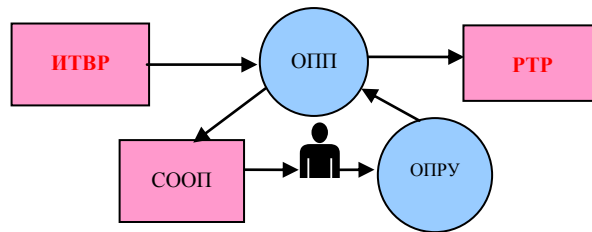


Рис. 4.3. Вторая укрупненная операторная модель формирования РТР на основе использования заполненных ИТВР

В целях использования типовых компонент модели исключим изображение человека, определив тем самым, что разработчик задействован в *СООП* и *ОПРУ*. В результате получим модифицированную вторую укрупненную операторную модель формирования РТР на основе использования заполненных ИТВР (рис. 4.4).

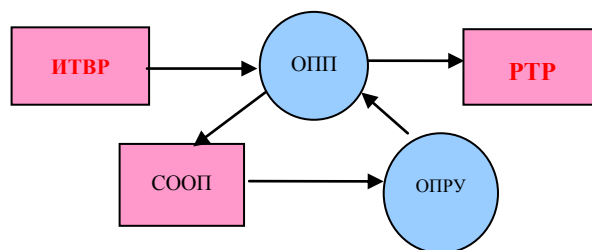


Рис. 4.4. Модифицированная вторая укрупненная операторная модель формирования РТР на основе использования заполненных ИТВР

Уже сейчас мы можем назвать основные задачи последующих исследований [37]:

- необходимо оценить полноту модели ИТВР в плане ее использования для разработки человеко-машинных процедур соответствующих операторам *ОПП*, *ОПРУ* и системе оценок *СООП*;

- необходимо оценить полноту модели РТР в плане ее использования для разработки человеко-машинных процедур соответствующих операторам *ОПП*, *ОПРУ* и системе оценок *СООП*;

- необходимо разработать человеко-машинные процедуры для реализации выполнения операторов *ОПП*, *ОПРУ*;

- необходимо разработать процедуры, соответствующие *СООП*.

Необходимо выполнить реализацию *СООП*, *ОПП* и *ОПРУ*, исходя из схемы иерархии разрабатываемых алгоритмов, предложенной в главе 1 (рис. 1.8).

Модели РТР и ИТВР разработаны во 2-й главе. И в той и другой модели описаны множества таблиц. Это обеспечивает возможность реализации *ОПП* и *ОПРУ*.

Основные отличия моделей ИТВР и РТР:

1. ИТВР, как правило представлены таблицами, которые не удовлетворяют требованиям к РТР.

2. В ИТВР первичные и внешние ключи не определены.

3. Между таблицами ИТВР нет связей.

Для выполнения *ОПП* и *ОПРУ* необходима система оценок. Эта система должна обеспечить оценку результатов указаний разработчика (*ОПРУ*) и обеспечить переход к оператору преобразования *ОПП*. Назовем соответствующее состояние модели ПППР (подсистема преобразований).

Четвертая операторная форма модели автоматизированного преобразования ИТВР в РТР представлена на рис. 4.5.

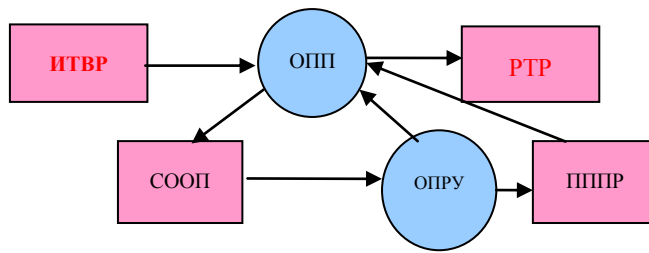


Рис. 4.5. Четвертая операторная форма модели автоматизированного преобразования ИТВР в РТР

Используя промежуточную операторную модель автоматизированного преобразования ИТВР в РТР, можно получить в операторной форме следующие выражение:

$$РТР = ОПП (ИТВР, СООП, ПППР);$$

$$ПППР = ОПРУ (СООП).$$

Эти выражения могут быть использованы для начальных изысканий по поводу реализации автоматизированной подсистемы преобразования ИТВР в РТР.

Построенная модель является предельным упрощением описания реального процесса преобразования информации табличного вида в реляционные таблицы.

В полной операторной модели методики преобразования необходимо рассмотреть все формы представления ИТВР. При этом необходимо рассмотреть как самый благоприятный случай представления ИТВР для преобразования в РТР, так и все неблагоприятные случаи.

Нетрудно предположить, что самым благоприятным случаем представления ИТВР для их автоматизированного преобразования в РТР является такой случай, когда ИТВР удовлетворяют требованиям к реляционным таблицам. Тогда значительная часть работ по преобразованию сводится к простому импорту таблиц, который обеспечивается во всех инструментальных СУБД. Поэтому в модель методики следует включить манипуляции, связанные с импортом таблиц, которые представлены не в формате инструментальной СУБД или в формате другой инструментальной

СУБД. Для этого необходимо в модель ввести соответствующий фрагмент, который отражен на рис. 4.6.

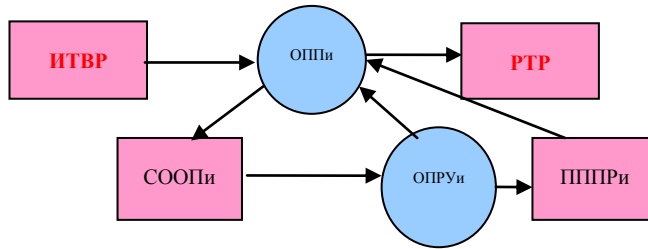


Рис. 4.6. Операторная модель импорта ИТВР в РТР

Здесь:

ОППи – оператор импорта таблиц ИТВР;

СООПи – состояние модели, соответствующее автоматической генерации системы оценок результатов импорта;

ОПРУи – оператор принятия решений относительно управляющих воздействий после анализа системы оценок результатов импорта;

ПППРи – состояние модели, соответствующее формированию системы оценок результатов импорта разработчиком РТР;

Даже в самом благоприятном случае остается необходимость назначать первичные и внешние ключи, формировать связи между таблицами. В ИТВР их просто нет.

В реальной ситуации ИТВР, как правило, не удовлетворяет требованиям к реляционным таблицам. Методика должна обеспечить решение задачи для всех случаев.

Для полноты картины приведем таблицу, в которой представлены возможные ситуации представления ИТВР (таблица 4.1).

Таблица 4.1

№	НРЛ	НРЛвп	НРЛвс	НКЛ	НКЛг	НКЛв	НСВ
1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	1
3	0	0	0	0	0	1	0
4	0	0	0	0	0	1	1
5	0	0	0	0	1	0	0
6	0	0	0	0	1	0	1

7	0	0	0	0	1	1	0
8	0	0	0	0	1	1	1
9	0	0	0	1	0	0	0
10	0	0	0	1	0	0	1
11	0	0	0	1	0	1	0
12	0	0	0	1	0	1	1
13	0	0	0	1	1	0	0
14	0	0	0	1	1	0	1
15	0	0	0	1	1	1	0
16	0	0	0	1	1	1	1
17	0	0	1	0	0	0	0
18	0	0	1	0	0	0	1
19	0	0	1	0	0	1	0
20	0	0	1	0	0	1	1
21	0	0	1	0	1	0	0
22	0	0	1	0	1	0	1
23	0	0	1	0	1	1	0
24	0	0	1	0	1	1	1
25	0	0	1	1	0	0	0
26	0	0	1	1	0	0	1
27	0	0	1	1	0	1	0
28	0	0	1	1	0	1	1
29	0	0	1	1	1	0	0
30	0	0	1	1	1	0	1
31	0	0	1	1	1	1	0
32	0	0	1	1	1	1	1
33	0	1	0	0	0	0	0
34	0	1	0	0	0	0	1
35	0	1	0	0	0	1	0
36	0	1	0	0	0	1	1
37	0	1	0	0	1	0	0
38	0	1	0	0	1	0	1
39	0	1	0	0	1	1	0
40	0	1	0	0	1	1	1
...
129	1	1	1	1	1	1	1

Здесь:

НРЛ – таблицы ИТВР не удовлетворяют традиционным требованиям к реляционным таблицам;

НКЛ – в таблицах ИТВР отсутствуют данные о ключевых полях;

НСВ – в таблицах ИТВР отсутствуют данные о связях между таблицами.

Выше рассмотрен фрагмент модели для случая 1, а именно, случай когда: таблицы ИТВР удовлетворяют требованиям к реляционным таблицам, в таблицах ИТВР имеются данные о ключевых полях, в таблицах ИТВР имеются данные о связях между таблицами;

НРЛвп – в таблицах ИТВР присутствуют внутренние подзаголовки;

НРЛвс – в таблицах ИТВР заголовки размещены в первом столбце;

НКЛт – для формирования внешнего ключа в таблицах необходимо более 3-х атрибутов;

НКЛв – в таблицах не назначены внешние ключи.

В работе [53] приведена близкая по смыслу таблица. Однако рассмотрены только первые три пункта (НРЛ, НКЛ, НСВ). Остальные пункты не рассматриваются. В связи с тем, что в данной работе учитываются все перечисленные сочетания, предполагается существенно расширить существующую методику.

Рассмотрим характерный случай 9, а именно случай когда: ключевые поля не определены, таблицы ИТВР удовлетворяют всем перечисленным требованиям к реляционным таблицам, в таблицах ИТВР имеются данные о связях между таблицами.

Соответствующая операторная модель выглядит следующим образом (рис. 4.7).

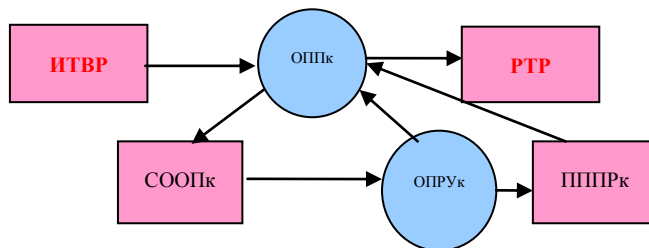


Рис. 4.7. Операторная модель процесса автоматизированного назначения ключевых полей в заполненных ИТВР

Здесь:

ОПК – оператор назначения ключевых полей в ИТВР;

СООПк – состояние модели, соответствующее автоматической генерации системы оценок результатов назначения ключевых полей в ИТВР;

ОПРУк – оператор принятия решений относительно управляющих воздействий после анализа системы оценок результатов назначения ключевых полей в ИТВР;

ПППР_к – состояние модели, соответствующее формированию системы оценок результатов назначения ключевых полей в ИТВР разработчиком РТР.

Этот фрагмент модели внешне напоминает соответствующий фрагмент, представленный в работе [37], хотя и включает в себя и другие компоненты. Но модель новой методики должна отражать не только новый состав компонент, но и разработанные автором новые способы и связи между ними. К таким способам прежде всего относятся:

- способ назначения ключевых полей на основе 3-х и более атрибутов;
- способ назначения внешних ключевых полей (глава 3).

Отразим их на рис. 4.8.

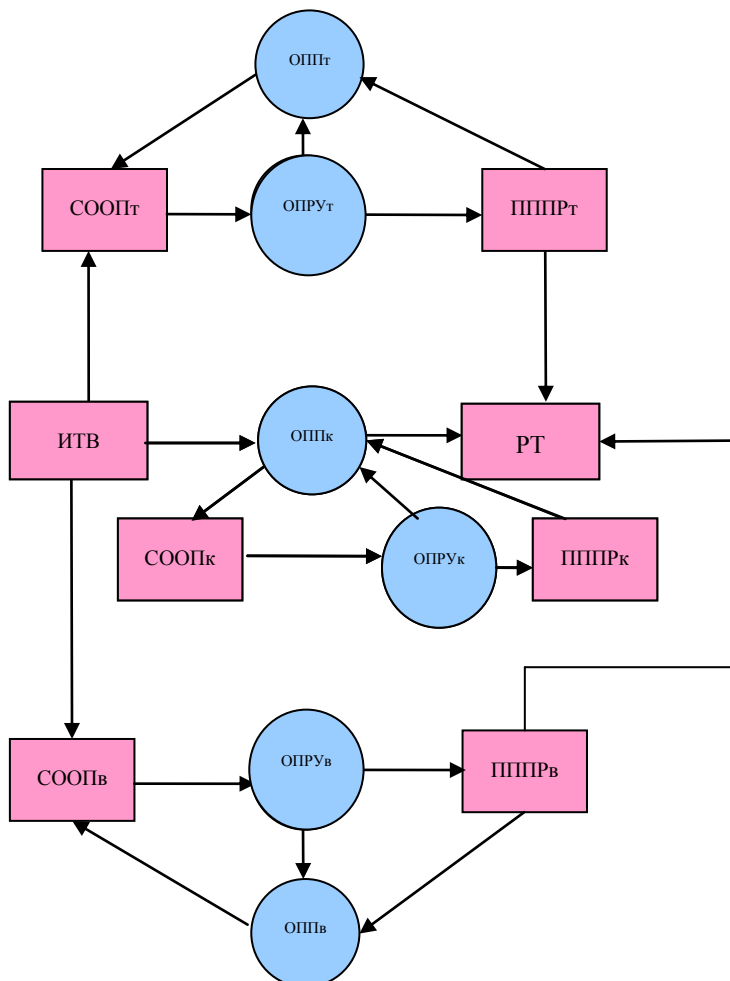


Рис. 4.8. Фрагмент расширенной операторной модели процесса назначения ключевых полей.

Здесь:

ОППт – оператор назначения ключевых полей с числом атрибутов более 3-х в ИТВР;

СООПт – состояние модели, соответствующее автоматической генерации системы оценок результатов назначения ключевых полей с числом атрибутов более 3-х в ИТВР;

ОПРУт – оператор принятия решений относительно управляющих воздействий после анализа системы оценок результатов назначения ключевых полей с числом атрибутов более 3-х в ИТВР;

ПППРт – состояние модели, соответствующее формированию системы оценок результатов назначения ключевых полей с числом атрибутов более 3-х в ИТВР разработчиком РТР;

ОППв – оператор назначения внешних ключевых полей в ИТВР;

СООПв – состояние модели, соответствующее автоматической генерации системы оценок результатов назначения внешних ключевых полей в ИТВР;

ОПРУв – оператор принятия решений относительно управляющих воздействий после анализа системы оценок результатов назначения внешних ключевых полей в ИТВР;

ПППРв – состояние модели, соответствующее формированию системы оценок результатов назначения внешних ключевых полей в ИТВР разработчиком РТР.

Рассмотрим еще один характерный случай 16, а именно случай, когда: таблицы ИТВР не удовлетворяют перечисленным требованиям к реляционным таблицам, в таблицах ИТВР имеются данные о связях между таблицами, в таблицах ИТВР имеются все перечисленные сведения о ключевых полях.

Соответствующая операторная модель выглядит следующим образом (рис. 4.9).

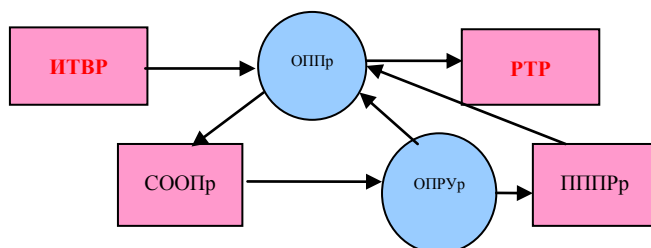


Рис. 4.9. Операторная модель процесса автоматизированного преобразования ИТВР в РТР

Здесь:

ОППр – оператор преобразования ИТВР в РТР;

СООПр – состояние модели, соответствующее автоматической генерации системы оценок результатов преобразования ИТВР в РТР;

ОПРУр – оператор принятия решений относительно управляющих воздействий после анализа системы оценок результатов преобразования ИТВР в РТР;

ПППРр – состояние модели, соответствующее формированию системы оценок результатов преобразования ИТВР в РТР разработчиком РТР.

Этот фрагмент модели внешне напоминает соответствующий фрагмент, представленный в работе [37], хотя и включает в себя и другие компоненты. Но, как сказано выше, модель новой методики должна отражать не только новый состав компонент, но и разработанные автором новые способы и связи между ними. К таким способам прежде всего относятся:

- способ исключения внутренних подзаголовков в ИТВР;
- способ исключения заголовков, сформированных в первом столбце таблицы ИТВР;
- способ исключения сложных подзаголовков в ИТВР;
- способ исключения повторяющихся записей в ИТВР (глава 2).

Последний способ тривиален и реализуется средствами инструментальной СУБД. Предпоследний способ является комбинированием двух предыдущих способов. Поэтому в расширенной операторной модели преобразования ИТВР в РТР отобразим только первые два способа (рис. 4.10).

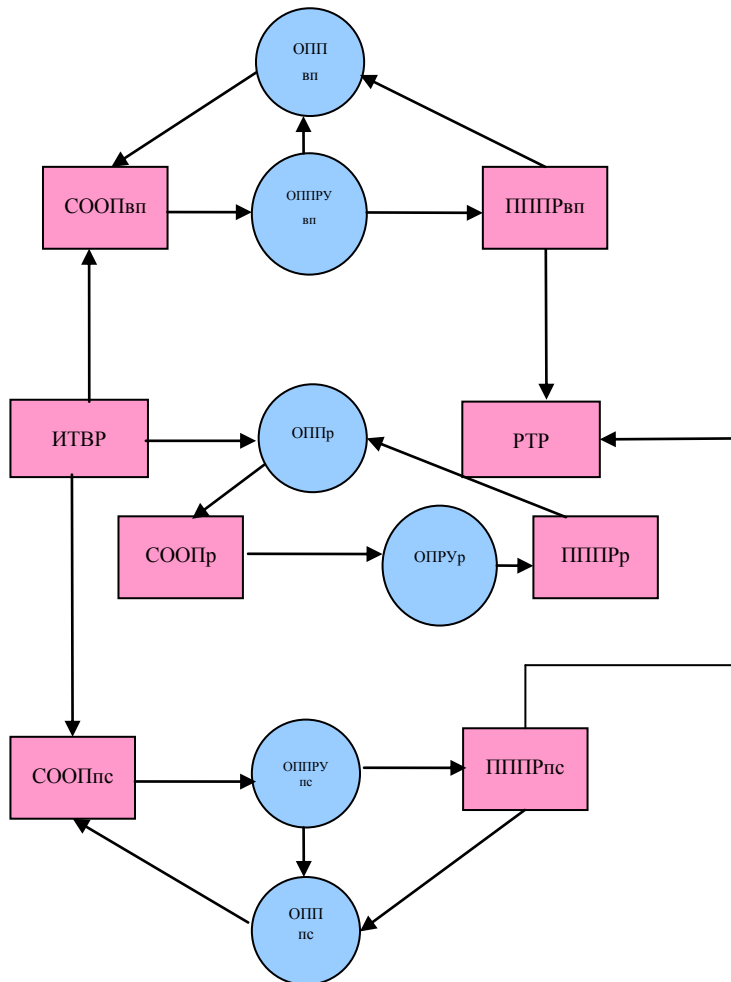


Рис. 4.10. Расширенная операторная модель преобразования ИТВР в РТР

Здесь:

ОППвп – оператор исключения внутренних подзаголовков в ИТВР;

СООПвп – состояние модели, соответствующее автоматической генерации системы оценок результатов исключения внутренних подзаголовков в ИТВР;

ОПРУвп – оператор принятия решений относительно управляющих воздействий после анализа системы оценок результатов исключения внутренних подзаголовков в ИТВР;

ПППР_{вп} – состояние модели, соответствующее формированию системы оценок результатов исключения внутренних подзаголовков в ИТВР разработчиком РТР;

ОПП_{пс} – оператор исключения заголовков первого столбца в ИТВР;

СООП_{пс} – состояние модели, соответствующее автоматической генерации системы оценок результатов исключения заголовков первого столбца в ИТВР;

ОПРУ_{пс} – оператор принятия решений относительно управляющих воздействий после анализа системы оценок результатов исключения заголовков первого столбца в ИТВР;

ПППР_{пс} – состояние модели, соответствующее формированию системы оценок результатов исключения заголовков первого столбца в ИТВР разработчиком РТР;

Строго говоря, оправданно рассмотреть все 129 случаев и в соответствии с каждым случаем детализировать операторную модель методики. Но выполненные выкладки для всех случаев типовые и схожи с рассмотренными сочетаниями. В связи с этим, а также в связи с тем, что близкие по форме выкладки приведены в работах [37, 53], представлена результирующая операторная модель процесса преобразования ИТВР в РТР, которая соответствует случаю 129 (рис. 4.11). Кроме того, последнее сочетание перекрывает все возможные случаи.

у

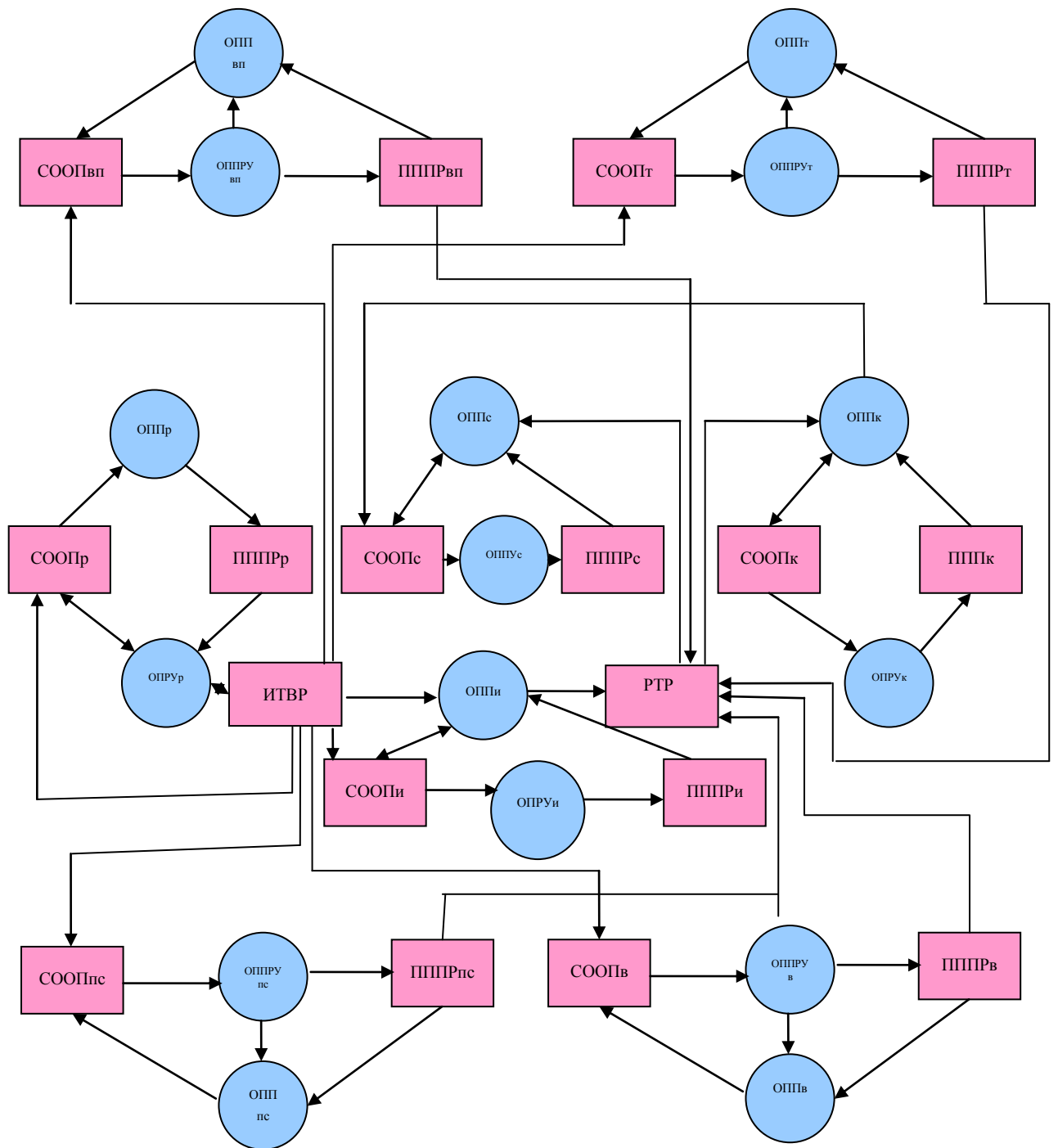


Рис. 4.11. Результирующая операторная модель процесса преобразования ИТВР в РТР

Здесь:

ОППс – оператор формирования связей между таблицами ИТВР;

СООПс – состояние модели, соответствующее автоматической генерации системы оценок результатов импорта;

ОПРУс – оператор принятия решений относительно управляющих воздействий после анализа системы оценок результатов формирования связей;

ПППРс – состояние модели, соответствующее формированию системы оценок результатов формирования связей разработчиком РТР.

При построении результирующей операторной модели процесса преобразования ИТВР в РТР использованы фрагменты модели, представленные на рисунках 4.8 - 4.10. Принято во внимание то, что модели ИТВР и РТР для всех случаев едины. Принято во внимание то, что модель, представленная на рис. 4.11, отражает все возможные сочетания.

Эта операторная модель соответствует методике автоматизированного формирования РТВ на основе использования заполненных ИТВР. Она позволяет определить состав процедур, которые необходимо реализовать для создания полноценной программной подсистемы, может быть использована в качестве формализованного руководства пользователя и выступает в качестве источника для дальнейшей формализации процесса преобразования и его исследования.

Модель (рис. 4.11) позволяет назвать основные процедуры подсистемы, которые необходимо реализовать. Этим процедурам соответствуют следующие кортежи операторов:

$$ОПП = \{ОППи, ОППр, ОППк, ОППс, ОППт, ОППв, ОППвн, ОППнс\}.$$

$$ОПРУ = \{ОПРУи, ОПРУр, ОПРУк, ОПРУс, ОПРУт, ОПРУв, ОПРУвн, ОПРУнс\}.$$

$$ПППР = \{ПППРу, ПППРр, ПППРк, ПППРс, ПППРн, ПППРв, ПППРвн, ПППРнс\}.$$

$$СООП = \{СООПи, СООПр, СООПк, СООПс, СООПн, СООПв, СООПвн, СООПнс\}.$$

До построения операторной модели автоматизированного преобразования ИТВР в РТР методика представлялась исходя из здравого смысла. Разработка человеко-машинных процедур формирования РТР на основе ИТВР в этом случае чревата ошибками по следующим причинам:

- последовательность использования разрабатываемых способов и алгоритмов должна быть строго определена;
- взаимосвязь между компонентами подсистемы преобразования при описании методики, исходя из здравого смысла, неочевидна;
- неформально затруднительно в полном объеме определить состав необходимых компонент системы преобразования.

Все перечисленные операторы и системы оценок представляют собой средства, которые необходимо разработать. Их разработке посвящено приложение 1.

4.3 Анализ методики для обнаружения дефектов ее представления

Мотивация перехода от операторной модели к модели, построенной на основе аппарата сетей Петри

Операторная модель, которая построена на основе анализа отличий моделей ИТВР и РТР, позволила выявить основные компоненты подсистемы преобразования ИТВР в РТР, определить порядок использования компонент, сформировать связи между компонентами. Поэтому она может быть использована в качестве начальной формализации для разработки средств преобразования ИТВР в РТР. Правда нет полной уверенности в том, что эта модель адекватна предмету исследования и свободна от концептуальных ошибок. Для исключения ошибок на ранних стадиях разработки подсистемы необходимо исследовать ее следующие характеристики:

- обеспечение баланса информационных потоков (устойчивость);
- достижимость всех состояний (живость);
- отсутствие тупиковых ситуаций при работе подсистемы (живость);

- отсутствие ситуаций, когда подсистема приходит в нестационарное состояние и число информационных потоков превышает критическую отметку (безопасность);

Несмотря на достоинства операторной модели, она не позволяет выполнить исследование перечисленных характеристик подсистемы.

Так как методика преобразования ИТВР в РТР автоматизирована, то есть в соответствии с ней в процессе формирования РТР активно участвует разработчик, имеет смысл использовать имитационные модели [74 - 75]. В рамках методики используются следующие компоненты: ИТВР, РТР, разработчик, способы, оценки. Причем эти компоненты не конкретизируются. Фиксируется факт их наличия, связи между ними, место в системе преобразования, последовательность и правила их использования. Представление такого рода наиболее близко к системному уровню, а на системном уровне обычно используют системы массового обслуживания и сети Петри [32, 53,74-75]. Кроме того, сети Петри позволяют исследовать моделируемые процессы на устойчивость, живость, безопасность и позволяют тем самым расширить круг исследований по сравнению с операторной моделью. В свою очередь операторная модель служит прототипом для разработки сетевой модели.

Поэтому для описания и исследования методики по автоматизированному формированию РТР на основе использования заполненных ИТВР используется аппарат сетей Петри [32, 53,74-75].

Классическая сеть Петри включает в себя следующие элементы: множество положений S , множество переходов T и отношение связанности F [32, 53,74-75].

Сетью Петри называется набор

« $N = (S, T, F)$, где

1. $S = \{s_1, \dots, s_n\}$ – множество мест;

2. $T = \{t_1, \dots, t_n\}$ – множество переходов таких,

что $S \cap T = \emptyset$.

3. $F \subseteq \mu S \times T \times \mu S$ – отношение

При этом:

(а)

$$\forall \langle Q_1', t_1, Q_1'' \rangle, \langle Q_2', t_2, Q_2'' \rangle \in F : \langle Q_1', t_1, Q_1'' \rangle \neq \langle Q_2', t_2, Q_2'' \rangle \Rightarrow t_1 \neq t_2$$

(б)

$$\{t | \langle Q', t, Q'' \rangle \in F\} = T \quad [32, 53].$$

Места нередко называют позициями или положениями.

Для каждого перехода $t \in T$ существует единственный элемент $\langle Q', t, Q'' \rangle \in F$, задающий для него входное мультимножество мест Q' и выходное мультимножество Q'' . Дадим определение входному и выходному мультимножеству.

Определение: входное и выходное мультимножества мест и переходов

Пусть задана сеть $N=(S, T, F)$.

Если для некоторого перехода t имеется $\langle Q', t, Q'' \rangle \in F$, то обозначают

$$\bullet t = Q' = \{(s, n) | (t, n) \in s^\bullet\}, t^\bullet = Q'' = \{(s, n) | (t, n) \in^\bullet s\}$$

;

$$\bullet s = \{(t, n) | (s, n) \in t^\bullet\}, s^\bullet = \{(t, n) | (s, n) \in^\bullet t\}.$$

Говорят, что $\|\bullet t\|$ – входные, а $\|t^\bullet\|$ – выходные места перехода t . Таким образом, согласно определению, справедливо $\forall t \in T : \langle \bullet t, t, t^\bullet \rangle \in F$.

Место s инцидентно переходу t , если $s \in \bullet t$ или $s \in t \bullet$.

Функции $\bullet(\cdot)$ и $(\cdot)\bullet$ могут быть расширены на мультимножества переходов. Пусть $\Theta \in \mu\Gamma$ есть мультимножество переходов такое, что $\Theta = n_1 t_1 + n_2 t_2 + \dots + n_k t_k$. Тогда положим

$$\bullet\Theta = n_1 \bullet t_1 + n_2 \bullet t_2 + \dots + n_k \bullet t_k$$

$$\Theta\bullet = n_1 t_1 \bullet + n_2 t_2 \bullet + \dots + n_k t_k \bullet \gg [74-75].$$

В графическом представлении сети Петри для изображения положений используются окружности, а для изображения переходов – отрезки (как правило, вертикальные).

Операторная модель, предложенная в работе, может быть представлена набором $M = (C, O, L)$, где

$$1. C = COOP \cup PPPP \cup MO$$

$COOP = \{CO_1, \dots, CO_q, \dots, CO_k\}$ – множество систем оценок проектных решений;

$$PPPP = \{CP_1, \dots, CP_f, \dots, CP_z\}$$
 – множество систем принятия решения;

$$MO = \{MO_1, \dots, MO_b, \dots, MO_p\}$$
 – множество моделей.

$$COOP \cap PPPP \cap MO = \emptyset$$

Для исходной операторной модели множества представлены следующим образом:

$$PPPP = \{PPPP_r, PPPP_c, PPPP_k, PPPP_u, PPPP_m, PPPP_v, PPPP_{vn}, PPPP_{nc}\};$$

$$COOP = \{COOP_r, COOP_c, COOP_k, COOP_u, COOP_m, COOP_v, COOP_{vn}, COOP_{nc}\};$$

$$MO = \{ИТВР, РТР\}.$$

При уточнении операторной модели состав множеств может быть модифицирован.

2. $O = \{ОПП, ОПРУ\}$.

$ОПП = \{ОППр, ОППс, ОППк, ОППи, ОППт, ОППв, ОППен, ОППнс \}$.

$ОПРУ = \{ ОПРУр, ОПРУс, ОПРУк, ОПРУи, ОПРУт, ОПРУв, ОПРУен, ОПРУнс \}$.

При уточнении операторной модели состав множеств может быть модифицирован.

3. $L \subseteq \mu C \times O \times \mu C$ – отношение инцидентности такое, что

(а) $\forall \langle G'_1, o_1, G''_1 \rangle, \langle G'_2, o_2, G''_2 \rangle \in L : \langle G'_1, o_1, G''_1 \rangle \Rightarrow o_1 \neq o_2$;

(б) $\{o | \langle G', o, G'' \rangle \in L\} = O$.

Условие в пункте 3 говорит о том, что для каждого оператора $o \in O$ существует единственный элемент $\langle G', o, G'' \rangle \in L$, задающий для него входное мультимножество систем оценки и принятия решений G' и выходное мультимножество G'' .

Входное и выходное мультимножества систем оценки и принятия решений, операторов операторной модели определяются по аналогии с входным и выходным мультимножествами мест переходов сети Петри

Формирование сетевой модели

Между набором $N = (S, T, F)$ сети Петри и набором $M = (C, O, L)$ операторной модели установим соответствие таким образом, что между элементами наборов обеспечивалось взаимно-однозначное соответствие: $C \leftrightarrow S; O \leftrightarrow T; L \leftrightarrow F$. [32]

Как следствие установленного соответствия:

$$|C| = |S|; |O| = |T|; |L| = |F|.$$

В итоговой операторной модели процесса преобразования ИТВР в РТР разметим положения сети (р) и переходы сети (t). Результат разметки приведен на рис. 4.12.

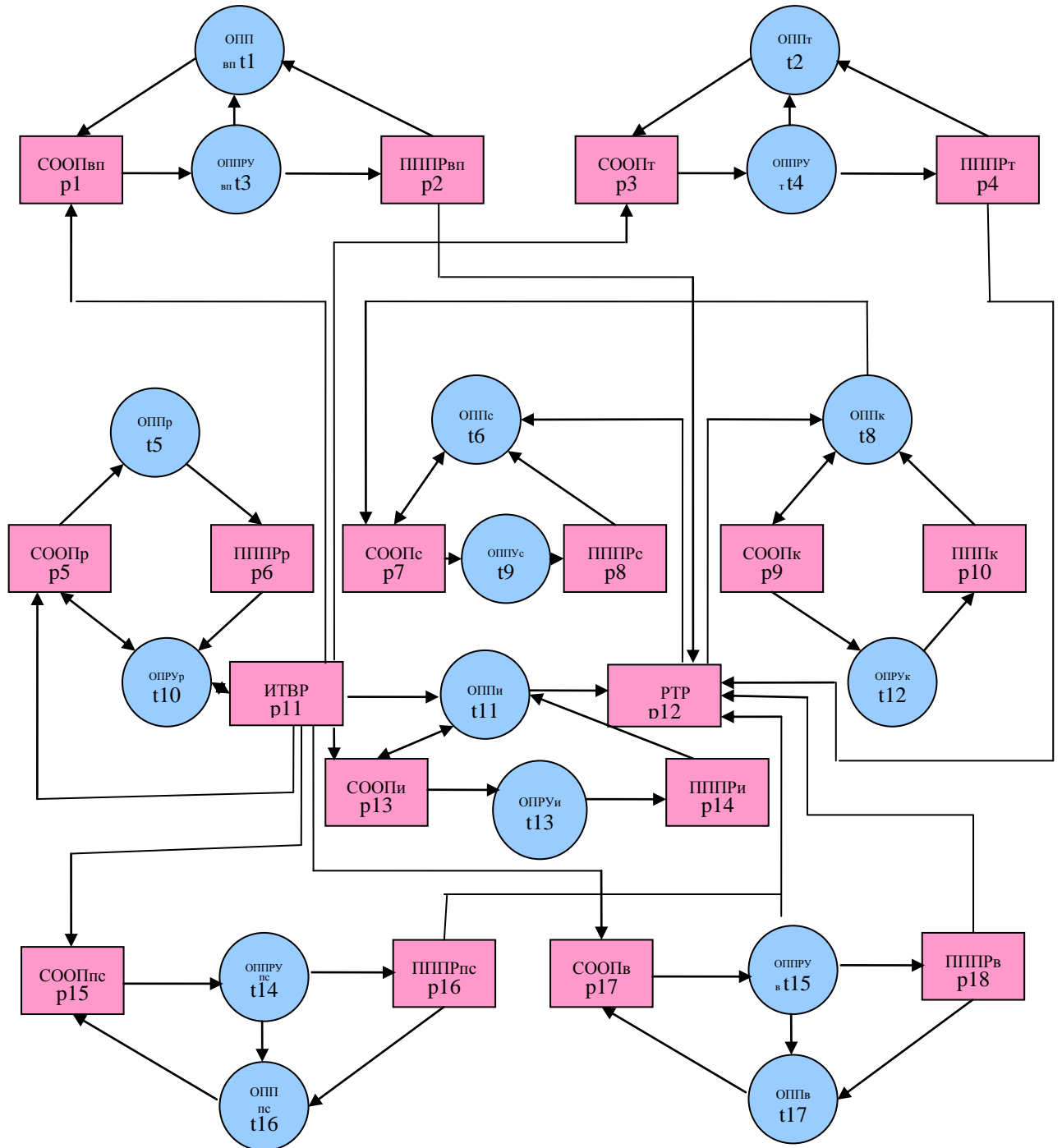


Рис. 4.12. Результат разметки итоговой операторной модели

Следуя описанным пунктам 1-3, установленным соответствиям между компонентами операторной и сетевой модели, а также исходя из рис. 4.12,

получим начальную модель процесса автоматизированного преобразования ИТВР в РТР в виде сети Петри (рис . 4.13).

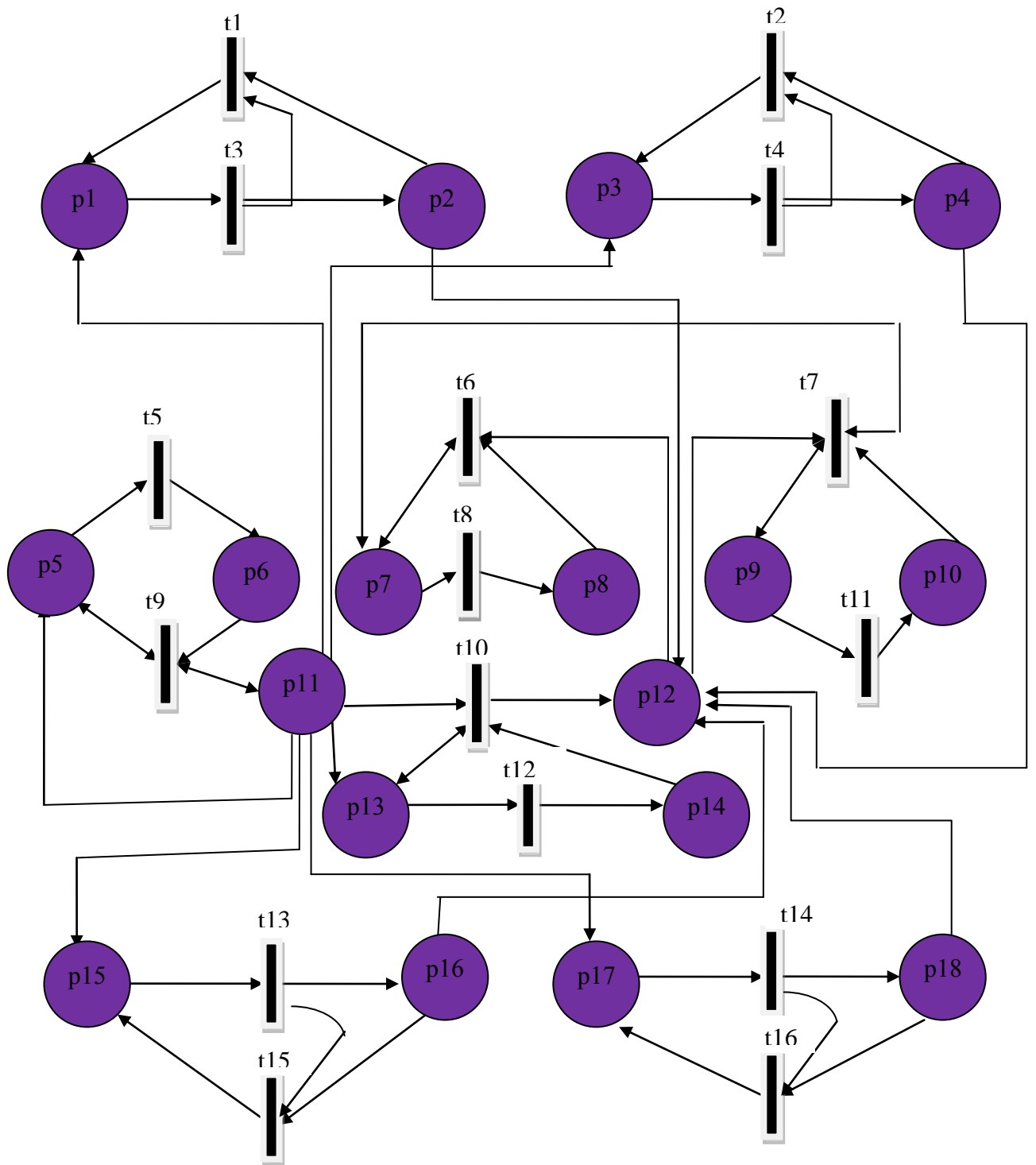


Рис. 4.13. Начальная сетевая модель процесса автоматизированного преобразования ИТВР в РТР в виде сети Петри

На основе построенной сетевой модели можно выполнить исследование описанного процесса. В ходе исследования можно выявить принципиальные ошибки, допущенные в ходе построения модели. Затем исключить эти ошибки. И, в конечном итоге, на основе сетевой модели свободной от принципиальных ошибок, построить соответствующую операторную модель процесса формирования РТР на основе использования заполненных ИТВР.

В частности можно проверить сетевую модель на устойчивость:

« Сеть Петри является устойчивой, если она имеет потоковое назначение $\varphi_i > 0$ для каждого $t_i \in T$. Потоковое назначение – это функция, которая приписывает каждому переходу $t_i \in T$ положительное рациональное число, называемое ее потоком. Потоковое назначение для сети Петри должно удовлетворять требованиям:

- каждая дуга переносит поток, равный потоку перехода, к которому присоединена эта дуга;
- для каждого положения сумма потоков входных дуг должна равняться сумме потоков выходных дуг » [93].

Для каждого перехода сети рис. 4.13 назначим поток φ_i . Для каждого положения P_i запишем уравнения потоков.

Уравнения сведены в таблицу 4.2.

Таблица 4.2

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12	t13	t14	t15	t16	
p1	+φ1		-φ3														=0
p2	-φ1		+φ3														=0
p3		+φ2		-φ4													=0
p4		-φ2		+φ4													=0
p5					-φ5				-φ9								=0
p6					+φ5			-φ9	+φ9								=0
p7						+φ6	+φ7	-φ8									=0
p8						-φ6		+φ8									=0
p9							+φ7				-φ11						=0
							-φ7				+φ11						=0
p10																	=0
p11								+φ9	-φ9	-φ10							=0
p12						-φ6	-φ7			+φ10							=0
p13										+φ10							=0
										-φ10							=0
										-φ12							=0
p14										-φ10		+φ12					=0
p15													-φ13		+φ15		=0
p16													+φ13		-φ15		=0
p17														-φ14		+φ16	=0
p18														+φ14		-φ16	=0

Перепишем уравнения потоков в более наглядном виде:

$$+\varphi_1-\varphi_3=0; \quad -\varphi_1+\varphi_3=0; \quad +\varphi_2-\varphi_4=0; \quad -\varphi_2+\varphi_4=0; \quad -\varphi_5-\varphi_9+\varphi_9=0; \quad +\varphi_5-\varphi_9=0;$$

$$+\varphi_6-\varphi_6+\varphi_7-\varphi_7=0; \quad -\varphi_6+\varphi_8=0; \quad +\varphi_7-\varphi_7-\varphi_{11}=0; \quad -\varphi_7+\varphi_{11}=0; \quad +\varphi_9-\varphi_9-\varphi_{10}=0;$$

$$-\varphi_6-\varphi_7+\varphi_{10}=0; \quad +\varphi_{10}-\varphi_{10}-\varphi_{12}=0; \quad -\varphi_{10}+\varphi_{12}=0; \quad -\varphi_{13}+\varphi_{15}=0;$$

$$+\varphi_{13}-\varphi_{15}=0; \quad -\varphi_{14}+\varphi_{16}=0; \quad +\varphi_{14}-\varphi_{16}=0;$$

Преобразуем уравнения и получим:

$$\varphi_1 = \varphi_3; \varphi_3 = \varphi_1; \varphi_2 = \varphi_4; \varphi_2 = \varphi_4; \varphi_5 = 0; \varphi_5 = \varphi_9; 0 = 0; \varphi_6 = \varphi_8;$$

$$-\varphi_{11} = 0; \varphi_7 = \varphi_{11}; \varphi_{10} = 0; \varphi_{10} = \varphi_6 + \varphi_7; \varphi_{10} = \varphi_{10} + \varphi_{12};$$

$$\varphi_{10} = \varphi_{12}; \varphi_{15} = \varphi_{13}; \varphi_{13} = \varphi_{15}; \varphi_{14} = \varphi_{16}; \varphi_{14} = \varphi_{16};$$

Исключим уравнения, соответствующие условию баланса потоков, и тождество. Получим:

$$\varphi_5 = 0; \varphi_{11} = 0; \varphi_{10} = 0; \varphi_{10} = \varphi_6 + \varphi_7; 0 = \varphi_{12};$$

Все эти уравнения не соответствуют условию баланса потоков. Кроме того, после выполнения подстановок получим:

$$\varphi_9 = 0; \varphi_7 = 0;$$

Эти уравнения также не соответствуют условию баланса потоков.

Такая ситуация имеет место, когда сеть Петри не является устойчивой. Из этого следует то, что и процессы, которые описывает данная сеть, неустойчивы.

В связи с этим необходимы преобразования сетевой модели, в ней допущены принципиальные ошибки.

Выполним анализ уравнения $\varphi_5 = 0$; Это уравнение связано с положением сети p_5 . А p_5 , в свою очередь, соответствует состоянию операторной модели СООПр. СООПр – состояние модели, соответствующее автоматической генерации системы оценок результатов преобразования ИТВР в РТР. В сетевой модели положение p_5 непосредственно связано с положением p_{11} . Положению p_{11} соответствует модель ИТВР в операторной модели. Таким образом представлена связь, которая не предусматривает процедур выполнения процедур генерирующих систему оценок результатов преобразования ИТВР в РТР. Это упущение. Процедуры генерации оценок результатов преобразования ИТВР в РТР необходимы. Для их отражения

введем дополнительный переход t_{17} (рис.4.14). Запишем уравнения потоков для положения p_5 :

$$\varphi_{17} + \varphi_9 = \varphi_9 + \varphi_5, \text{ или } \varphi_{17} = \varphi_5$$

Противоречий нет.

Рассмотрим уравнения:

$$\varphi_{11} = 0; \quad 0 = \varphi_{12};$$

Потоковые значения φ_{11} и φ_{12} соответствуют положения сети p_{11} и p_{12} . А они соответствуют ИТВР и РТР в операторной модели. Это начальное и целевое состояния операторной и сетевой моделей. Иницирует и завершает процесс преобразования ИТВР в РТР разработчик. Таким образом в этих состояниях модели превалирует человеческий фактор. Учитывая это, можно констатировать, что уравнения потоков для этих особых состояний может быть в принципе любое. В нашем случае даже показательно, что начальное и конечное потоковые значения нулевые. Особые положения для наглядности выделим (рис. 4.14).

Рассмотрим уравнения:

$$\varphi_{10} = 0; \quad \varphi_{10} = \varphi_6 + \varphi_7;$$

Первое уравнение получилось из уравнения потоков для состояния p_{11} (см. таблицу 4.2). Как сказано выше это состояние особое и соответствует модели ИТВР. С учетом человеческого фактора баланс потоков для положения p_{11} необязателен.

Второе уравнение получилось из уравнения потоков для состояния p_{12} (см. таблицу 4.2). Это состояние особое и соответствует модели РТР. С учетом человеческого фактора баланс потоков для положения p_{12} необязателен.

Рассмотрим уравнения:

$$\varphi_9 = 0; \quad \varphi_7 = 0;$$

Уравнение $\varphi_9=0$ получено из $-\varphi_5-\varphi_9+\varphi_9=0$; $+\varphi_5-\varphi_9=0$; или

$$\varphi_5=0; \varphi_5=-\varphi_9;$$

Для положения φ_5 ситуация исправлена – введен дополнительный переход t_{17} . Таким образом исправлена ситуация и для φ_9 .

Уравнение $\varphi_7=0$; получено из $+\varphi_7-\varphi_7-\varphi_{11}=0$; $-\varphi_7+\varphi_{11}=0$; или

$$\varphi_{11}=0; \varphi_7=\varphi_{11};$$

φ_{11} соответствует, как сказано выше, особому положению p_{11} (ИТВР).

Для этого положения соблюдение баланса потоковых значений необязательно.

Несмотря на то, что положения сети p_{11} и p_{12} обладают особым статусом, по правилам формирования сети Петри эти положения не могут быть непосредственно связаны с другими положениями. Связи такого типа имеют место (рис. 4.13). Для них введем дополнительные переходы – t_{18} - t_{26} (рис. 4.14).

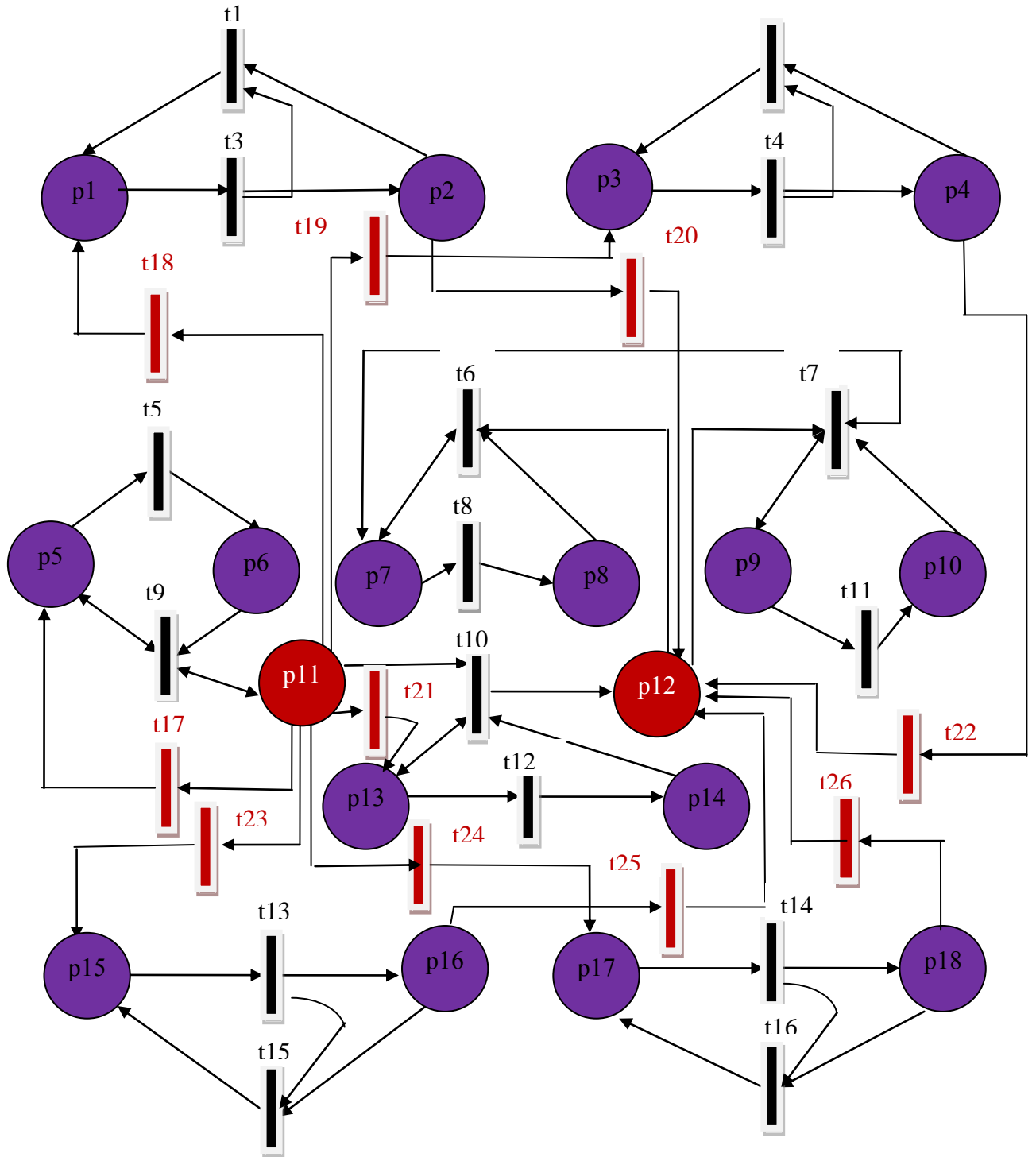


Рис. 4.14. Окончательная сетевая модель процесса автоматизированного преобразования ИТВР в РТР в виде сети Петри

Отразим на операторной модели процесса автоматизированного преобразования ИТВР в РТР изменения, сделанные в сетевой модели (рис. 4.15).

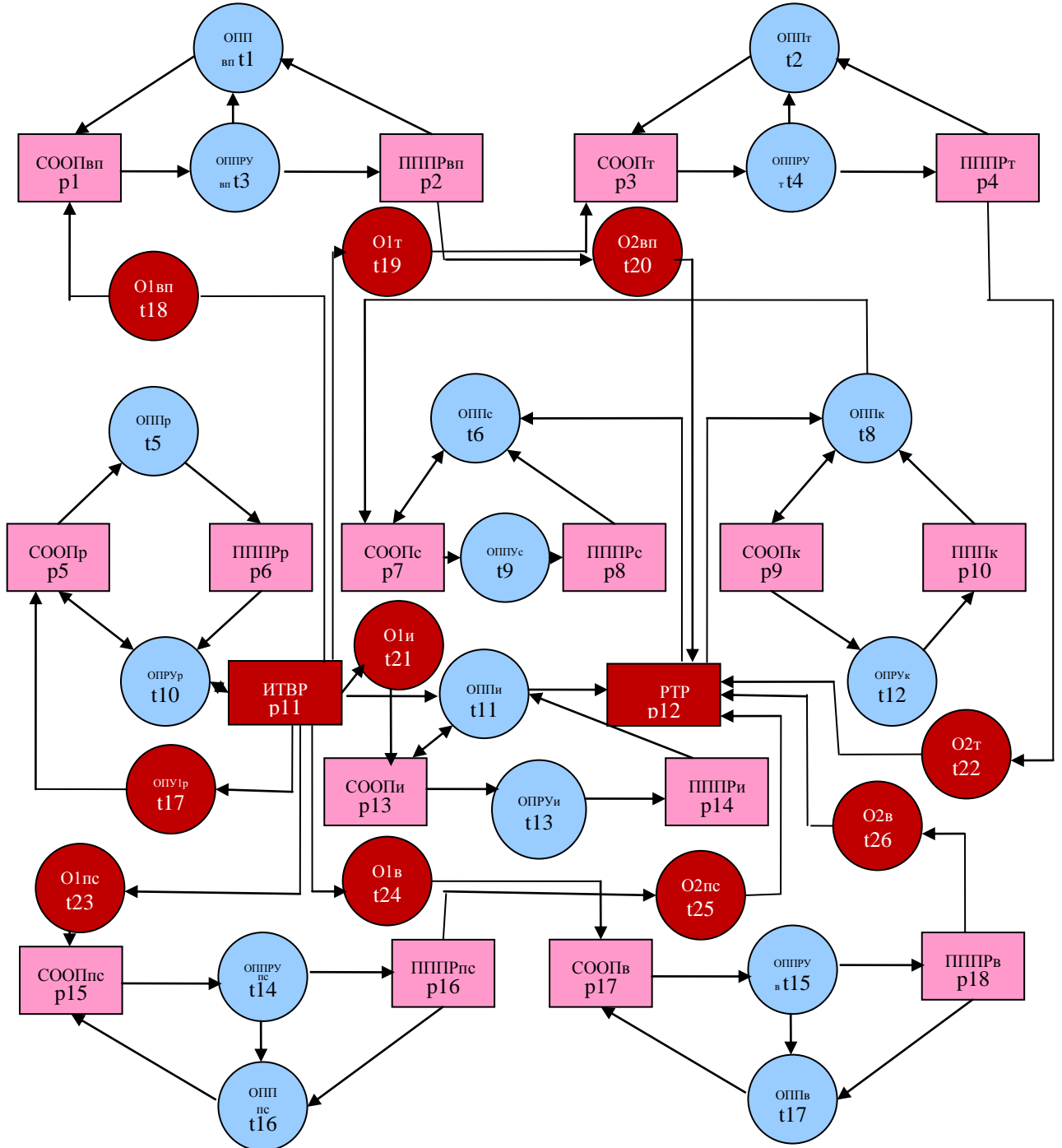


Рис. 4.15. Окончательная операторная модель процесса автоматизированного преобразования ИТВР в РТР

Как видно из рисунка, в операторной модели отображен новый оператор ОПУ1р. Он соответствует процедурам генерации оценок результатов преобразования ИТВР в РТР. Кроме того, выделены особые состояния модели процесса, которые соответствуют начальному и целевому состоянию ИТВР.

А самое существенное то, что сформировано 9 дополнительных операторов: О1вп, О1т, О2вп, О1и, О2т, О1пс, О1в, О2пс, О2в.

Прокомментируем их смысловое назначение:

О1вп – соответствует процедурам генерации оценок результатов исключения внутренних подзаголовков;

О1т – соответствует процедурам генерации оценок результатов формирования ключей на основе более 3-х атрибутов;

О2вп – соответствует процедурам анализа совместимости таблицы с исключенными внутренними подзаголовками с моделью РТР;

О1и – соответствует процедурам генерации оценок результатов импорта таблиц, соответствующих модели ИТВР;

О2т – соответствует процедурам анализа совместимости таблицы с назначенными ключевыми полями на основе более 3-х атрибутов с моделью РТР;

О1пс – соответствует процедурам генерации оценок результатов исключения заголовков в первом столбце;

О1в – соответствует процедурам анализа присутствия внутренних подзаголовков в объекте, соответствующем модели ИТВР;

О2пс – соответствует процедурам анализа совместимости таблицы с исключенными заголовками в первом столбце с моделью РТР;

О2в – соответствует процедурам анализа совместимости таблицы с исключенными внутренними подзаголовками с моделью РТР.

Окончательная операторная модель методики формирования РТР на основе использования заполненных ИТВР позволяет определить необходимые:

- номенклатуру способов, алгоритмов и процедур, которые следует разработать для реализации полнофункциональной подсистемы;
- номенклатуру систем оценок, которые следует разработать для реализации полнофункциональной подсистемы;
- состав связей между средствами в разрабатываемой подсистеме.

Операторная модель помимо этого отображает порядок и правила использования средств, которые применяются в процессе решения задач преобразования ИТВР в РТР в разрабатываемой подсистеме.

4.4 Анализ методики для обнаружения дефектов ее функционирования

Динамику функционирования системы можно моделировать перемещением маркеров в сети в соответствии с правилами перехода:

$$\ll M' (P) = M (P) - P (t_i) + H (t_i),$$

где $M (P) = (M (P1), M (P2), \dots, M (PN))$ – разметка сети

» [63].

Используем это определение для анализа функционирования сети, представленной на рис. 4.14, во времени.

Визуально перемещение маркеров в сети отображается точками в активном положении. Перемещение маркеров через переход осуществляется лишь тогда, когда во всех входных положениях перехода находятся маркеры. То есть тогда, когда все входные положения перехода активны.

Пример правильного перемещения маркеров в сети представлен на рис. 4.16.

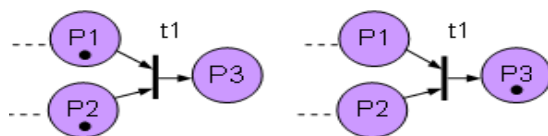


Рис. 4.16. Пример правильного перемещения маркеров в сети

Перемещение маркеров по сети позволит сделать вывод о том, что:

- является ли сеть живой, т.е. есть ли потенциальная возможность срабатывания всех переходов;
- является ли сеть достижимой, т.е. есть ли потенциальная возможность достигнуть всех положений сети.

На практике это означает:

- принципиальную возможность выполнения всех процедур, задействованных в предложенной методике;
- принципиальную возможность достижения всех состояний, предусмотренных в методике, и всех средств выполнения промежуточных и окончательных решений.

Для визуализации сети Петри, представленной на рис. 4.10 со всеми возможными разметками потребовалось несколько десятков рисунков. Эта работа выполнена. В результате чего сделан вывод, что сеть является живой и достижимой. А из этого следует то, что живой и достижимой является методика формирования РТР на основе ИТВР.

Для более компактного представления перемещения маркеров в сети воспользуемся представлением разметки сети в виде кортежей [75].

Кортеж в этом случае представляет набор цифр, число которых равно общему числу положений в сети. В нашем случае их 18. При этом порядковый номер цифры в кортеже ставится в соответствие номеру положения, а значения цифры – количеству маркеров в соответствующем положении сети.

Исходная маркировка сети Петри будет следующая:

$$M1 = \{0000000000010000000\}.$$

Срабатывание перехода t_{10} приведет к следующей разметке сети Петри:

$$M_2 = \{000010000010000000\}.$$

Срабатывание перехода t_5 приведет к следующей разметке сети Петри:

$$M_3 = \{000001000010000000\}.$$

Срабатывание перехода t_{10} приведет к следующей разметке сети Петри:

$$M_4 = \{000000000010000000\}.$$

Срабатывание перехода t_{21} приведет к следующей разметке сети Петри:

$$M_5 = \{000000000010100000\}.$$

Срабатывание перехода t_{13} приведет к следующей разметке сети Петри:

$$M_6 = \{00000000001001000\}.$$

Срабатывание перехода t_{11} приведет к следующей разметке сети Петри:

$$M_7 = \{00000000001100000\}.$$

Срабатывание переходов может осуществляться неограниченное число раз. При исследованиях выбран такой порядок перемещения маркеров, что для срабатывания всех переходов потребовалось менее двух сотен итераций.

В результате проверки сработали все переходы, поэтому данная сеть является достижимой, а следовательно достижимы все состояния объекта исследования методики процесса преобразования ИТВР в РТР.

Возможность срабатывания всех переходов позволяет свидетельствовать еще об одном ее свойстве – отсутствии тупиковых состояний. Это свидетельствует о том, что в объекте исследования отсутствуют тупиковые ситуации.

Перемещение маркеров в сетевой модели позволило сделать вывод о том, что данная сеть является безопасной, то есть выполняется условие:

« $\forall M(P) \subset R(P) \{ \forall P_i / M(P_i) \leq 1 \}$, где
 $i = 1, n; n = |P|; R(P)$ – разметки сети. » [63].

В связи с этим можно утверждать: моделируемый процесс выполняется в стационарном режиме.

Подводя итоги, можно сделать вывод о том, что разработанная модель обладает следующими свойствами. Она является: устойчивой, живой, достижимой, безопасной. Кроме того, в ней отсутствуют тупиковые положения.

Между сетевой и операторной моделью обеспечено взаимно однозначное соответствие. Таким образом и операторная модель процесса преобразования ИТВР в РТР обладает такими же свойствами.

В связи с этим разработанная операторная модель наряду с сетевой моделью является формальным описанием методики преобразования ИТВР в РТР. Эти описания могут быть использованы в качестве исходной формализации для разработки и реализации алгоритмов преобразования ИТВР в РТР.

В качестве другой формы представления методики преобразования ИТВР в РТР на рис. 4.17. приведена укрупненная схема взаимодействия разработчика и подсистемы в процессе преобразования ИТВР в РТР.

Следует обратить внимание, что под блоками “типовой процесс” подразумевается комплекс человеко-машинных процедур. Например, блок “Поиск ПК из нескольких атрибутов” включает в себя человеко-машинные процедуры поиска первичного ключа на основе двух и более атрибутов. И для каждого случая используются разные алгоритмы. Соответственно для каждого случая используется комплект программных процедур и комплект реакций разработчика, связанных с каждой процедурой.

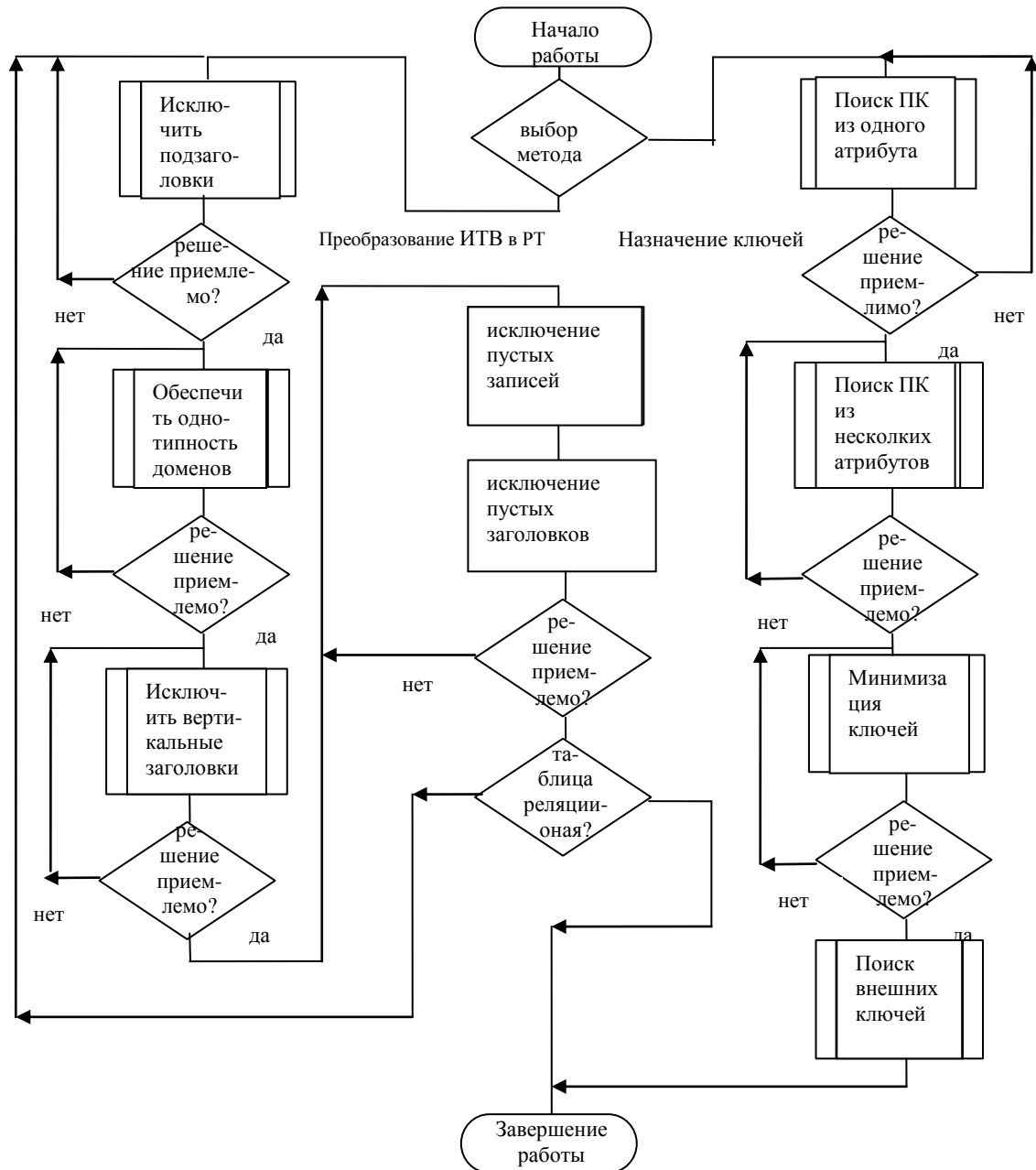


Рис. 4.17. Укрупненная схема взаимодействия разработчика и подсистемы в процессе преобразования ИТВР в РТР.

Выводы по главе 4

1. Сформулирована проблема формализации методики преобразования объектов, соответствующих модели ИТВР в объекты, соответствующие РТР. До настоящего времени эта проблема в полном объеме сформулирована не была.

2. Разработаны операторная и сетевая модели методики. Исследования, выполненные на основе этих моделей, позволили выявить и исключить ошибки представления и функционирования методики.

3. Операторная модель методики послужила основой для разработки подсистемы, которая ее реализует и позволяет исключить применение ручных методов, сократить трудоемкость преобразования, исключить дефекты преобразования и сократить время преобразования в десятки раз.

ОСНОВНЫЕ РЕЗУЛЬТАТЫ РАБОТЫ

1. Предложена расширенная модель информации табличного вида, которая отражает ранее нерассмотренные концептуальные особенности этих объектов.
2. Предложена расширенная модель реляционных таблиц, которая отражает ранее нерассмотренные концептуальные особенности этих объектов.
3. Разработан метод автоматизированного преобразования информации табличного вида в реляционные таблицы, который использует адекватные модели и обеспечивает исключение ручных способов и снижение трудоемкости и времени преобразования в десятки раз.
4. Разработан метод автоматизированного назначения ключевых полей в заполненных таблицах, который использует адекватные модели и обеспечивает исключение ручных способов и снижение трудоемкости и времени преобразования в десятки раз.
5. Разработана методика автоматизированного формирования реляционных таблиц на основе использования существующей информации табличного вида, в которой задействованы предложенные модели и методы, и которая сводит к минимуму дефекты преобразования, сокращает в десятки раз трудоемкость и время преобразования.

ЛИТЕРАТУРА

1. Агальцов В.П. Базы данных. – М.: Мир, 2002. – 375 с.
2. Аграновский А.В., Арутюнян Р.Э, Хади Р.А. Современные аспекты проблемы поиска в текстовых базах данных//Телекоммуникации. – М., 2003. – №3. – С. 25– 23.
3. Ахаян Р., Горев А., Макатирипов С. Эффективная работа с СУБД. – СПб.: Питер, 1997. – 704 с.
4. Брешенков А.В. О перспективах развития информационных технологий при комплектации машин изделиями автотракторного электрооборудования//Автоэлектрооборудование. – М., 2001. –№1– С. 7– 22.
5. Брешенков А.В. Методы решения задач проектирования реляционных баз данных на основе использования существующей информации табличного вида – М.: Изд-во МГТУ им. Н.Э. Баумана, 2007. – 154 с.
6. Балдин А.В., Брешенков А.В. Анализ проблемы проектирования реляционных баз данных на основе использования существующей информации табличного вида//Вестник Московского государственного технического университета им. Н.Э. Баумана. Серия Приборостроение. – М., 2007. – №2. – С. 66–80.
7. Балдин А.В., Брешенков А.В. Исследование временных свойств системы проектирования реляционных баз данных на основе использования информации табличного вида//Вестник Московского государственного технического университета им. Н.Э. Баумана. Серия Приборостроение. – М., 2007. – №3. – С. 9–23.
8. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем. – СПб.: Питер, 2000. – 384 с.
9. Григорьев Е. А. Представление идентифицируемых сложных объектов в реляционной базе данных//Открытые системы.– М., 2000.–№ 1– 2.

10. Григорьев Ю.А., Ревунков Г.И. Банки данных: Учебник для вузов. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. – 320 с.
11. Дейт К. Дж. Введение в системы баз данных: Пер. с англ. – М.: Наука, 1980. – 464 с.
12. Дейт К. Дж. Введение в системы баз данных. 6-е изд.: Пер. с англ. – Киев: Диалектика, 1998. – 784 с.
13. Дейт К. Дж. Введение в системы баз данных. 7-е изд.: Пер. с англ. – М.: Вильямс, 2001. – 1072 с.
14. Замулин А.В. Системы программирования баз данных и знаний. – Новосибирск: Наука, 1990. – 352 с.
15. Корнеев В.В. и др. Базы данных. Интеллектуальная обработка информации. – М.: Нолидж, 2000. – 162 с.
16. Розмахов О.Г. Основы проектирования баз данных. – М.: Московский авиационный институт, 1993. – 24 с.
17. Chen P. P. – S. The Entity – Relationship Model – Toward Unified View of Data//ACM TODS. – March 1976. – 1, № 1. (Переиздано: M. Stonebraker (ed.) Readings in Database Systems. – San Mateo, Calif.: Morgan Kaufmann, 1988.)
18. Cleaveland J.C. An Introduction to Data Types//Reading, Mass.: Addison-Wesley, 1986.
19. Codd E.F. Data Base Sublanguage Founded on the Relational Calculus//Proc. 1971 ACM SIGFIDET Workshop on data Description, Access and Control. – San Diego, Calif. – November, 1971.
20. Codd E.F. A relational model data for larger shared data banks//Comm; ACM. – 1970. V.13 – № 6 – P. 377– 387.
21. Codd E.F. Further normalization of the database relational model, in data base systems (R. Rustin, ed.). Prentice Hall, Endlewood Cliffs, NJ, 1972.
22. Codd E.F. Recent Investigations into Relational Data Base Eystems//Proc IFIP Congress – Stockholm, Sweden, 1974
23. Codd E.F. The Relational Model For Database Management Version 2.

Reading, Mass.: Addison – Wesley, 1990.

24. Паспорт специальности 05.13.17. Теоретические основы информатики.

25. Информатика: Учебник//Под общ. ред. А.Н. Данчула. – М.: Изд-во РАГС, 2004. – 528 с.

26. Брешиков А.В. Неформальная постановка проблемы преобразования информации табличного вида в файлы баз данных//Сб. трудов АУ МВД России "Актуальные вопросы технологий в деятельности органов внутренних дел". – М., 2004. – С. 55– 70.

27. Брешиков А.В. Избавление от сложных атрибутов в заполненных нереляционных таблицах//Сб. трудов кафедры ИУ-6 – М.: Эликс+, 2006. – С. 10–15.

28. Брешиков А.В. Преобразование заполненных таблиц ко второй нормальной форме//Инженерное образование, 2007. – №2. – 16 с. (Наука и образование: Эл. науч. издание. Номер гос. регистрации 0420700025/0005.)

29. Брешиков А.В. Приведение заполненных таблиц к третьей нормальной форме//Инженерное образование, 2007. – №4. – 15с. (Наука и образование: Эл. науч. издание. Номер гос. регистрации 0420700025/0016.)

30. Брешиков А.В. Разработка модели методики проектирования реляционных баз данных на основе использования информации табличного вида//Вестник Московского государственного технического университета им. Н.Э. Баумана. Серия Приборостроение. – М., 2007. – №2. – С. 40–56.

31. Брешиков А.В. Исследование методики проектирования реляционных баз данных на основе сетевой модели//Вестник Московского государственного технического университета им. Н.Э. Баумана. Серия Приборостроение. – М., 2007. – №3. – С. 55–70.

32. Балдин А.В., Брешиков А.В. Анализ проблемы проектирования реляционных баз данных на основе использования существующей информации табличного вида//Вестник Московского государственного

технического университета им. Н.Э. Баумана. Серия Приборостроение. – М., 2007. – №2. – С. 66–80.

33. Брешенков А.В., Балдин А.В. Исследование временных свойств системы проектирования реляционных баз данных на основе использования информации табличного вида//Вестник Московского государственного технического университета им. Н.Э.Баумана. Серия Приборостроение. – М., 2007. – №3. – С. 9 – 23.

34. Брешенков А.В. Преобразование заполненных таблиц к первой нормальной форме // Инженерное образование, 2007. – №2. – 14 с. (Наука и образование: Эл. науч. издание. Номер гос. регистрации 0420700025/0005.)

35. Брешенков А.В. Приведение заполненных таблиц к четвертой нормальной форме // Инженерное образование, 2007. – №4. – 15с. (Наука и образование: Эл. науч. издание. Номер гос. регистрации 0420700025/0017.)

36. Брешенков А.В., Бараков Д. Д. Вопросы преобразования электронных таблиц в таблицы реляционных баз данных//Современные информационные технологии. Сб. трудов кафедры ИУ–6. – М.: Эликс +, 2004. – С. 44–50.

37. Брешенков А.В. Методология проектирования реляционных баз данных с использованием данных табличного вида. Дис. доктор техн. наук (05.25.05) – М., 2007

38. Хоменко А.Д., Цыганков В.М, Мальцев М.Г. Базы данных: Учебник для высших учебных заведений // Под ред. проф. А.Д. Хомоненко – 6-е изд. – СПб КОРОНА-Век: Бином-Пресс, 2007. – 736 с.

39. Date C. J. Why Quantifier Order Is Important // Date C. J. and Hugh Darwen. Relational Database Writings 1989 – 1991. – Reading, Mass.: Addison-Wesley, 1992.

40. Date C. J. What's Wrong with SQL? // Date C. J. Relational Database Writings 1985 – 1989. – Reading, Mass.: Addison-Wesley, 1990.

41. Date C. J. How We Missed the Relational Boat // Date C. J. Relational Database Writings 1991 – 1994. – Reading, Mass.: Addison-Wesley, 1995.
42. Date C. J. Why Relational? // C.J. Date. Relational Database Writings 1985 – 1989. – Reading, Mass.: Addison-Wesley, 1990.
43. Date C. J.: “There’s Only One Relational Model!”, <http://www.dbdebunk.com> (February 2001).
44. Date C. J. What Not How: The Business Rules Approach to Application Development.-Reading, Mass.: Addison-Wesley, 2000
45. Гэри Хансен, Джэймс Хансен. Базы данных: разработка и управление: Пер. с англ. – М.: Бином, 1999. – 699 с.
46. Ульман Дж. Основы систем баз данных: Пер. с англ. М.Р. Когаловского и В.В. Когутовского.– М.: Финансы и статистика, 1983. – 334 с.
47. Ульман Д., Уидом Д. Введение в системы баз данных: Пер. с англ. – М.: Лори, 2000. – 319 с.
48. Тихомиров Ю.В. Microsoft SQL Server 7.0. – СПб.: БХВ-Петербург, 1999. – 720 с.
49. Тихомиров Ю.В. Microsoft SQL Server 7.0: разработка приложений. – СПб.: БХВ – Петербург, 1999. – 352 с.
50. Аткинсон М., Бансилон Ф., Девитт Д., Дитрих К., Майнер Д., Здоник С. Манифест систем объектно-ориентированных баз данных СУБД. – М., 1995. – №4.
51. Атре Ш. Структурный подход к организации баз данных: Пер. с англ. – М.: Финансы и статистика, 1983. – 317 с.
52. Карпова Т.С. Базы данных: модели, разработка, реализация. – СПб.: Питер, 2001. – 304 с.
53. Брешенков А.В. Базы данных. Проектирование баз данных на основе информации табличного вида. – Germany: LAP LAMBERT Academic Publishing GmbH & Co. KG Dudweiler, rbr, 66123 Saarbrucken, 2011. – 394 с.
54. Бабанов А.М. Теория семантически значимых отображений и ее применение для проектирования реляционных баз данных: Дисс. ... канд.

техн. наук (05.13.11). – М., 2005. – 182 с.

55. Берзтисс А.Т. Структуры данных. – М.: Статистика, 1974. – 408 с.

56. Арсеньев Б.П., Яковлев С.А. Интеграция распределенных баз данных. – СПб.: Лань, 2001. – 461 с.

57. Буре Р. XML и базы данных // Открытые системы. – М., 2000. – № 10. – С. 62– 65.

58. Eisenberg A., Melton J/ SQL: 1999, Formerly Known as SQL3 // ACM SIGMOD Record. – March 1999. – 28, № 4.

59. Брешиков А.В. Выявление и формирование связей один - к одному между заполненными реляционными таблицами: Сборник трудов №5 молодых ученых, аспирантов и студентов "Информатика и системы управления в XXI веке" (часть 2). – М.: МГТУ им. Н.Э. Баумана, 2007. – С. 61– 65.

60. Брешиков А.В. Выявление и формирование связей один-ко многим в заполненных реляционных таблицах//Современные информационные технологии: Сб. трудов кафедры ИУ–6. – М.: Эликс+, 2010. Том 1. – С. 48– 51.

61. Брешиков А.В. Выявление и формирование связей многие-ко многим в заполненных реляционных таблицах//Современные информационные технологии: Сб. трудов кафедры ИУ–6. – М.: Эликс+, 2010. Том 2– С. 60–69.

62. Брешиков А.В., Белоус В.В. Метод назначения первичных ключей в информации табличного вида//Инженерное образование, 2010. – №4. (Наука и образование: Эл. науч. издание. Номер гос. регистрации 0321000195)

63. Кузнецов О.П., Адельсон-Вельский Г.М. Дискретная математика для инженера. – 2-е изд. перераб. и доп.– М.: Энергоатомиздат, 1988. – 480 с.

64. Вентцель Е.С. Теория вероятностей. – М.: Высшая школа, 2001. – 576 с.

65. Колесников А. EXCEL 97 (русифицированная версия). – Киев:

BHV, 1998. – 480 с.

66. Энциклопедия пользователя. Oracle8.: Пер. с англ.//Компания Advanced Information Systems и др. – Киев: ДиаСофт, 1999. – 864 с.

67. Урман С. Oracle 8. Программирование на языке PL/SQL. – М.: Лори, 1999. – 607 с.

68. Тихомиров Ю.В. Microsoft SQL Server 7.0. – СПб.: БХВ-Петербург, 1999. – 720 с.

69. Тихомиров Ю.В. Microsoft SQL Server 7.0: разработка приложений. – СПб.: БХВ – Петербург, 1999. – 352 с.

70. Брешиков А.В. Разработка и исследование метода проектирования регистровых структур в интерактивном режиме: Дисс. ... канд. техн. наук (05.13.12). – М., 1988. – 192 с.

71. Брешиков А.В. Интерактивный анализ регистровых структур // Тез. доклада Всесоюзной школы-семинара "Разработка и применение в народном хозяйстве ЕС ЭВМ /ЕС ЭВМ-85/". – Кишинев, 1985. – С. 17– 19.

72. Брешиков А.В. Структура программного обеспечения диалоговой системы анализа операционных устройств ЭВМ на уровне регистровых передач // Труды МВТУ им. Н.Э. Баумана. – М., 1987. – Вып. 482. – С. 31– 40.

73. Брешиков А.В., Павлов Ю.Е. Интерактивные средства генерации описания моделей регистровых структур // Тез. доклада Всесоюзной научно-технической конференции "Актуальные проблемы информатики, управления и вычислительной техники". – М., 1987. – С. 19– 26.

74. Норенков И.П. Разработка систем автоматизированного проектирования: Учебник для вузов. – М.: Изд-во МГТУ им. Н.Э. Баумана, 1994. – 207 с.

75. Норенков И.П. Основы автоматизированного проектирования: Учебник для вузов. – 2-е изд., переработ. и доп. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. – 336 с.

76. Брешиков А.В., Гудзенко Д.Ю., Казаков Г.И. Проектирование

реляционных баз данных на основе информации табличного типа: Учебное пособие – М.: Изд-во МГТУ им. Н.Э. Баумана, 2009. – 150 с.

77. Система управления базами данных. Руководство по проектированию структур данных. – М.: Информ Икс, 1997. – 14 с.

78. Гудман С., Хидетниemi С. Введение в разработку и анализ алгоритмов. – М.: Мир, 1981. – 368 с.

79. Зиглер К. Методы проектирования программных систем. – М.: Мир, 1985. – 328 с.

80. Иванова Г.С. Технология программирования. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2003. – 320 с.

81. Иванова Г.С., Ничушкина Т.Н. Проектирование программного обеспечения: Методическое пособие по выполнению и оформлению курсовых, дипломных и квалификационных работ. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. – 83 с.

82. Лингер Р., Миллс Х., Уитт Б. Теория и практика структурного программирования. – М.: Мир, 1992. – 406 с.

83. Хьюз Дж., Мичтом Дж. Структурный подход к программированию. – М.: Мир, 1980. – 278 с.

84. Бекаревич Ю.Б., Пушкина Н. В. Microsoft Access 2000. – СПб.: БХВ – Петербург, 2001. – 480 с.

85. Боровиков В.В. Microsoft Access 2002: Программирование и разработка баз данных и приложений. – М.: Солон – Р, 2002. – 560 с.

86. Харитоновна И.А., Михеева В.Д. Microsoft Access 2000. – СПб.: БХВ-Петербург, 2001. – 819 с.

87. Харитоновна И.А., Михеева Л.В. Рудикова. Microsoft Access 2007. – СПб.: БХВ-Петербург, 2008. – 1280 с.

88. Овчинников В.А. Автоматизация комбинаторно-оптимизационных задач при проектировании ЭВМ и систем: Учеб. для вузов. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2001. – 288 с.

89. Формирование связей многие ко многим в заполненных

реляционных таблицах//Вестник МГТУ им. Н.Э. Баумана. Серия Приборостроение. – М., 2011. – С. 105– 117

90. Брешиков А. В., Мин Т. Т. Мотивы разработки метода преобразования информации табличного вида в реляционное представление. // Инженерное образование, 2012. – №3 – 13 с (Наука и образование: Эл. науч. издание. Номер гос. регистрации 0421200025.)

91. Брешиков А. В., Мин Т. Т. Аналитический обзор традиционного подхода формирования реляционных таблиц с учетом использования существующей информации табличного вида // Инженерное образование, 2012. – №8. – 16 с. (Наука и образование: Эл. науч. издание. Номер гос. регистрации 0421200025.)

92. Брешиков А. В., Мин Т. Т. Модели реляционных таблиц и информации табличного вида// Инженерное образование, 2012. – №7. – 10 с. (Наука и образование: Эл. науч. издание. Номер гос. регистрации 0421200025.)

93. Брешиков А. В., Мин Т. Т. Алгоритмы назначения первичных ключей в заполненных таблицах// Инженерное образование, 2012. – №6. – 14 с. (Наука и образование: Эл. науч. издание. Номер гос. регистрации 0421200025.)

94. Брешиков А.В., Мин Т.Т. Преобразование нереляционных таблиц к реляционному виду без использования сложных атрибутов//Вестник Московского государственного технического университета им. Н.Э.Баумана. – М.: МГТУ им. Н.Э.Баумана, 2012. – №2. – С. 59– 60.

95. Брешиков А.В., Мин Тхет Тин. Электронная цифровая подпись. Современные информационные технологии//Сб. трудов кафедры ИУ-6. – М.: НИИ РЛ МГТУ им. Н.Э. Баумана, 2011. – С. 112–115.

96. Брешиков А.В., Мин Тхет Тин. Исключение внутренних подзаголовков и избавление от сложных атрибутов при преобразовании нереляционных таблиц к реляционному виду // Современные информационные технологии: Сб. трудов кафедры ИУ-6. – М.: НИИ РЛ

МГТУ им. Н.Э. Баумана, 2011. – С. 176–183.

97. Брешенков А.В., Мин Т.Т. Вычислительная сложность процедур назначения первичных ключей в заполненных таблицах // Информатика и системы управления в XXI веке: Сб. трудов МГТУ им. Н.Э. Баумана. – М.: МГТУ им. Н.Э. Баумана, 2012.– №9. – С. 136–142.

98. Мин Тхет Тин, Брешенков А.В., Гудзенко Д.Ю. Назначение внешних ключей в заполненных реляционных таблицах//Современные компьютерные системы и технологии: Сб. трудов кафедры ИУ-6. – М.: МГТУ им. Н.Э. Баумана, 2012. – С. 128–135.

99. Мин Тхет Тин, Брешенков А.В., Гудзенко Д.Ю. Анализ типов атрибутов информации табличного вида//Современные компьютерные системы и технологии: Сб. трудов кафедры ИУ-6. – М.: МГТУ им. Н.Э. Баумана, 2012. – С. 15—23.

100. Мин Т. Т. Анализ проблем разработки методики формирования реляционных таблиц на основе использования информации табличного вида. Россия в XXI веке: проблемы, тенденция, перспективы // Материалы XIV Международного симпозиума “Уникальные феномены и универсальные ценности культуры”: Сборник научных статей. – М.: МГТУ им. Н.Э. Баумана, 2012. – С. 270–272.

101. Баранчиков А.И. Методы и модели синтеза информационных структур хранения на основе результатов извлечения закономерностей в актуальных данных предметных областей: Дисс. ... докт. техн. наук (05.13.17). – Рязань, 2014. - 395 с.

ПРИЛОЖЕНИЕ 1.

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ МЕТОДИКИ ФОРМИРОВАНИЯ РТ НА ОСНОВЕ ИТВР

1. Общие принципы разработки комплекса программ преобразования ИТВР в РТР

Так как программное обеспечение (ПО), с одной стороны, реализует закладываемые в него способы и алгоритмы преобразования ИТВР в РТР, а с другой стороны, организует общение разработчика и средств преобразования, эффективность ПО в значительной степени сказывается на эффективности использования средств автоматизированного проектирования [74, 75]. Ввиду того, что взаимодействие разработчика и комплекса программ осуществляется посредством экранных форм, а само взаимодействие ориентировано на обработку ИТВР, разработка ПО тесно связана с разработкой лингвистического и информационного обеспечений.

Так как программы преобразования ИТВР в РТР имеют значительный объемом и сложность, трудоемкость их разработки может составлять несколько человеко-лет, а объем – тысячи операторов языка высокого уровня [77]. С целью сокращения сроков разработки ПО системы интерактивного проектирования РБД на основе ИТВР использовался подход, изложенный в [53]. Основные его положения следующие:

- разработка и исследование комплекса программ и его отдельных компонент таким образом, который обеспечит выявление принципиальных ошибок в нем на ранних этапах его создания, и исключит ситуации, приводящие к необходимости перепроектирования;
- всестороннее использование опыта предыдущих разработок подобных комплексов, а также систем интерактивного взаимодействия как в плане применения эффективных методов создания сложных программных комплексов, так и в плане использования готовых программных модулей;

- проведение анализа требований к разрабатываемому комплексу программ в рамках всех этапов ее разработки, чтобы исключить возможность распространения ошибок проектирования от предыдущих этапов к последующим;

- детальное определение спецификаций подсистемы, унификация лингвистического и информационного обеспечений с целью обеспечения совместимости программных модулей;

- использование методов структурного проектирования и структурного программирования для распараллеливания процесса кодирования программного комплекса и сокращения времени кодирования.

Так как строго детерминированный диалог зачастую сводит на нет основные достоинства интерактивных процедур, разработаны средства, позволяющие изменять число проектных процедур, задаваемых директивами, выполнять переход к любому уровню диалога [74].

Сама структура диалога позволяет пользователю построить непротиворечивую модель поведения системы. Ясности поведения системы также способствуют информационные сообщения, соответствующие директивам диалога [37].

Структура информационных областей и методы доступа к ним в значительной степени определяют производительность средств преобразования ИТВР к РТР. При разработке логического и физического представления данных в информационных областях интерактивных систем особо остро встает проблема разрешения противоречий между необходимостью минимизации времени доступа к информационным областям и минимизацией занимаемого ими объема оперативной памяти. При разработке информационных областей системы эти противоречия были учтены [53].

Как и всякая программная система, интерактивные средства преобразования ИТВР в РТР должна удовлетворять таким требованиям, как

правильность, точность, совместимость, надежность, универсальность, защищенность, полезность, проверяемость и адаптируемость [75].

Исходными данными для разработки программного обеспечения системы являются:

- математическое обеспечение системы (глава 2, глава 3);
- информационные и управляющие кадры терминала.

Общие требования к программному обеспечению сложных программных систем, также как и средства их реализации, широко освещены в литературе, в частности [78, 79-83]. Они всесторонне учитывались при разработке ПО системы.

Анализ существующих средств преобразования информации табличного вида в файлы РТР

Во всех инструментальных СУБД предусмотрены средства импортирования данных. В частности, MS Access позволяет импортировать данные, представленные в форматах MS Access, HTML, MS Excel, текстовом формате и др. [84, 86]. Это существенно упрощает решение проблемы преобразования отдельных таблиц в файлы БД. В случае, если на основе имеющихся таблиц создается новая БД, то необходимо создать БД, импортировать в нее таблицы и, используя средства СУБД, построить необходимые связи между таблицами. Если в существующую БД добавляются новые таблицы, необходимо открыть БД, импортировать в нее таблицы и, используя средства СУБД, построить необходимые связи между таблицами [37].

Однако таблицы, построенные вне СУБД, как правило, нереляционные, а импортирование таблиц такого рода практически невозможно. Тем не менее инструментальные средства существующих СУБД могут быть полезны как при решении задач импорта, так и при преобразования ИТВР в РТ.

В системе Oracle предусмотрено специализированное средство для перемещения данных – SQL*Loader [66]. В нем используются файлы отвергнутых и некорректных записей – записей, которые по каким-либо причинам не могут быть включены в БД. Это, с одной стороны, говорит о том, что проблема преобразования информации табличного вида в файлы РТР актуальна, а с другой стороны, эта проблема не всегда имеет корректное решение.

В Microsoft SQL Server существует служба DTS (Data Transformation Services), предназначенная для преобразования данных. Структура DTS основана на библиотеке Microsoft OLE DB, которая представляет собой набор, предлагающий общий интерфейс различным типам источников данных. DTS в сочетании с OLE DB позволяет установить связь с любым источником данных, для которого имеется драйвер OLE DB или ODBC (Open Data Base Connectivity). [68, 69] Это – набор соглашений, которые позволяют получить доступ к информации из баз данных В SQL Server 2000 предусмотрены драйверы OLE DB для SQL Server, Oracle, Excel, Access, текстовых файлов ASCII-файлов и других источников данных, поддерживающих ODBC.

Однако и здесь есть недостатки. Служба DTS работает с нормализованными данными, поэтому требуются предварительные работы по подготовке данных с помощью средств SQL Server, но они не всегда приводят к положительному результату [68, 69].

Подводя итоги, можно сделать вывод, что средства импорта СУБД, а также другие механизмы обмена данными до определенной степени обеспечивают преобразование ИТВР в РТР, однако их зачастую бывает недостаточно.

Разрабатываемый комплекс программ предназначен для круга пользователей средней квалификации, с начальными знаниями в области БД. В связи с этим в качестве инструментальной СУБД оправдано выбрать доступную систему, обладающую малой субъективной сложностью и

удовлетворяющую необходимым требованиям. К такой инструментальной СУБД относится Microsoft Access.

MS Access включает в себя интегрированную среду объектно-ориентированного программирования Visual Basic for Applications (VBA), позволяющую реализовать любые программные решения.

В VBA БД рассматривается как совокупность объектов (таблиц, форм, отчетов, их элементов и т.д.). Эти объекты имеют свойства и методы, которые реализуют действия над объектами.

Среда VBA – одна из лучших сред систем программирования. Она объединяет наглядные инструменты: редактор VBA, окно разрабатываемого проекта, окно свойств объектов проекта, окно просмотра объектов, отладчик и др. Все инструменты унифицированы и являются общими для всех продуктов Microsoft Office [84, 86].

2. Программная реализация методики назначения ключевых полей

Может вызвать недоумение, что программная реализация назначения ключевых полей обсуждается до программной реализации преобразования ИТВР в РТР. Таким образом, специально подчеркивается, что методы по сути независимы и самодостаточны. Дело в том, что ключевые поля могут быть не назначены в таблицах существующих БД. И очень вероятно, что возникнет необходимость их назначить. В этом случае процедуры назначения ключевых полей могут оказаться полезными вне зависимости от процедур преобразования ИТВР в РТР.

2.1. Назначение первичных ключей на базе одного атрибута

Постановка задачи звучит предельно просто – необходимо в таблице найти домены, значения атрибутов которых уникальны.

В первую очередь напрашивается визуальный анализ всех столбцов таблицы. И это оправдано для таблиц с небольшими степенями и мощностями. Например, степень таблицы 10 столбцов, а ее мощность 50

строк. Опыт показывает, что таблицы большей размерности проанализировать “вручную” без ошибок очень сложно.

Во вторую очередь естественно попытаться использовать существующие инструментальные средства. Рассмотрим такие средства. В MS Access предусмотрен мастер просмотра повторяющихся записей. Он позволяет выбрать любое сочетание атрибутов таблицы и просмотреть, сколько значений связанных с выбранным атрибутом повторяются. Ниже на рис. 1 - 6 приведены шаги мастера для анализа двух полей “Имя” и “Отчество” таблицы “Сотрудники”.

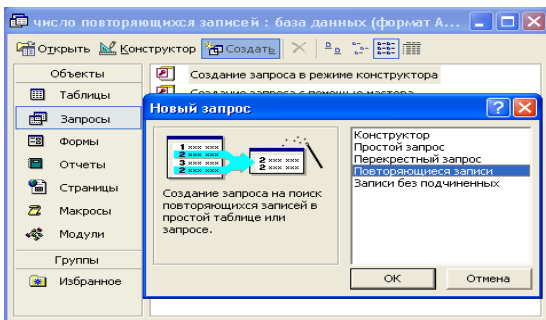


Рис. 1. Первый шаг мастера

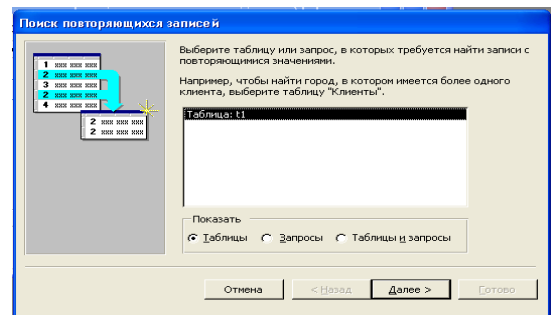


Рис. 2. Второй шаг мастера

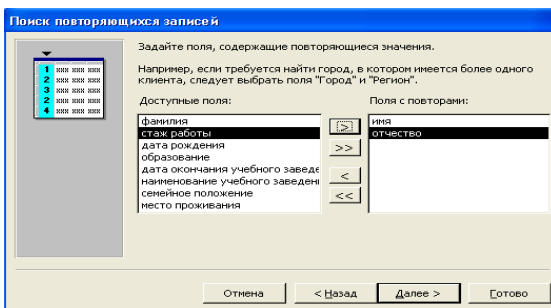


Рис. 3. Третий шаг мастера

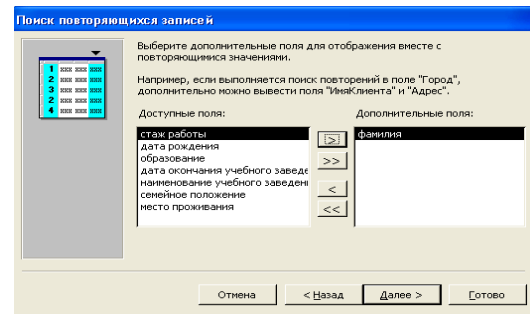


Рис. 4. Четвертый шаг мастера

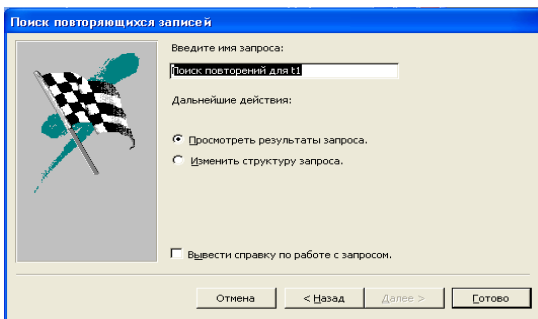


Рис. 5. Пятый шаг мастера

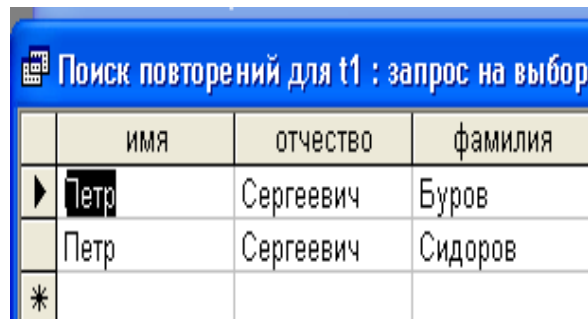


Рис. 6. Шестой шаг мастера

Первый шаг мастера позволяет выбрать мастер, второй – выбрать анализируемую таблицу, третий – выбрать атрибуты для анализа, четвертый

– выбрать отображаемые поля, пятый – указать имя запроса для отображения повторяющихся полей, шестой – выводит повторяющиеся поля. В нашем случае значения полей “Имя” и “Отчество” повторяются два раза. Таким образом, эти атрибуты не могут использоваться в качестве первичного ключа.

Следует обратить внимание на то, что эти 6 шагов нужно выполнять для каждого сочетания полей. В случае анализа одного атрибута для 30 столбцов манипуляции нужно выполнить 30 раз. Но для такой же таблицы при анализе двух атрибутов манипуляции придется проделать 435 раз, при анализе трех атрибутов – 4060 раз, при анализе четырех атрибутов – 27405 раз. Это связано с числом возможных сочетаний атрибутов.

В конечном случае все шаги мастера приводят к формированию следующего запроса на языке SQL.

```
SELECT [имя] FROM [Сотрудники] As Tmp GROUP BY [имя],[отчество] HAVING Cnt(*)>1 And [отчество] = [t1].[отчество])
```

Но написать несколько сотен запросов для каждой таблицы задача непростая.

В связи с этим реально использование существующих средств только при анализе одного атрибута. Конечно, удобнее получить исчерпывающую информацию за несколько секунд, чему и посвящены разработанные процедуры.

На рис. 7 приведен фрагмент таблицы, на которой будут продемонстрированы разработанные процедуры.

фамилия	имя	отчество	стаж работы	дата рождения	образование	дата окончания учебного заведения	наименование учебного
Агеев	Александр	Семенович	11	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана
Андреев	Алексей	Александрович	11	01.01.1980	высшее	07.07.1993	МГТУ им. Н.Э. Баумана
Акылов	Антон	Алтуфьевич	10	01.01.1980	высшее	07.07.1993	МГТУ им. Н.Э. Баумана
Брыкалов	Афиноген	Андреевич	10	01.01.1980	высшее	07.07.1993	МГТУ им. Н.Э. Баумана
Борисов	Борис	Алексеевич	9	01.01.1980	высшее	07.07.1993	МГТУ им. Н.Э. Баумана
Брейдо	Виктор	Абрамович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана
Витин	Иван	Антипович	11	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана
Волгин	Петр	Борисович	9	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана
Вьдрин	Сидор	Викторович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана
Гусев	Владимир	Власович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана

Рис .7. Фрагмент анализируемой таблицы

В целях обеспечения возможности обработки таблицы и использования в процедурах типовых имен необходимо выполнить запрос на добавление (рис. 8).

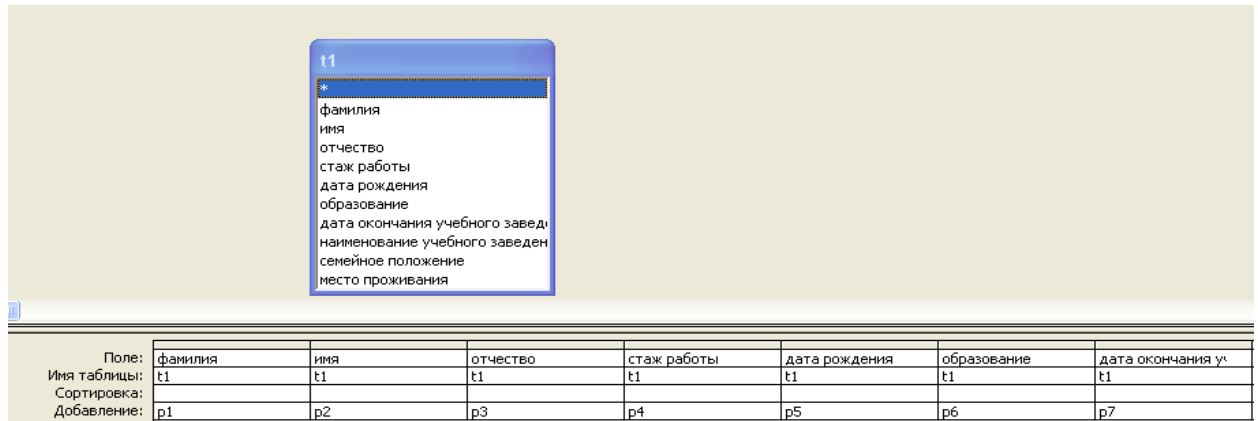


Рис. 8. Запрос на добавление исходной таблицы

В формате SQL запрос выглядит следующим образом:

```
INSERT INTO t2 ( p1, p2, p3, p4, p5, p6, p7, p8, p9, p10 )
```

```
SELECT t1.фамилия, t1.имя, t1.отчество, t1.[стаж работы], t1.[дата рождения],  
t1.образование, t1.[дата окончания учебного заведения], t1.[наименование  
учебного заведения], t1.[семейное положение], t1.[место проживания]
```

```
FROM t1;
```

В результате выполнения запроса на добавлении сформируется рабочая таблица (рис. 9)

r1	r2	r3	r4	r5	r6	r7	r8
Агеев	Александр	Семенович	11	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баум.
Андреев	Алексей	Александрович	11	01.01.1980	высшее	07.07.1993	МГТУ им. Н.Э. Баум.
Акылов	Антон	Алтуфьевич	10	01.01.1980	высшее	07.07.1993	МГТУ им. Н.Э. Баум.
Брыкалов	Афиноген	Андреевич	10	01.01.1980	высшее	07.07.1993	МГТУ им. Н.Э. Баум.
Борисов	Борис	Алексеевич	9	01.01.1980	высшее	07.07.1993	МГТУ им. Н.Э. Баум.
Брейдо	Виктор	Абрамович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баум.
Витин	Иван	Антипович	11	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баум.
Волгин	Петр	Борисович	9	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баум.
Вьядрин	Сидор	Викторович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баум.
Гусев	Владимир	Власович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баум.

Рис. 9. Рабочая таблица

После этого пользователю остается только воспользоваться интерфейсом представленном на рис. 10.



Рис. 10. Интерфейс пользователя для поиска первичных ключей

Ниже приведен фрагмент процедуры, обеспечивающей поиск первичного ключа на базе одного атрибута. Команды для удобства пояснений пронумерованы.

1. Private Sub Кнопка0_Click()
2. Dim mdb As Database ' mdb объявляется как база данных
3. Dim rst As Recordset ' rst объявляется как массив записей
4. Dim a(10) As Variant ' Массив заголовков исходной таблицы
5. Dim kp As String
6. Dim np As String
7. Set mdb = CurrentDb ' mdb назначается текущая база данных
8. Set rst = mdb.OpenRecordset("t1") ' rst назначается массив записей из таблицы t1
9. DoCmd.OpenForm "t1", acNormal, "", "", acEdit, acNormal ' открывается форма t1 для таблицы t1
10. DoCmd.GoToRecord , , acFirst ' переход на первую запись формы
11. kp = ""
12. np = ""
13. n = 0 ' счетчик имен столбцов исходной таблицы
14. For Each fld In rst.Fields ' в цикле перебираются все имена таблицы для их запоминания
15. ' MsgBox fld.Name
16. n = n + 1
17. a(n) = fld.Name
18. END fld
19. ' DoCmd.Close ' закрывается форма t1
20. DoCmd.OpenForm "t2", acNormal, "", "", acEdit, acNormal
21. DoCmd.GoToRecord acForm, "t2", acLast
22. DoCmd.GoToRecord , , acFirst
23. c = Forms!t2!cpol ' запоминается число полей
24. f = 0
25. For I = 1 To c - 1


```

26.p11 = Forms!t2!p1
27.MsgBox (p11)
28.For j = I + 1 To c
29.DoCmd.GoToRecord , , acEND
30.p22 = Forms!t2!p1
31.MsgBox (p22)
32.If p11 = p22 Then
33.f = 1
34.If f = 1 Then
35.' MsgBox a(1)
36.MsgBox ("столбец не ключевой: " & a(1))
37.np = np & ", " & a(1)
38.End If
39.Exit For
40.End If
41.END j
42.If f = 1 Then Exit For
43.DoCmd.GoToRecord , , acFirst
44.For m = 1 To I
45.DoCmd.GoToRecord , , acEND
46.END m
47.If f = 1 Then
48.' MsgBox a(1)
49.MsgBox ("столбец не ключевой: " & a(1))
50.np = np & ", " & a(1)
51.End If
52.END I
53.If f = 0 Then
54.' MsgBox a(1)
55.MsgBox ("столбец ключевой: " & a(1))
56.kp = kp & ", " & a(1)
57.End If
58.MsgBox (" НЕ КЛЮЧЕВЫЕ ПОЛЯ: " & np & " КЛЮЧЕВЫЕ ПОЛЯ: "
& kp)
59.MsgBox ("ПОЛЯ, ПРЕТЕНДУЮЩИЕ НА ПЕРВИЧНЫЙ КЛЮЧ: " &
kp)
60.DoCmd.Close acForm, "t1"
61.End Sub

```

Дадим краткое пояснение основной компоненты при процедуре назначения первичных ключей на базе одного атрибута.

В командах 5, 6 объявляются строковые переменные, в которых накапливаются сообщения о ключевых и неключевых полях. В Заголовков

таблицы запоминаются заголовки таблицы. Они впоследствии потребуются при формировании сообщений о первичных ключах.

Посредством оператора 20 открывается рабочая форма. С помощью операторов 21, 22 она “перетряхивается”.

В форме “t2” имеется поле “сrol”, где подсчитывается число записей таблицы. Это поле используется для организации цикла просмотра всех записей, а в их рамках – полей, соответствующих анализируемому столбцу.

В 24 операторе обнуляется флажок, который фиксирует отсутствие ключевых полей.

Оператор 25 организует цикл для сравнения каждого значения атрибута со всеми остальными. Все остальные значения перебираются посредством цикла, который начинается 28-м оператором.

В операторах 26 и 29 значениям переменных присваиваются значения полей таблицы, которые сравниваются в операторе 32.

С помощью оператора 37 накапливаются имена неключевых атрибутов. Операторы 43 – 46 позволяют перейти (вернуться) к следующей сравниваемой записи, ведь в цикле по “j” мы достигаем конца таблицы.

Оператор 59 формирует итоговое сообщение со списком полей, которые могут претендовать на ключевые поля.

Кратко рассмотрим пример функционирования полного комплекта процедур выявления первичного ключа на базе одного атрибута после того, как пользователь нажмет соответствующую кнопку (рис. 10).

На рис. 11 приведен пример одного из текущих сообщений процедуры.

p1	p2	p3	p4
Брыкалов	Афиноген	Андреевич	10
Борисов	Борис	Алексеевич	9
Брейдо	Виктор	Абрамович	10
Витин		ич	11
Волгин		ич	9
Вьдрин		вич	10
Гусев		ч	10
Гуров	Григорий	Гордеевич	10
Демин	Геннадий	Григорьевич	9

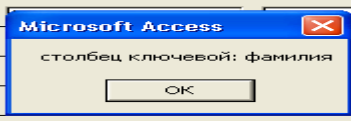


Рис. 11. Пример одного из текущих сообщений процедуры.

На рис. 12 приведен пример другого типа текущих сообщений процедуры.

Витин	Иван	Антипович
Волгин	Петр	Борисович
Вьдрин		Викторович
Гусев		Власович
Гуров		Гордеевич
Демич		Григорьевич
Долгов	Демьян	Гурович
Зинин	Данила	Дмитриевич
Ершов	Кирилл	Денисович
Гасов	Борис	Петрович
Санаев	Иван	Иванович

Рис. 12. Пример другого типа текущих сообщений процедуры.

На рис. 13 приведено итоговое сообщение процедуры о полях таблицы, которые могут претендовать на роль первичного ключа.

фамилия	имя	отчество	стаж работы	дата рождения
Агеев	Александр	Семенович	11	02.02.1978
Андреев	Алексей	Александрович	11	01.01.1980
Акылов				01.01.1980
Брыкалов				01.01.1980
Борисов				01.01.1980
Брейдо	Виктор	Абрамович	10	02.02.1978
Витин	Иван	Антипович	11	02.02.1978

Рис. 13. Итоговое сообщение процедуру о полях таблицы, которые могут претендовать на роль первичного ключа.

2.2 Назначение первичных ключей на базе двух атрибутов

Ниже приведен фрагмент процедуры, обеспечивающей поиск первичного ключа на базе двух атрибутов. Команды для удобства пояснений пронумерованы.

1. Private Sub Кнопка2_Click()
2. Dim mdb As Database ' mdb объявляется как база данных
3. Dim rst As Recordset ' rst объявляется как массив записей
4. Dim a(10) As Variant ' Массив заголовков исходной таблицы
5. Dim kp As String
6. Dim np As String
7. Set mdb = CurrentDb ' mdb назначается текущая база данных
8. Set rst = mdb.OpenRecordset("t1") ' rst назначается массив записей из таблицы t1
9. DoCmd.OpenForm "t1", acNormal, "", "", acEdit, acNormal ' открывается форма t1 для таблицы t1
10. DoCmd.GoToRecord , , acFirst ' переход на первую запись формы
11. n = 0 ' счетчик имен столбцов исходной таблицы
12. For Each fld In rst.Fields ' в цикле перебираются все имена таблицы для их запоминания
13. ' MsgBox fld.Name
14. n = n + 1

```

15.a(n) = fld.Name
16.END fld
17.' DoCmd.Close ' закрывается форма t1
18.DoCmd.OpenForm "t2", acNormal, "", "", acEdit, acNormal
19.DoCmd.GoToRecord acForm, "t2", acLast ' перетряхивание
20.DoCmd.GoToRecord , , acFirst ' формы
21.c = Forms!t2!crol ' запоминается число полей
22.kp = ""
23.np = ""
24.f = 0
25.For I = 1 To c - 1
26.p11 = Forms!t2!p1 & " " & Forms!t2!p2
27.MsgBox (p11)
28.For j = I + 1 To c
29.DoCmd.GoToRecord , , acEND
30.p22 = Forms!t2!p1 & " " & Forms!t2!p2
31.MsgBox (p22)
32.If p11 = p22 Then
33.f = 1
34.If f = 1 Then
35.MsgBox a(1) & " " & a(2)
36.MsgBox ("столбцы не могут составить ключевой столбец: " & a(1)) & " "
    & a(2)
37.np = np & a(1) & " и " & a(2) & ", "
38.End If
39.Exit For
40.End If
41.END j
42.If f = 1 Then Exit For
43.DoCmd.GoToRecord , , acFirst
44.For m = 1 To I
45.DoCmd.GoToRecord , , acEND
46.END m
47.If f = 1 Then
48.MsgBox a(1) & " " & a(2)
49.MsgBox ("столбцы не могут составить ключевой столбец: " & a(1)) & " "
    & a(2)
50.np = np & a(1) & " и " & a(2) & ", "
51.End If
52.END I
53.If f = 0 Then
54.MsgBox a(1) & " " & a(2)
55.MsgBox ("столбцы могут составить ключевой столбец: " & a(1)) & " " &
    a(2)

```

```

56.kp = kp & a(1) & " и " & a(2) & ", "
57.End If
58.DoCmd.Close acForm, "t2"
59.MsgBox ("ПОЛЯ, ПРЕТЕНДУЮЩИЕ НА ПЕРВИЧНЫЙ КЛЮЧ: " & kp)
60.End Sub

```

По структуре и составу операторов этот фрагмент похож на предыдущий фрагмент. Принципиальным отличием является то, что анализируется пара полей, а не одно поле. Это отражено в операторах 26, 30, 35– 37, 48– 50, 54, 56.

Промежуточные сообщения могут быть полезны, но в принципе их можно убрать. При проверке конкатенации нескольких атрибутов на принадлежность первичному ключу их может быть очень много – сочетание атрибутов велико.

На рис. 14 приведено итоговое сообщение процедуры о парах полей таблицы, которые могут претендовать на роль первичного ключа. Промежуточные сообщения в данном случае опущены.

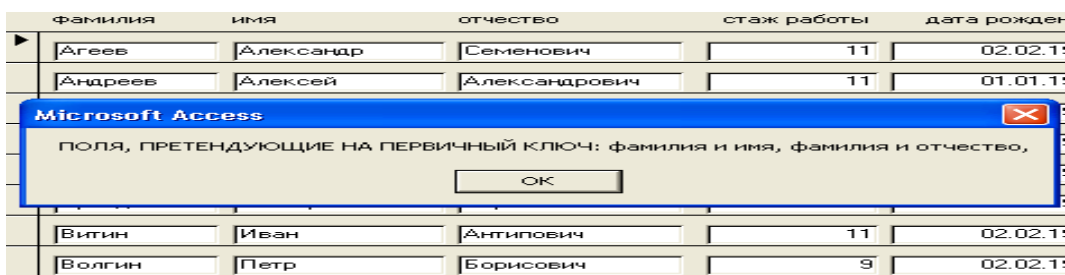


Рис. 14. Итоговое сообщение процедуру о парах полей таблицы

2.3. Назначение первичных ключей на базе трех атрибутов

Ниже приведен фрагмент процедуры, обеспечивающей поиск первичного ключа на базе трех атрибутов. Команды для удобства пояснений пронумерованы.

1. Private Sub Кнопка5_Click()
2. Dim mdb As Database ' mdb объявляется как база данных
3. Dim rst As Recordset ' rst объявляется как массив записей
4. Dim a(10) As Variant ' Массив заголовков исходной таблицы
5. Dim kp As String
6. Dim np As String

```

7. Set mdb = CurrentDb ' mdb назначается текущая база данных
8. Set rst = mdb.OpenRecordset("t1") ' rst назначается массив записей из
   таблицы t1
9. DoCmd.OpenForm "t1", acNormal, "", "", acEdit, acNormal ' открывается
   форма t1 для таблицы t1
10.DoCmd.GoToRecord , , acFirst ' переход на первую запись формы
11.n = 0 ' счетчик имен столбцов исходной таблицы
12.For Each fld In rst.Fields ' в цикле перебираются все имена таблицы для
   их запоминания
13.' MsgBox fld.Name
14.n = n + 1
15.a(n) = fld.Name
16.END fld
17.' DoCmd.Close ' закрывается форма t1

18.DoCmd.OpenForm "t2", acNormal, "", "", acEdit, acNormal
19.DoCmd.GoToRecord acForm, "t2", acLast ' перетряхивание
20.DoCmd.GoToRecord , , acFirst ' формы
21.c = Forms!t2!cpol ' запоминается число полей
22.kp = ""
23.np = ""
24.f = 0
25.For I = 1 To c - 1
26.p11 = Forms!t2!p1 & " " & Forms!t2!p2 & " " & Forms!t2!p3
27.'MsgBox (p11)
28.For j = I + 1 To c
29.DoCmd.GoToRecord , , acEND
30.p22 = Forms!t2!p1 & " " & Forms!t2!p2 & " " & Forms!t2!p3
31.'MsgBox (p22)
32.If p11 = p22 Then
33.f = 1
34.If f = 1 Then
35.'MsgBox a(1) & " " & a(2)& " " & a(3)
36.MsgBox ("столбцы не могут составить ключевой столбец: " & a(1)) & "
   " & a(2) & " " & a(3)
37.np = np & a(1) & " " & a(2) & " " & " " & a(3) & ", "
38.End If
39.Exit For
40.End If
41.END j
42.If f = 1 Then Exit For
43.DoCmd.GoToRecord , , acFirst
44.For m = 1 To I
45.DoCmd.GoToRecord , , acEND

```

```

46.END m
47.If f = 1 Then
48.'MsgBox a(1) & " " & a(2) & " " & a(3)
49.MsgBox ("столбцы не могут составить ключевой столбец: " & a(1)) & "
    " & a(2) & " " & a(3)
50.np = np & a(1) & " " & a(2) & " " & " " & a(3) & ", "
51.End If
52.END I
53.If f = 0 Then
54.'MsgBox a(1) & " " & a(2) & " " & a(3)
55.MsgBox ("столбцы могут составить ключевой столбец: " & a(1)) & "
    " & a(2) & " " & a(3)
56.kp = kp & a(1) & " " & a(2) & " " & " " & a(3) & ", "
57.End If
58.DoCmd.Close acForm, "t2"
59.'MsgBox (" НЕ КЛЮЧЕВЫЕ ПОЛЯ: " & np & " КЛЮЧЕВЫЕ ПОЛЯ: "
    & kp)
60.MsgBox ("ПОЛЯ, ПРЕТЕНДУЮЩИЕ НА ПЕРВИЧНЫЙ КЛЮЧ: " &
    kp)
61.DoCmd.Close acForm, "t1"
62.End Sub

```

По структуре и составу операторов этот фрагмент похож на предыдущий фрагмент. Принципиальным отличием является то, что анализируется пара полей, а не одно поле. Это отражено в операторах 26, 30, 35– 37, 48– 50, 54, 56.

На рис. 15 приведено одно из промежуточных сообщений процедуры.

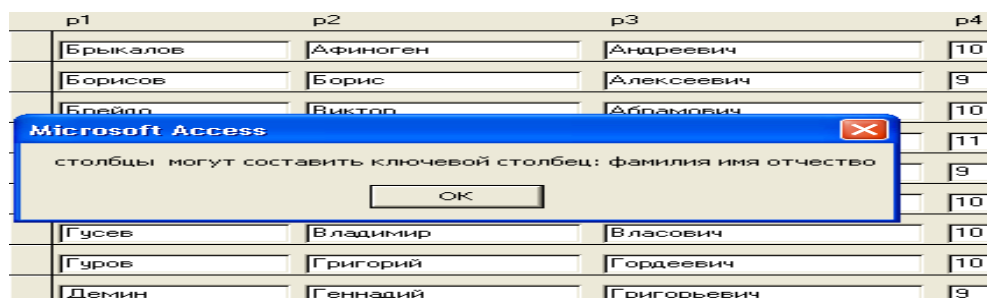


Рис .15. Одно из промежуточных сообщений процедуры.

На рис. 16 приведено итоговое сообщение процедуры о тройках полей таблицы, которые могут претендовать на роль первичного ключа.

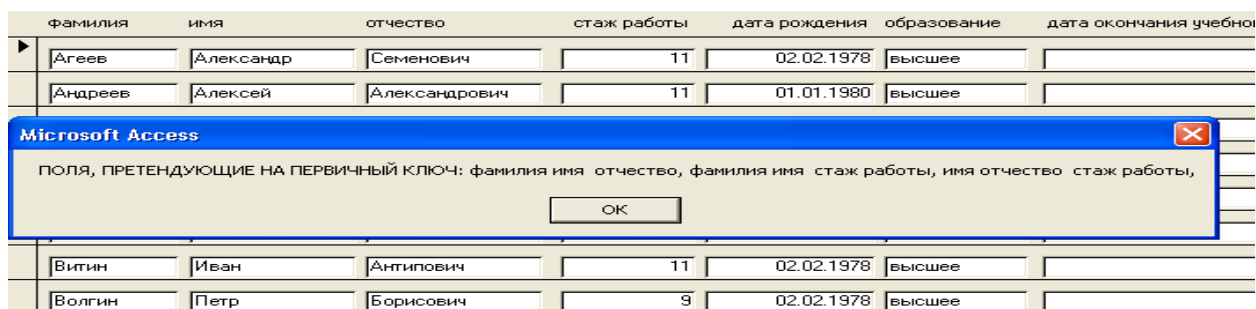


Рис. 16. Итоговое сообщение процедуры о тройках полей таблицы, которые могут претендовать на роль первичного ключа.

2.4. Оценка вычислительной сложности процедур назначения первичных ключей

Выполним аналитическую оценку вычислительной сложности разработанных программ.

В качестве базовой для аналитической оценки вычислительной сложности процедур операции выбрана операция сравнения. Во-первых, эта операция одна из самых длительных, а во-вторых, она встречается внутри всех циклов. Остальные компоненты программы добавляют в общее время выполнения программы незначительные значения.

Важной компонентой оценки является число сочетаний. Ведь для анализируемой таблицы необходимо проверить все сочетания полей.

Число возможных сочетаний $C_n^k = n!/(n-k)!/k!$, где n – общее число атрибутов таблицы, а k – количество проверяемых атрибутов на принадлежность к первичному ключу.

Для предварительного анализа написана небольшая процедура, которая позволила подсчитать количества сочетания для различных n и k . Результаты сведены в таблицы. Таблица 1– число сочетаний из 1– 10 по 1, 2, 3, 4. Таблица 2 – число сочетаний из 11– 20 по 1, 2, 3, 4. Таблица 3 - Число сочетаний из 21– 30 по 1, 2, 3, 4.

Таблица 1

	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	0	1	3	6	10	15	21	28	36	45
3	0	0	1	4	10	20	35	56	84	120
4	0	0	0	1	5	15	35	70	126	210

Таблица 2

	11	12	13	14	15	16	17	18	19	20
1	11	12	13	14	15	15	17	18	19	20
2	55	66	78	91	105	120	136	153	171	190
3	165	220	286	364	455	560	680	816	969	1140

Таблица 3

	21	22	23	24	25	26	27	28	29	30
1	21	22	23	24	25	26	27	28	29	30
2	210	231	253	276	300	325	351	378	406	435
3	1330	1540	1771	2024	2300	2600	2925	3276	3654	4060
4	5985	7315	8855	10626	12650	14950	17550	20475	23751	27405

Важно отметить, что сравнивать необходимо значения полей всех записей таблицы, при этом необходимо сравнивать поля каждой записи со всеми остальными. Поэтому к числу сочетаний добавляется сомножитель. Определим его эмпирическим методом.

Например, возьмем таблицу из одиннадцати записей. Тогда для записи 1-й потребуется 10 проверок, для 2-й – 9, для 3-й – 8, для 4-й – 7, для 5-й – 6, для 6-й – 5, для 7-й – 4, для 8-й – 3, для 9-й – 2, для 10-й – 1. Это следует из фрагмента программы – каждая текущая запись сравнивается со всеми остальными. И с увеличением номера текущей записи уменьшается число записей, с которыми ее необходимо сравнить. Таким образом, общее количество сравнений будет $10+9+8+7+6+5+4+3+2+1 = 10+1 + 9+2 + 8+3 + 7+4 + 6+5 = 55$. Итого мы получили 5 пар или $(11-1)/2$ или $(m-1)/2$, где m – число записей таблицы. В каждой паре сумма равна 11 или m . Таким образом, число сравнений будет равным $m(m-1)/2$. Получено аналитическое выражение для оценки временной сложности алгоритма: $VS = m(m-1)/2n!/(n-k)!/k!$, где

n – общее число атрибутов таблицы;

k – количество проверяемых атрибутов на принадлежность к первичному ключу;

m – число записей таблицы.

Оценка VS впечатляет и внушает пессимизм. Для дальнейшего анализа вычислительной сложности программ построены графики (рис. 17 и рис. 18).

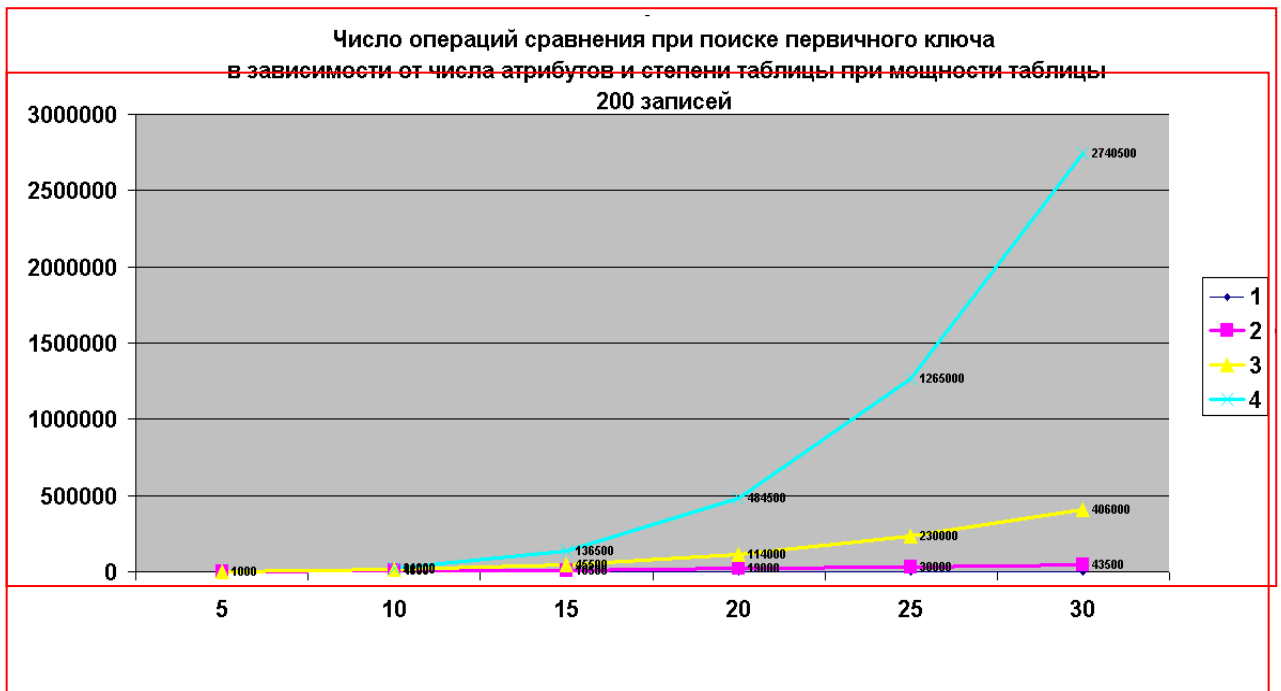


Рис.17. График изменения числа сравнений для таблицы из 200 записей.

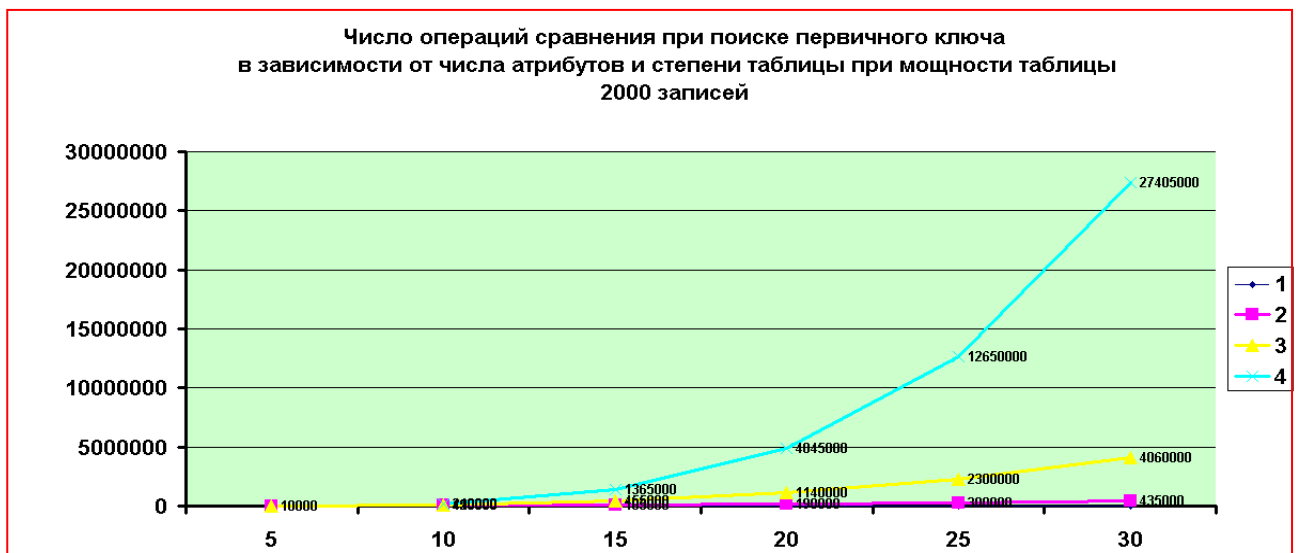


Рис.18. График изменения числа сравнений для таблицы из 2000 записей.

Самое большое число, которое можно увидеть на графиках это 27.450.000. В качестве утешения можно сказать, что такая ситуация встречается исключительно редко. Кроме того, специальный эксперимент показал, что 100.000.000 сравнений на среднем компьютере (1,6 ГГц, 2 ядра) выполняется за 7 секунд. В качестве доказательства приводится код программы и результат ее выполнения (рис. 5.19).

```
Private Sub Кнопка4_Click()
Forms!даты!дата1 = Time
For i = 1 To 100000000
    If i = 1000000 Then r = 0
END i
Forms!даты!дата2 = Time
End Sub
```

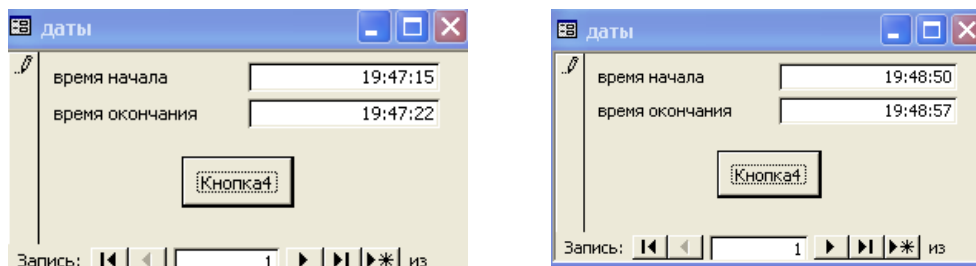


Рис.19. Результаты измерения времени 100.000.000 сравнений

Выполним экспериментальную оценку вычислительной сложности разработанных программ.

Посредством экспериментов выполнены оценки времен выполнения процедур обработки 2000 записей при количестве анализируемых атрибутов 1, 2, 3, 4 и общем числе атрибутов 5, 10, 15, 20, 25, 30, которые сведены в таблицу 4.

Таблица 4 взята за основу для построения, графиков, приведенных на рис. 18.



Рис.18. Графики времени выполнения процедур в зависимости от числа столбцов и числа анализируемых атрибутов.

Как видно из графика, времена вполне приемлемы. При этом

	5	10	15	20	25	30
1	5000	10000	15000	20000	25000	30000
2	10000	45000	105000	190000	300000	435000
3	100000	120000	455000	1140000	2300000	4060000
4	5000	210000	1365000	4845000	12650000	27405000

характеристики таблиц и числа атрибутов, как показывает опыт разработок, исключительно редко превышают рассмотренные значения.

Однако бывают и исключения. В связи с этим далее анализируются границы применимости предлагаемого способа и соответствующих процедур.

Подсчитаны числа сочетаний для 100 столбцов таблицы:

Из 100 по 2 – 4950

Из 100 по 3 – 161600

Из 100 по 4 – 3921225

Число необходимых итераций для анализа 2000 записей равно $1999 \cdot 2000 / 2 = 1999000$

Число необходимых операций сравнений для 100 столбцов и 3-х анализируемых атрибутов для таблицы в 2000 записей равно:

$$161600 * 1999000 = 9.905.045.000$$

Экспериментальная проверка показала, что для анализа таблицы из 100 столбцов и 2000 записей проверка всех сочетаний по 3-и атрибута, претендующих в своем составе на первичный ключ, необходимо 4 мин. 45 сек. Это вполне приемлемое время, учитывая то, что выполнить миллиарды сравнений вручную просто невозможно.

Для анализа таблицы из 100 столбцов и 2000 записей проверка всех сочетаний по 4-и атрибута, претендующих в своем составе на первичный ключ, необходимо 153 мин. 15 сек. Такая ситуация скорее всего гипотетическая. В основном из-за того, что для назначения первичного ключа очень редко используют 4-е атрибута. Но, как известно, в целях решения важных задач зачастую идут и на такие затраты.

Таким образом, можно сделать вывод, что предложенный способ и соответствующие процедуры могут успешно использоваться для анализа таблиц из несколько десятков столбцов, нескольких тысяч записей и одного, двух, трех атрибутов, используемых в качестве ключевых.

3. Программная реализация метода преобразования ИТВР в РТР

3.1 Анализ типов атрибутов таблицы

В ИТВР типы элементов в пределах любого столбца таблицы могут различаться. Такие столбцы недопустимо использовать в РТР БД. Ниже описаны программные средства, которые позволяют разработчику выявить наиболее вероятный тип для каждого столбца ИТВР и принять правильное решение.

Все столбцы ИТВР имеют тип строковый. Поэтому задача состоит в том, чтобы проанализировать в каждом столбце все его элементы и по

внешнему виду элементов определить его тип. Предложен следующий подход.

Определяется не тип элементов, а наоборот непринадлежность к типу. Формируются данные о том, сколько и какие элементы столбца не соответствуют каждому из базовых типов. После этого нетрудно сделать вывод о том, какого типа столбец. Могут создаваться неочевидные ситуации. Тогда решение принимает разработчик, после чего вид элементов он приводит в соответствии с выбранным типом, а затем назначает выбранный тип в целевой таблице. При этом рассматриваются базовые типы: дата-время, числовой и строковый.

Сначала кратко рассмотрим, что сделано по этому поводу, а потом – как это сделано.

На рис. 20 приведен фрагмент интерфейса для анализа типов ИТВР.



Рис. 20. Фрагмент интерфейса для анализа типов ИТВР.

После нажатия на соответствующую кнопку сформируется сообщение о том, сколько полей не являются датами для анализируемого столбца. В данном случае это “фамилия” (рис. 21).

Антипович	11	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	холост	Бэ
Борисович	9	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат	Ми
Викторович	10	02			м. Н.Э. Баумана	холост	Ми
Петрович	11	02			м. Н.Э. Баумана	холост	Ми
Гордеевич	10	01			м. Н.Э. Баумана	холост	Бэ
Власович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат	Ми

Рис. 21. Сообщение о том, сколько полей не являются датами.

Затем сформируется сообщение о том, в каких записях фамилия не является датой. Это показано на рис. 22.

Антипович	11	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	холост	Б
Борисович	9	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат	М
Викторович						холост	М
Петрович						холост	М
Гордеевич						холост	Б
Власович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат	М

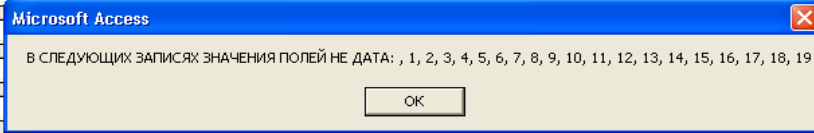


Рис. 22. Сообщение о том, в каких записях фамилия не является датой.

Далее сформируется сообщение о том, сколько полей не являются числами для анализируемого столбца (рис. 23).

Борисович	9	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат	
Викторович	10				Н.Э. Баумана	холост	
Петрович	11				Н.Э. Баумана	холост	
Гордеевич	10				Н.Э. Баумана	холост	
Власович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат	

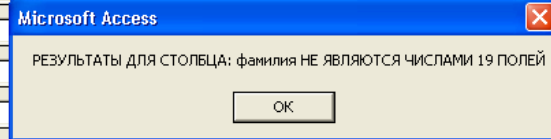


Рис. 23. Сообщение о том, сколько полей не являются числами для анализируемого столбца.

Затем сформируется сообщение о том, в каких записях фамилия не является датой. Это показано на рис. 24.

Антипович	11	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	холост	
Борисович	9	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат	
Викторович						холост	
Петрович						холост	
Гордеевич						холост	
Власович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат	

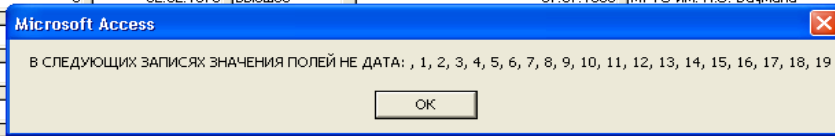


Рис. 24. Сообщение о том, в каких записях фамилия не является датой.

Учитывая то, что в таблице всего 19 полей, и все они не являются ни датами ни числами, легко сделать вывод о том, что столбец “фамилия” строкового типа.

Рассмотрим противоположный случай. Анализируется 5-й столбец с датами. В процессе анализа этого столбца сформируется сообщение о том, сколько полей не являются датами для анализируемого столбца (рис. 25).

Антипович	11	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	холост	
Борисович	9	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат	
Викторович	10				Н.Э. Баумана	холост	
Петрович	11				Н.Э. Баумана	холост	
Гордеевич	10				Н.Э. Баумана	холост	
Власович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат	

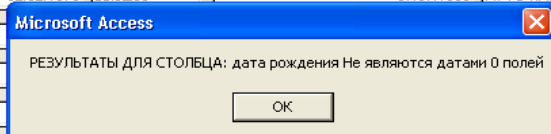


Рис. 25. Сообщение о том, сколько полей не являются датами для анализируемого столбца.

Затем сформируется сообщение о том, в каких записях фамилия не является датой. Это показано на рис. 26.

Антипович	11	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	холост
Борисович	9	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат
Зикторович	10	02.02.1978			МГТУ им. Н.Э. Баумана	холост
Петрович	11	02.02.1978			МГТУ им. Н.Э. Баумана	холост
Гордеевич	10	01.01.1980			МГТУ им. Н.Э. Баумана	холост
Зласович	10	02.02.1978	высшее	07.07.1993	МГТУ им. Н.Э. Баумана	женат

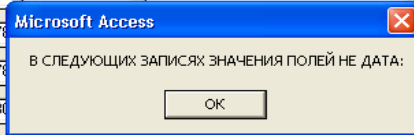


Рис. 26. Сообщение о том, в каких записях поля не являются датой.

Легко сделать вывод о том, что столбец “дата рождения” имеет тип дата.

Рассмотрим ситуацию менее очевидную. Например, в столбце “дата рождения” в некоторых полях указана информация следующего вида: 13.10.55 года. Строго говоря, такая запись не удовлетворяет типу “дата”. При анализе такой ситуации сформируется сообщение, представленное на рис. 27.

Microsoft Access					
РЕЗУЛЬТАТЫ ДЛЯ СТОЛБЦА: дата рождения Не являются датами 2 полей					
OK					

Рис. 27. Сообщение о несоответствии дате 2-х полей

Далее сформируется сообщение (рис. 28).

Microsoft Access					
В СЛЕДУЮЩИХ ЗАПИСЯХ ЗНАЧЕНИЯ ПОЛЕЙ НЕ ДАТА: , 18, 19					
OK					

Рис. 28. Сообщение о номерах записей, где поле “дата рождения” не соответствует типу “дата”

В соответствии с полученными сообщениями разработчик может принять решение о том, что поля в записях 18 и 19 необходимо исправить.

Событию “нажатие кнопки” “Анализ типов элементов столбцов” поставлена в соответствие процедура, фрагмент которой приведен ниже. Для удобства разьяснения основных приемов операторы пронумерованы.

1. Private Sub Кнопка0_Click()
2. On Error GoTo nd
3. cnd = 0
4. Dim mdb As Database ' mdb объявляется как база данных
5. Dim rst As Recordset ' rst объявляется как массив записей
6. Dim a(10) As Variant ' Массив заголовков исходной таблицы
7. Dim kp As String
8. Dim np As String
9. Set mdb = CurrentDb ' mdb назначается текущая база данных
10. Set rst = mdb.OpenRecordset("t1") ' rst назначается массив записей из таблицы t1
11. DoCmd.OpenForm "t1", acNormal, "", "", acEdit, acNormal ' открывается форма t1 для таблицы t1
12. DoCmd.GoToRecord , , acFirst ' переход на первую запись формы
13. kp = ""
14. np = ""
15. n = 0 ' счетчик имен столбцов исходной таблицы
16. For Each fld In rst.Fields ' в цикле перебираются все имена таблицы для их запоминания
17. ' MsgBox fld.Name
18. n = n + 1
19. a(n) = fld.Name
20. END fld
21. ' DoCmd.Close ' закрывается форма t1
22. DoCmd.OpenForm "t2", acNormal, "", "", acEdit, acNormal
23. DoCmd.GoToRecord acForm, "t2", acLast
24. DoCmd.GoToRecord , , acFirst
25. c = Forms!t2!cpol ' запоминается число полей
26. cnd = 0
27. For i = 1 To c
28. p11 = Forms!t2!p1
29. v = FormatDateTime(p11)
30. GoTo d
31. nd: cnd = cnd + 1
32. np = np & ", " & i
33. Resume END
34. GoTo d
35. d: DoCmd.GoToRecord , , acEND

```

36.END i
37.DoCmd.Close acForm, "t2"
38.MsgBox ("РЕЗУЛЬТАТЫ ДЛЯ СТОЛБЦА: " & a(1) & " Не являются
    датами " & cnd & " полей")
39.MsgBox ("В СЛЕДУЮЩИХ ЗАПИСЯХ ЗНАЧЕНИЯ ПОЛЕЙ НЕ
    ДАТА: " & np)
40.DoCmd.Close acForm, "t1"
41.funnnunber (0)
42.End Sub

```

Большинство операторов совпадает с операторами процедуры, рассмотренной в предыдущем параграфе. Рассматриваются принципиальные отличия.

Все построено на выявлении ошибок преобразования типа, фиксирования количества этих ошибок, номеров строк и имен столбцов. Для этого используется оператор 2 (On Error GoTo nd). Ошибка преобразования может возникнуть в операторе 29 $v = \text{FormatDateTime}(p11)$. Здесь осуществляется попытка преобразования типа текущего поля в тип “дата”. Если преобразование невозможно, то возникает ошибка и переход на метку nd (см. предыдущий оператор). В операторе 31, помеченным этой меткой, увеличивается счетчик недат. В операторе 32 запоминается номер строки, в которой анализируемое поле не является дата. В операторе 33 выполняется сброс ошибки. В операторе 34 выполняется переход к дальнейшей обработке записей. Выбирается следующая запись и начинается следующая итерация цикла перебора записей.

Если ошибки преобразования не возникло после выполнения оператора 29, то осуществляется переход к дальнейшей обработке записей. Выбирается следующая запись и начинается следующая итерация цикла перебора записей.

Таким образом, в результате работы процедуры будут выявлены все поля анализируемого столбца, которые не являются датой и сохранятся номера соответствующих записей.

После окончания цикла закрывается вспомогательная таблица “t2” и выводятся сообщения о числе полей, не являющимися датами и сообщение о номерах соответствующих строк.

Но поля проверены только на соответствие типу “дата”. Проверку на соответствие типу “число” в данном контексте выполнить невозможно, т.к. неизвестно по какому поводу возникла ситуация On Error.

Для выхода из этой ситуации написана функция, которая вызывается в операторе 42 (funnnunber (0)).

Ниже представлена эта функция.

1. Private Function funnnunber(r) As Integer
2. On Error GoTo nd
3. cnd = 0
4. Dim mdb As Database ' mdb объявляется как база данных
5. Dim rst As Recordset ' rst объявляется как массив записей
6. Dim a(10) As Variant ' Массив заголовков исходной таблицы
7. Dim kp As String
8. Dim np As String
9. Set mdb = CurrentDb ' mdb назначается текущая база данных
10. Set rst = mdb.OpenRecordset("t1") ' rst назначается массив записей из таблицы t1
11. DoCmd.OpenForm "t1", acNormal, "", "", acEdit, acNormal ' открывается форма t1 для таблицы t1
12. DoCmd.GoToRecord , , acFirst ' переход на первую запись формы
13. kp = ""
14. np = ""
15. n = 0 ' счетчик имен столбцов исходной таблицы
16. For Each fld In rst.Fields ' в цикле перебираются все имена таблицы для их запоминания
17. ' MsgBox fld.Name
18. n = n + 1
19. a(n) = fld.Name
20. END fld
21. ' DoCmd.Close ' закрывается форма t1
22. DoCmd.OpenForm "t2", acNormal, "", "", acEdit, acNormal
23. DoCmd.GoToRecord acForm, "t2", acLast
24. DoCmd.GoToRecord , , acFirst
25. c = Forms!t2!cpol ' запоминается число полей
26. cnd = 0
27. For i = 1 To c
28. p11 = Forms!t2!p1

```

29.v = FormatNumber(p11)
30.GoTo d
31.nd: cnd = cnd + 1
32.np = np & ", " & i
33.Resume END
34.GoTo d
35.d: DoCmd.GoToRecord , , acEND
36.END i
43.DoCmd.Close acForm, "t2"
44.MsgBox ("РЕЗУЛЬТАТЫ ДЛЯ СТОЛБЦА: " & a(1) & " НЕ
    ЯВЛЯЮТСЯ ЧИСЛАМИ " & cnd & " ПОЛЕЙ")
45.MsgBox ("В СЛЕДУЮЩИХ ЗАПИСЯХ ЗНАЧЕНИЯ ПОЛЕЙ НЕ
    ЧИСЛА: " & np)
46.DoCmd.Close acForm, "t1"
47.End Function

```

По сути, эта функция практически не отличается от вызывающей процедуры. Главное отличие в операторе 29 ($v = \text{FormatNumber}(p11)$). В этом операторе осуществляется попытка преобразования текущего поля в число. Далее работа программы выполняется аналогично. Конечно, в конце функции формируются соответствующие ей сообщения.

3.2. Избавление от дублирующихся записей

Избавление от дублирующихся записей является несложной процедурой. Для ее реализации необходимо 3-и запроса: запрос на создание новой таблицы на базе исходной, в которой отсутствуют повторяющиеся записи; запрос на очистку исходной таблицы; запрос на добавление таблицы, в которой отсутствуют повторяющиеся записи в исходную таблицу.

Продемонстрируем разработки, выполненные в этом направлении.

На рис. 29 продемонстрирована таблица, в которой 3-и последние записи повторяются.

Выдрин	Сидор	Викторович	10	02.02.1978
Гасов	Борис	Петрович	11	02.02.1978
Гуров	Григорий	Гордеевич	10	01.01.1980
Гусев	Владимир	Власович	10	02.02.1978
Демин	Геннадий	Григорьевич	9	02.02.1978
Долгов	Демьян	Гурович	10	02.02.1978
Ершов	Кирилл	Денисович	10	02.02.1978
Зинин	Данила	Дмитриевич	9	02.02.1978
Санаев	Иван	Иванович	11	01.01.1980
Сидоров	Петр	Сергеевич	12	01.01.1979
Фомин	Сидор	Михайлович	11	02.02.1978
Фомин	Сидор	Михайлович	11	02.02.1978
Фомин	Сидор	Михайлович	11	02.02.1978

Рис. 29. Таблица, в которой 3-и последние записи повторяются.

Далее на рис. 30 приведен фрагмент интерфейса для исключения дублирования записей.



Рис. 30. Фрагмент интерфейса для исключения дублирования записей.

После нажатия кнопки “Исключение дублирования записей” сформируется уведомляющее сообщение (рисунок 31).

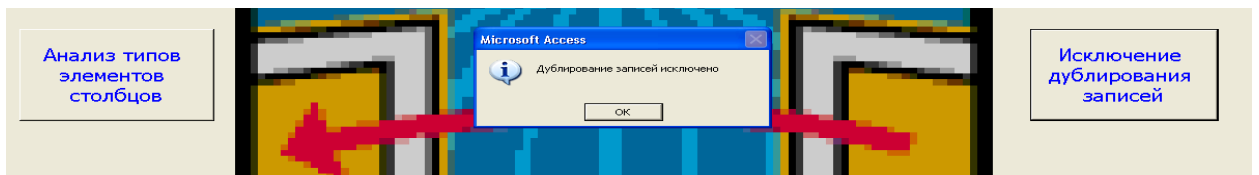


Рис. 31. Уведомляющее сообщение.

После выполнения устранения дублирования исходная таблица примет вид (рис. 32).

Выдрин	Сидор	Викторович	10	02.02.1978
Гасов	Борис	Петрович	11	02.02.1978
Гуров	Григорий	Гордеевич	10	01.01.1980
Гусев	Владимир	Власович	10	02.02.1978
Демин	Геннадий	Григорьевич	9	02.02.1978
Долгов	Демьян	Гурович	10	02.02.1978
Ершов	Кирилл	Денисович	10	02.02.1978
Зинин	Данила	Дмитриевич	9	02.02.1978
Санаев	Иван	Иванович	11	01.01.1980
Сидоров	Петр	Сергеевич	12	01.01.1979
Фомин	Сидор	Михайлович	11	02.02.1978

Рис. 32. Вид таблицы после устранения дублирования записей

Ниже приведены сформированные запросы.

Запрос на создание новой таблицы на базе исходной, в которой отсутствуют повторяющиеся записи.

```
SELECT DISTINCT t1.* INTO t3
FROM t1;
```

Запрос на очистку исходной таблицы.

```
DELETE t2.*
FROM t2;
```

Запрос на добавление таблицы, в которой отсутствуют повторяющиеся записи в исходную таблицу.

```
INSERT INTO t1
SELECT t3.*
FROM t3;
```

Все эти запросы сгруппированы в макрос и выполняются автоматически друг за другом при обращении к макросу. Макрос поставлен в соответствии событию “нажатие кнопки” “Исключение дублирования записей”.

3.3. Оценка вычислительной сложности процедур анализа типов атрибутов таблицы и избавления от дублирующихся записей

Оценка вычислительной сложности процедур анализа типов

Базовым оператором для оценки вычислительной сложности является оператор преобразования типа. Во-первых, это один из самых долго выполняемых операторов, во-вторых, он присутствует во всех циклах.

Главная составляющая оценки – это число записей таблицы или ее мощность (m). Действительно необходимо проверить поля всех записей.

Второй составляющей оценки является число столбцов или степень таблицы (n), ведь необходимо проверить все столбцы.

Третья составляющая оценки является число преобразуемых таблиц (k).

Таким образом, вычислительная сложность процедуры анализа типов составит $VS = \sum_k m_k n_k$.

Эта вычислительная сложность для данной предметной области небольшая. Ведь для 100 таблиц с 1000 записями и 100 столбцами (БД более чем средней сложности) число операций преобразований типа будет 10.000.000. Как показал эксперимент, данное количество сравнений на компьютере с частотой 1,6 МГц выполняется за 11 секунд.

Самой большой временной составляющей в процессе анализа и формирования типов столбцов ИТВР являются временные затраты на действия разработчика, которые трудно оценить. Но, как показывает опыт разработок, при использовании разработанных процедур время назначения типов столбцов в ИТВР сокращается в десятки раз.

ПРИЛОЖЕНИЕ 2.

**АКТ ВНЕДРЕНИЯ И ИСПОЛЬЗОВАНИЯ РЕЗУЛЬТАТОВ
ДИССЕРТАЦИОННОЙ РАБОТЫ**