# Организация эффективной системы хранения фактов в онтологиях

А.А. Левков

**Аннотация**. Предлагается способ трансляции терминологических выражений дескрипционной логики в иерархически-реляционные структуры схемы данных, что позволяет существенно упростить верификацию фактов и снизить вычислительную сложность операций с ними.

**Ключевые слова**: онтология, дескрипционная логика, база данных, реляционная модель, иерархическая реляционная модель, верификация фактов.

## Введение

В настоящее время все большее распространение как формальное описание различных предметных областей получает онтологический подход. Использование онтологий унифицирует форму описания знаний о предметной области, позволяет использовать предикатную систему вывода для получения ответов на вопросы и в целом способно заменить среднеквалифицированного эксперта при ответе на правильно составленные запросы.

Для описания онтологий наиболее часто используются дескрипционные логики (ДЛ). Существующие реализации ДЛ высокоэффективны при работе с терминологическими аксиомами (*TBox*) в режиме вывода. Одной из основных проблем построения эффективных онтологий является проблема обработки больших массивов фактов (*ABox*): их поиска, анализа принадлежности к концепту и, особенно, верификации – проверки на соответствие терминологическим аксиомам онтологии.

Наиболее распространенным и эффективным средством обработки данных в настоящее время являются реляционные базы данных (РБД), основанные на псевдо-реляционном исчислении *sql* (на прикладном уровне разница между чисто реляционным и псевдо-реляционным исчислением невелика, однако в теоретической части имеются существенные различия). Так как и реляционная алгебра и ДЛ происходят из алгебры логики, они имеют много общих черт, однако, несмотря на их значительную схожесть, автором не было найдено материалов, посвященных возможностям интеграции механизмов описания знаний (ДЛ) и хранения данных (РБД). В [1] было отмечено, что ДЛ можно использовать в качестве системы контроля целостности данных, но предложенные решения соотносились только с простейшими структурами в объектно-ориентированных языках и не затрагивали такой области массового хранения данных, как РБД. В работах [2,3] рассматривались только вопросы трансляции выражений ДЛ и *sparql* в выражения реляционной алгебры для нереляционных моделей «Сущность»-«Атрибут»-«Значение» (*EAV-model*), без рассмотрения возможных механизмов адаптации и оптимизации реляционных схем данных.

В данной статье предлагается методика построения эффективного хранилища *ABox* на основе иерархически-реляционной схемы данных, которая наиболее полно соответствует *TBox* онтологии и не нарушает реляционных требований к нормальным формам, что позволяет существенно повысить эффективность системы верификации данных и упрощает вычисление *ABox* запросов.

## 1. Трансляция терминологических аксиом в иерархичеки-реляционную схему данных

В терминологических аксиомах ДЛ (*TBox*) постулируются следующие понятия: концепт (*C*) и роль (*R*). Ближайшим аналогом этих понятий в реляционной алгебре являются сущность (*K*) и связь (*FK*) (в дальнейшем будет показано, что не любая связь в реляционной теории может трактоваться как роль). Поскольку реляционная алгебра является замкнутой, для нее выполняются индуктивные правила: любая операция над реляционными отношениями (сущностями) дает новое реляционное отношение [3], так же как и любая операция над концептами дает концепт.

### 1.1. Таксономия концептов

Одним из основных конструктивных приемов построения онтологии является использование аксиом вложенности концептов. Использование вложенности дает возможность строить таксономии, которые являются основой построения онтологий. Реляционная алгебра, предлагая 5 основных операций над реляционными сущностями (отношениями), никак не рассматривает построение иерархий, определяя только ассоциативные связи между отношениями. Типичная реляционная схема является сложно-связным ориентированным графом, имеющим несколько центров концентрации.

Несмотря на это, в реляционной теории не существует запрета на построение иерархий, которые возможно определить как связи 0-к-1 между «родительской» и «дочерней» сущностями, определенные на первичных ключах (*pk*). Так, следующие суждения о вложенности концептов:

$B \sqsubseteq A \qquad D \sqsubseteq B \qquad E \sqsubseteq B \qquad H \sqsubseteq C$

$C \sqsubseteq A \qquad F \sqsubseteq B \qquad G \sqsubseteq C$

можно выразить в виде таксономии реляционных сущностей

$B \frac{pk}{\theta} A \qquad D \frac{pk}{\theta} B \qquad F \frac{pk}{\theta} B \qquad H \frac{pk}{\theta} C$

$C \frac{pk}{\theta} A \qquad E \frac{pk}{\theta} B \qquad G \frac{pk}{\theta} C,$

как это показано на Рис.1.

При построении данной структуры каждая последующая по иерархии сущность является уточняющей, а не самостоятельной полноценной сущностью. Для получения полной сущности (*K*) в данной структуре необходимо произвести соединение всей ветви наследования сущности по следующей рекуррентной формуле:

$$\mathrm{K}^{full} = \begin{cases} если \nexists \mathrm{K}_{пред}, то \mathrm{K} \\ \mathrm{K} \frac{pk}{\theta} \mathrm{K}^{full}_{пред} \end{cases}. \qquad (1)$$

Атрибуты ($\Lambda$), описывающие данную сущность, могут быть получены при помощи операции объединения всех атрибутов согласно подобной же рекуррентной формуле:

$$\Lambda_{\mathrm{K}^{full}} = \begin{cases} если \nexists \mathrm{K}_{пред}, то \Lambda_{\mathrm{K}} \\ \Lambda_{\mathrm{K}} \bigcup \Lambda_{\mathrm{K}^{full}_{пред}} \end{cases}. \qquad (2)$$

Т.е. полная сущность *E* может быть определена следующими отношениями:

$E^{full} = (A) \frac{ID}{\theta} (B) \frac{ID}{\theta} (E),$

$\Lambda_{E^{full}} = \Lambda_A \bigcup \Lambda_B \bigcup \Lambda_E.$

### 1.2. Конструкторы

Организация предложенного механизма наследования сущностей позволяет не только выразить любые отношения вложенности концептов как отношения наследования реляционных сущностей, но и определять дизъюнктивные суждения в рамках конструкций реляционной схемы данных (СД), что ранее считалось возможным выразить только в виде составных отношений [4]. Строгая дизъюнкция $\propto = B \dot{\sqcup} C = A$ может быть определена в виде простого графа наследования отношений $B \frac{pk}{\theta} A$, $C \frac{pk}{\theta} A$, как это показано на Рис.2.

Нестрогая дизъюнкция $\propto = B \sqcup C = X \dot{\sqcup} Y \dot{\sqcup} Z = A$ может быть реализована при организации множественного наследования в таксономии реляционных сущностей $B \frac{pk}{\theta} A$, $C \frac{pk}{\theta} A$, $X \frac{pk}{\theta} C$, $Y \frac{pk}{\theta} C$, $Y \frac{pk}{\theta} B$, $Z \frac{pk}{\theta} B$, как это показано на Рис.3.

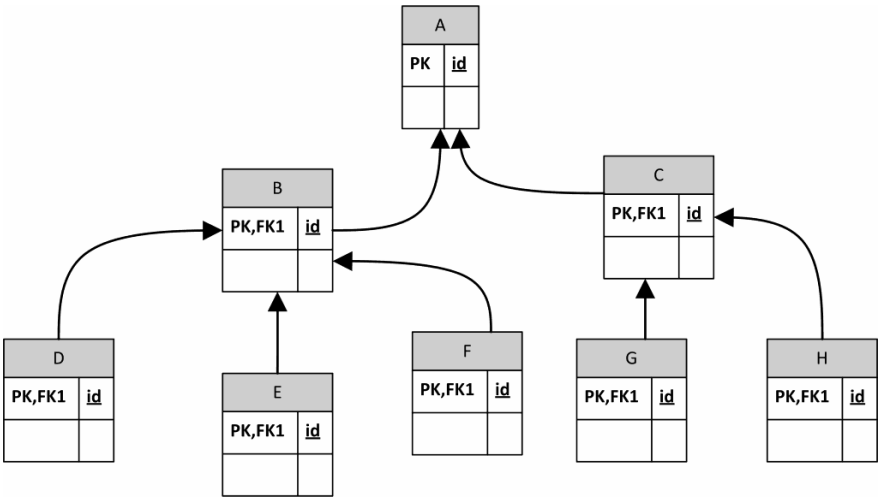


Рис.1. Построение таксономии реляционных сущностей

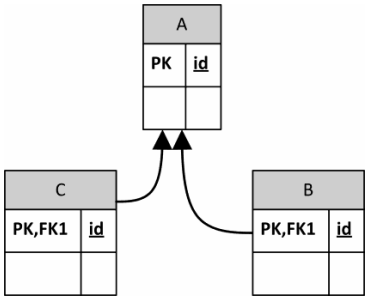


Рис.2. Отображение строгой дизъюнкции в реляционную СД

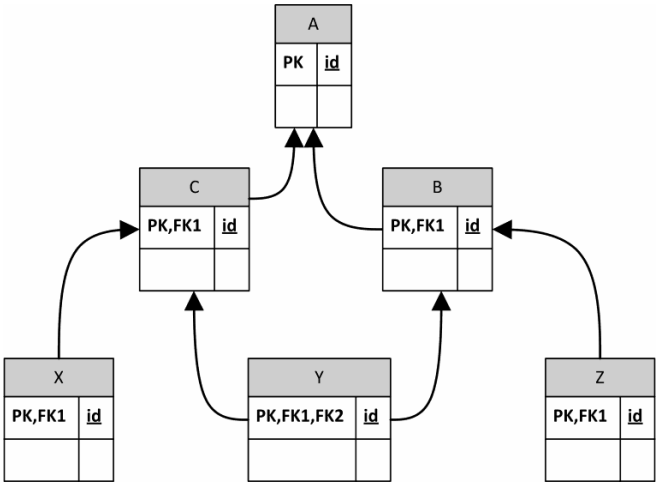


Рис.3. Отображение нестрогой дизъюнкции в реляционную СД

Множественно наследование, кроме нестрогой дизъюнкции, позволяет определять и конъюнктивные суждения на уровне концептов (сущностей) $\propto= B \sqcap C = D$ в виде $D \frac{pk}{\theta} C$, $D \frac{pk}{\theta} B$ (Рис.4.)

Согласно аксиоме ДЛ о том, что дополнение концепта также является концептом, можно разрешить дополнение концепта $\propto= A \sqcap \neg C = E$ как введение новой реляционной сущности ($E$) в схему данных, как показано на Рис.5. (отрицание=предок-концепт).

Таким образом, реализация реляционной СД как таксономии реляционных сущностей на основе связей между первичными ключами позволяет выражать конструкторы ДЛ в виде различных структурных организаций наследования.

### 1.3. Кванторы

Подобным же образом можно выразить кванторы ДЛ через реляционные выражения. Несмотря на наличие как квантора существования $\exists$, так и квантора всеобщности $\forall$ в реляционном исчислении, все непосредственные его реализации (имеется ввиду *sql*-исчисление, близкое, но не тождественное реляционному) предлагает единственный квантор exists, аналог квантору существования $\exists$. Связано это с тем, что согласно правилам эквивалентности через него можно выразить всеобщности следующим

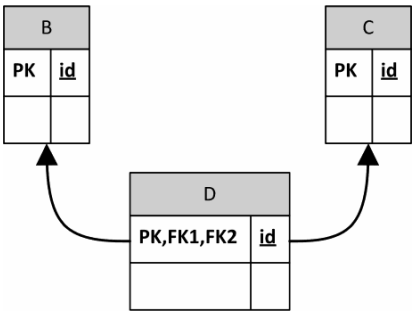


Рис.4. Отображение конъюнкции в реляционную СД

образом: $\forall A(pred) = \nexists A(\overline{pred})$ [3]. Таким образом, кванторы ДЛ имеют прямое отображение в реляционном исчислении и эквиваленты в sql-реализациях.

### 1.4. Роли

Естественным аналогом ролей в ДЛ являются связи в реляционной СД. Однако существует ряд ограничений на использование связей реляционной СД – не все они архитектурно могут выступать в качестве ролей ДЛ. Как известно, в реляционной теории существует только один тип связи – внешний ключ (*FK*), но использоваться он может по-разному. Самый простой способ реализации – это связи один-ко-многим, которые могут быть отражены как введение в зависимую сущность нового атрибута, содержащего значение первичного ключа (потенциального ключа) в главной сущности. Такая связь явно ограничена мощностью, именно она используется для построения реляционной таксономии. Данная связь не подходит для отражения ролей ДЛ ввиду своей неуниверсальности: ограничение на кардинальность ролей являются внешними по отношению к самим ролям – в зависимости от конкретной роли (и расширения ДЛ) это могут быть произвольные ограничения типа $\leq nR.C$, причем определенные для обоих направлений роли. Таким образом, общим универсальным отражением ролей ДЛ в реляционной СД могут быть только реляционные связи многие-ко-многим, которые реализуются через введение дополнительной реляционной сущности, содержащей атрибуты, хранящие значения первичных ключей связуемых сущностей, как это показано на Рис.6.

Такой подход позволяет интерпретировать кванторы для ролей в виде концептов: поскольку роль представляется в виде реляционного отношения, то и любые операции с ней согласно замкнутости реляционной теории дадут в результате реляционное отношение, которое в рамках предлагаемой трансляции трактуется как концепт. Кроме того, такое позволяет реализовать любые многоместные роли.

В предложенной реализации для каждой роли $R$ существует ее инверсия $R^-$, т.к. роль представлена в виде реляционного отношения с двумя внешними ключами. Направление соединения («слева» и «справа») и формирует либо роль, либо ее инверсию, что позволяет выражать инверсию ролей (***I*-расширение ДЛ**).

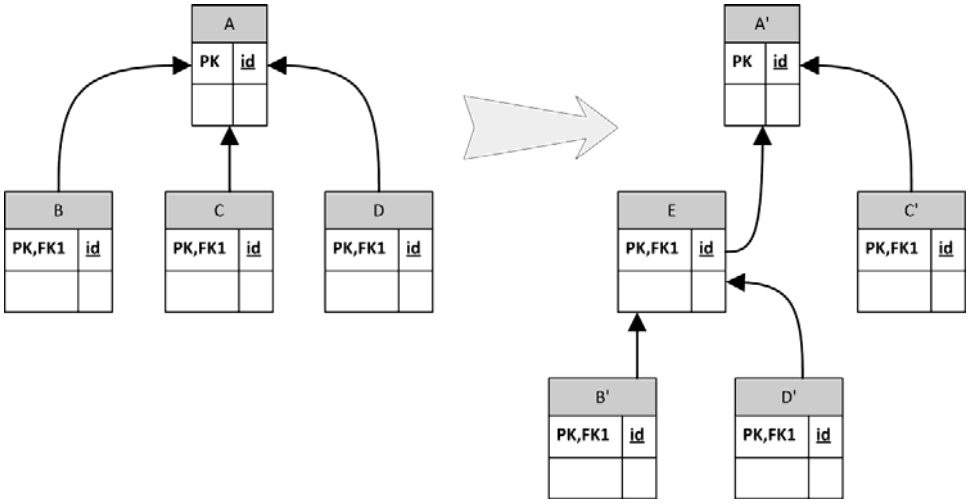


Рис.5.Отображение отрицания в реляционную СД

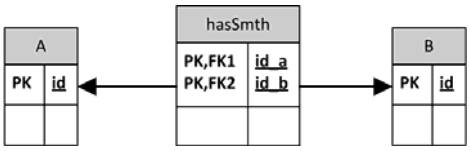


Рис.6. Реализация роли в реляционной СД

Если обобщить *TBox* и *RBox* на конструктивном уровне и интерпретировать роли как особый вид концептов, то это приведет к включению ролей в общую иерархию реляционных сущностей, т.е. даст возможность реализовывать иерархии ролей (***H*-расширение ДЛ**) и создавать роли относительно ролей, что позволит формулировать составные аксиомы вложенности ролей (***R*-расширение ДЛ**) как это показано на Рис.7.

Для задания транзитивных ролей (***S*-расширение ДЛ**) $aRb; bRc \Rightarrow aRc$ в современных реляционных БД возможно использовать такие *sql*-расширения, как общие табличные выражения, реализующие рекурсивные запросы. Так, для реализации роли «Часть-целое», которая является транзитивной, можно построить следующее *sql*-представление:

```
create view «A_Part_Of» as
with _CTE(Part, Whole) as (
select Part, Whole from «part_whole»
union all
select b.Part, a.Whole from «part_whole» a
inner join _CTE b on b.Whole=a.Part)
select Part, Whole from _CTE.
```

Подобным же образом можно выразить любую другую транзитивную зависимость, в том числе саму иерархию наследования концептов онтологии (следует отметить, что вычисление транзитивных зависимостей в случае *sparql*-запросов до сих пор остается нерешенной в общем виде проблемой [5]).

Реализовать ***F, N, Q*-расширения ДЛ** возможно на операционном уровне реляционной СД – на уровне триггеров *sql* (на самом деле процедурные расширения позволяют накладывать любые исчислимые ограничения, вплоть до эвристических методов).

***(D)* – расширение** языка конкретными типами данных неотъемлемо присутствует в реляционной трансляции ДЛ в связи с тем, что любая реляционная СУБД предоставляет ряд стандартных типов для работы с данными. Наиболее современные СУБД предлагают так-

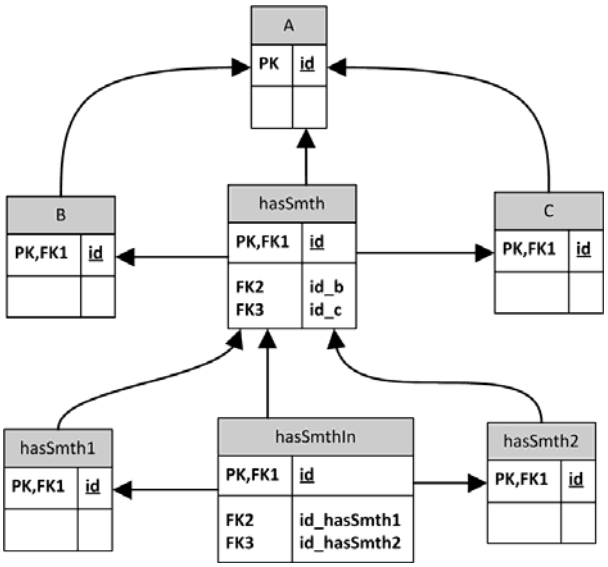


Рис.7. Иерархия и композиция ролей в реляционной СД

же и средства для создания собственных пользовательских типов данных.

Таким образом, предложенная трансляция *TBox* на реляционный базис в обязательном порядке соответствует логике *SI(D)*, но может быть использована для построения любой из *HRFNQ* логик и их комбинаций. Предложенная методика трансляции выражений дескрипционной логики (т.е. онтологических моделей) в реляционное пространство приводит к появлению нового класса иерархически-реляционных моделей, которые существенно отличаются от классических реляционных моделей. Иерархически-реляционные модели имеют сквозную иерархизацию (в отличие от локальной иерархизации в случае супертипов в ER-модели), что приводит к группировке общих атрибутов и связей реляционных сущностей на более высоких уровнях иерархии. Данная структурная организация при росте количества уровней в иерархии приводит к существенному уменьшению общего атрибутивно-ссылочного базиса всей схемы данных и ее сложности в целом [6].

## 2. Анализ вычислительной сложности запросов к фактам в иерархически-реляционной схеме данных

Одним из ключевых преимуществ предлагаемой иерархически-реляционной схемы данных является эффективная проверка корректно-

сти фактов, добавляемых (изменяемых и удаляемых) в *ABox*. Так как терминологические аксиомы выражаются в виде ограничений целостности реляционной схемы данных, то процесс верификации фактов сводится к хорошо отработанному процессу контроля реляционных данных. Таким образом, использование иерархически-реляционной схемы данных для хранения фактов, прямо отражающей *TBox* онтологии, позволяет напрямую использовать язык *sql* для написания запросов к фактам без потери гибкости в формулировке выражений.

Одним из основных запросов к фактам в онтологии является определение принадлежности объекта концепту ($a \in C$) [7]. В связи с реляционной природой хранимых фактов, возможно ввести ограничивающее условие на концепты: в иерархически-реляционной схеме необходимо наличие для каждого составного концепта эквивалентного ему простого концепта (синонима) – т.е. каждому концепту должна соответствовать реляционная сущность или представление. Наличие этого ограничивающего условия позволяет существенно упростить вычислительную сложность определения принадлежности объекта концепту.

Организация иерархии в реляционной схеме данных требует организации вторичных ключей по первичным ключам всех отношений, что при сквозной иерархизации приводит к наличию единого центра генерации первичных ключей отношений, находящегося в корне дерева наследования (сущность *A* на Рис.1). Наличие первичного ключа факта в каждой сущности (концепте) по иерархии наследования позволяет при известном первичном ключе и проверяемом концепте определить принадлежность объекта концепту на основе единичного запроса формы

```
select 1 from <entity> where pk=:?obj.
```

Первичный ключ реляционного отношения (в общем случае) является кластерным и организован на основе *B*-дерева, поэтому поиск элемента в худшем случае находится в диапазоне *DLOGTIME* и *LOGSPACE* [8] в зависимости от мощности *ABox* для всех диалектов, формулируемых через иерархически-реляционную схему ДЛ. В реальности вычислительная сложность будет даже несколько ниже в связи с тем, что мощность $|ABox|$ (количество фактов) в иерархически-реляционной схеме распределяется по всем сущностям; в простейшем случае при равномерном распределении данных это дает сложность $\frac{|ABox|}{|C|}$, где $|C|$ - количество концептов (реляционных сущностей).

Эффект от применения предложенного механизма хранения фактов проявляется и относительно размера терминологии системы. Так как каждому концепту соответствует реляционная сущность, а метаданные современных РБД также относятся к реляционным структурам (т.е. каждая сущность представлена в виде соответствующего кортежа служебного отношения), то время нахождения соответствующей сущности в общем случае также ограничено *DLOGTIME* и *LOGSPACE* [9]. То есть *NEXPTIME*-сложная задача $a \in C$ относительно мощности *TBox* [8] также сводится к *DLOGTIME* и *LOGSPACE* сложности в общем случае. Следует отдельно отметить, что специфика работы РБД такова, что для большинства систем сложность $a \in C$ относительно мощности *TBox* будет даже ниже *DLOGTIME*, что связано с использованием пре-компилирования запросов в РСУБД. В этом случае сложность оказывается в пределах $O(c)$ для $|TBox| < T'$ (где $T' = f(V_{RAM})$ – зависит от объема доступной внутренней памяти) и имеет *DLOGTIME* сложность для $|TBox| \geq T'$.

Такое значительное снижение вычислительной сложности принадлежности объекта концепту возможно только при условии неизменности *TBox*, т.к. любое добавление или удаление концепта приводит к перестроению иерархически-реляционной схемы данных, что является вычислительно сложной задачей. Так, добавление нового концепта приводит к добавлению новой реляционной сущности и требует заполнения ее корректными данными. Сложность этой операции относительно мощности *ABox* относится к классу *P*-полных [8] (при равномерном распределении относитель-

но $\frac{|ABox|}{|C|}$) и *DLOGTIME* и *LOGSPACE* сложной относительно мощности *TBox*.

Таким образом, использование иерархически-реляционной схемы данных позволяет существенно упростить работу с фактами в онтологии при неизменном *TBox* за счет эффективного распределения данных в иерархи и реляционных сущностей, сводя в общем случае *NEXPTIME*-сложную задачу [9] к *DLOGTIME* и *LOGSPACE* сложной. Платой за данное повышение эффективности является существенный рост сложности операций с *TBox*, что делает предлагаемое решение эффективным только для систем с большим объемом изменений в *ABox* при относительно стационарном *TBox*.

## Заключение

С использованием предложенного механизма трансляции онтологии в реляционно-иерархическую схему данных становится возможным выразить терминологические аксиомы (*TBox*) логик семейства *SI(D)+HRFNQ* в виде ограничений целостности – реляционной схемы данных (и триггеров на сущностях для реализации *F,N,Q*-логик). Набор утверждений об индивидах (*ABox*) может быть интерпретирован как данные (кортежи), наполняющие реляционные отношения, принадлежащие созданной СД. Это дает возможность автоматизировать построение реляционных СД на основе существующих онтологий, что позволяет обойтись без выполнения сложных запросов для верификации данных, как это предлагается в работах других авторов [2].

Таким образом, если ранее реляционные базы данных использовались либо в качестве хранилищ нереляционных моделей (для хранения RDF-схем), что приводило к необходимости формирования «внешнего» механизма верификации фактов, либо требовали сложных экспертных преобразований RDF-схем в реляционные, то предлагаемый метод позволяет на основе онтологии сразу генерировать реляционные СД, комплементарные ей. Применение данного подхода позволяет существенно упростить вычислительную сложность *ABox* (проверка принадлежности концепту $-a \in C$) для сложных ДЛ (*SI(D)+HRFNQ*), сведя его к классам сложности *DLOGTIME* и *LOGSPACE* за счет постулирования статического *TBox*.

## Литература


1. E. Sirin J. Tao // Towards Integrity Constraints in OWL // OWLED, 2009.
2. B. Motik, I. Horrocks, U. Sattler // Bridging the gap between owl and relational databases // In Proceedings of the 16th international conference on WWW2007, ACM Press, 2007.
3. R. Lu, F. Cao, L. Ma, Y. Yu, Y. Pan // An Effective SPARQL Support over Relational Databases // In Proc. Of SWDB-ODBIS07 co-located with VLDB 2007.
4. К. Дж. Дейт. Введение в системы баз данных — 8-е изд. // М.: Вильямс, 2006.
5. M. Schmidt // Foundations of SPARQL Query Optimization // Albert-Ludwigs-Universitat Freiburg, 2009.
6. Б.Г. Ильясов, А.А. Левков // Оценка структурно-алгоритмической сложности реляционных схем данных // Вестник компьютерных и информационных технологий, 2011 №4.
7. F. Baader // The Description Logic Handbook // New York: Cambridge University Press, 2003.
8. Дональд Кнут // Искусство программирования, том 3. Сортировка и поиск 2-е изд.
9. А.А. Бездушный // Математическая модель системы интеграции данных на основе онтологий // Вестник НГУ 2008.


**Левков Александр Александрович.** Докторант УГАТУ. Окончил Уфимский государственный авиационный технический университет в 2000 году. Кандидат технических наук, доцент. Опубликовано 34 работы, в том числе 1 монография. Область научных интересов: реляционные базы данных, корпоративные информационные системы, интеллектуальные системы управления. E-mail: projekctor@gmail.com.