ИНЖИНИРИНГ ОНТОЛОГИЙ

УДК 001.92 *Научная статья* DOI: 10.18287/2223-9537-2025-15-4-552-565

МП-целостность при проектировании реляционных моделей баз данных

© 2025, В.В. Миронов ✉, К.В. Миронов

Уфимский университет науки и технологий (УУНиТ), Уфа, Россия

Аннотация

Обсуждается особый вид целостности в базах данных – целостность «множественности предка» (МП-целостность), понятие которого было введено авторами на концептуально-онтологическом уровне применительно к моделям «сущность–связь». В данной статье это понятие распространяется на реляционную модель с целью практического применения при создании баз данных. Рассматриваются связанные с этим видом целостности понятия: линия восходящего родства, отношение и ограничение множественности предка. Для перехода к реляционной модели требуется учёт также первичных и внешних ключей, ссылочной целостности, табличных триггеров. В качестве универсального средства обеспечения целостности в реляционной среде предлагается подход на основе применения триггеров базы данных. Триггеры обнаруживают и блокируют операции вставки и обновления строк в таблицах, которые ведут к нарушению целостности. Отмечается необходимость процедурного программирования триггеров, а также сложность переноса данных между системами баз данных разного вида из-за различий языков процедурного программирования. Предлагается подход на основе использования возможностей поддержания ссылочной целостности. В этом случае целостность множественности предка обеспечивается как часть ссылочной целостности. Конкретное решение зависит от использования простых/составных, натуральных/суррогатных ключей, а также от технических ограничений среды реализации базы данных. Может потребоваться введение избыточных компонентов в состав ключей для отслеживания экземпляра предка вдоль линий восходящего родства. Приведены тестовые примеры реляционных моделей в различных реляционных средах (полностью – в *MySQL* и *MariaDB*, и частично в *PostgreSQL*, *MS SQL Server, Oracle Database*).

Ключевые слова: *модель «сущность–связь», реляционная модель, множественность предка, линия восходящего родства, триггеры базы данных, внешние ключи, ссылочная целостность.*

Цитирование: *Миронов В.В., Миронов К.В.* МП-целостность при проектировании реляционных моделей баз данных. *Онтология проектирования*. 2025. Т.15, №4(58). С.552-565. DOI: 10.18287/2223-9537-2025-15-4-552-565.

Вклад авторов: *Миронов В.В.*– идея, концепция, формализм; *Миронов К.В.*– модели, алгоритмы, программирование и отладка в различных средах.

Конфликт интересов: авторы заявляют об отсутствии конфликта интересов.

Введение

Онтологический подход при проектировании баз данных (БД) [1, 2] позволяет задавать целевые информационные потребности и ограничения целостности на этапе концептуального проектирования БД [3, 4]. В работе [5], базируясь на онтологическом подходе, введена целостность множественности предка (МП-целостность) – специфическое ограничение целостности, которое часто встречается на практике. Рассмотрение МП-целостности выполнено в рамках модели «сущность–связь» (СС-модель) – модели концептуального уровня абстракции, отражающей онтологические особенности предметной области в виде системы классов

сущностей и связей. Подобные модели разрабатываются на начальной стадии проектирования БД независимо от способа её реализации. На следующей стадии создаются логические модели, учитывающие возможности и ограничения используемой системы управления БД (СУБД) [6–8]. Наиболее распространённой основой для построения БД являются реляционные СУБД [9].

Данная статья посвящена исследованию практического применения МП-целостности в реляционной среде. Обсуждаются базовые положения МП-целостности, существенные особенности реляционной модели (РМ), обеспечение МП-целостности на основе суррогатных и натуральных ключей. На это исследование значительное влияние оказали работы в области философия «реляционной онтологии» [10, 11], онтологических аспектов системного анализа [12–14], применение онтологического подхода [15, 16].

1 Базовые положения МР-целостности

Здесь представлено развитие базовых понятий и положений МР-целостности, введённых в [5] (см. рисунок 1).

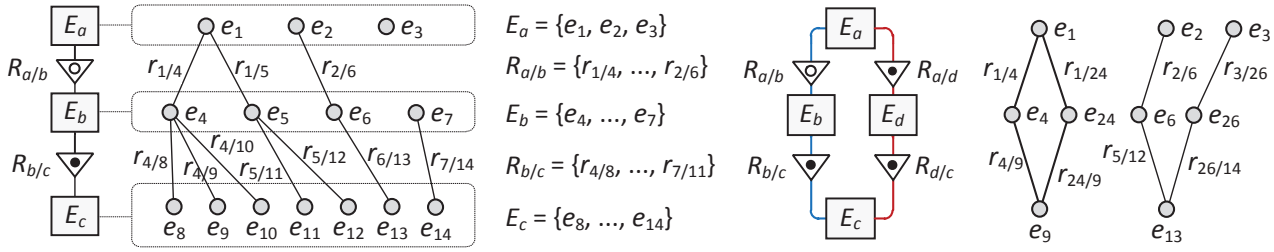


Рисунок 1 – К понятию МП-целостности:
а) – пример СС-модели как модели классов; б) – пример экземпляров сущностей и связей;
в) – сущности и связи как множества экземпляров; г) – МП-отношение двух ЛВР;
д) – экземпляры с несогласованной МП-целостностью

СС-модель задаёт онтологию БД в виде множества классов сущностей и множества классов бинарных связей типа «родитель–ребёнок» («один-ко-многим»). Рассматривается связка «класс–экземпляр», в которой каждый класс содержит множество экземпляров.

Линия восходящего родства (ЛВР) – цепочка «ребёнок–родитель», ведущая от некоторого потомка к некоторому предку («снизу вверх»). ЛВР L представляет собой последовательность классов сущностей E, в которой каждая последующая сущность является родителем своей предшествующей сущности через некоторую связь R из множества классов связей (см. рисунок 1а):

$$L = E_c \xrightarrow{R_{b/c}} E_b \xrightarrow{R_{a/b}} E_a.$$

ЛВР как класс представляет собой множество экземпляров l, где экземпляры – это цепочки экземпляров сущностей e, ведущих от экземпляра потомка к экземпляру своего предка (см. рисунок 1б):

$$L = \{l_{8\backslash 1}, l_{9\backslash 1}, l_{10\backslash 1}, l_{11\backslash 1}, l_{12\backslash 1}, l_{13\backslash 2}, l_{14\backslash \mathrm{null}}\}.$$

Например, ЛВР-экземпляры $l_{8\backslash 1}$ и $l_{14\backslash \mathrm{null}}$ представляют собой цепочки

$$l_{8\backslash 1} = e_8 \xrightarrow{r_{4/8}} e_4 \xrightarrow{r_{1/4}} e_1 \quad \text{и} \quad l_{14\backslash \mathrm{null}} = e_{14} \xrightarrow{r_{7/14}} e_7 \rightarrow \mathrm{null}.$$

ЛВР как функция для каждого аргумента – экземпляра потомка – задаёт экземпляр предка (или null-значение). Например, на рисунке 1б

$$L(e_8) = L(e_9) = L(e_{10}) = L(e_{11}) = L(e_{12}) = e_1, \quad L(e_{13}) = e_2, \quad L(e_{14}) = \mathrm{null}.$$

МП-отношение – это совокупность M нескольких ЛВР, ведущих разными путями от общего потомка (МП-потомка) к общему предку (МП-предку). На рисунке 1г представлено МП-отношение, содержащее две ЛВР:

$$M = \{L_1 | L_2\}, \quad L_1 = E_c \xrightarrow{R_{b/c}} E_b \xrightarrow{R_{a/b}} E_a, \quad L_2 = E_c \xrightarrow{R_{d/c}} E_d \xrightarrow{R_{a/d}} E_a.$$

МП-ограничение – это условие (предикат), заданное на МП-отношении, которое накладывается на совместные значения экземпляров МП-предка, полученные для каждого экземпляра МП-потомка через различные ЛВР. Это может быть:

- МП-равенство (МП$^{+}$), если требуется, чтобы у каждого экземпляра МП-потомка был один и тот же экземпляр МП-предка для всех ЛВР;
- МП-неравенство (МП$^{–}$), если требуется, чтобы у каждого экземпляра МП-потомка были разные экземпляры МП-предка всех ЛВР;
- более сложное условие.

Таким образом, МП-целостность СС-модели соответствует выполнению всех установленных МП-ограничений. В примере на рисунке 1д приведены два экземпляра МП-отношения, которые не удовлетворяют условиям МП$^{+}$/МП$^{–}$. Если в этом примере требуется МП$^{+}$-целостность, то у экземпляров e_6 и e_{26} должен быть один общий экземпляр МП-предка; если же требуется МП$^{–}$-целостность, то у экземпляров e_4 и e_{24} должно быть два разных экземпляра МП-предка.

2 Особенности реляционной модели для МП-целостности

Для выполнения МП-целостности в РМ требуется учитывать их особенности [10]. В РМ классам сущностей соответствуют таблицы реляционной БД. Атрибутам сущностей соответствуют столбцы таблицы. Экземплярам сущностей соответствуют строки таблицы. Строки идентифицируются значениями столбцов, составляющих первичный ключ (*Primary Key – PK*). *PK* может быть задан двумя способами:

- в виде суррогатного ключа (*Surrogate Key – SK*) – дополнительного числового столбца в таблице, для которого при вставке новой строки автоматически генерируются уникальные значения (значения *SK* неизменны в течение жизни строки таблицы);
- в виде натурального ключа (*Natural Key – NK*) – одного или нескольких столбцов таблицы, соответствующих первичным атрибутам сущности (*NK* могут быть составными, значения *NK* могут изменяться в течение жизни строки таблицы).

Помимо *PK* в таблице может быть задано несколько альтернативных ключей (*Unique Key – UK*), каждый из которых тоже идентифицирует строки таблицы. Связи между сущностями реализуются с помощью внешних ключей (*Foreign Key – FK*), которые представляют собой *PK* (или *UK*) таблицы-родителя, скопированные в таблице-ребёнке. Таким образом, *FK* является ссылкой из таблицы-ребёнка на таблицу-родителя. Для идентифицирующих связей *FK* входит в состав *PK* таблицы, а для неидентифицирующих не входит.

В БД автоматически поддерживается ссылочная целостность (*Referential Integrity – RI*), основанная на принципе: не должно быть детей несуществующих родителей, т.е. для каждого значения *FK* должно существовать такое же значение *PK* (*UK*) в таблице-родителе. В СУБД реализуется *RI* путём контроля операций обработки данных в соответствии с выбранными типами *RI*-правил, например:

- правило RESTRICT («строгое», действует по умолчанию) блокирует операцию вставки, удаления или обновления, если она ведёт к нарушению ссылочной целостности;
- правило ON UPDATE CASCADE («каскадное обновление») автоматически обновляет значения *FK* при обновлении соответствующего *PK/UK* и др.

Для обеспечения нестандартных требований целостности предусмотрены триггеры БД (*DB Triggers*) – особые процедуры, хранимые в БД. Триггер связан с таблицей и автоматически запускается при работе с данными в этой таблице. Например, триггер типа BEFORE INSERT запускается перед выполнением операции вставки в таблицу новой записи, а триггер типа BEFORE UPDATE – перед выполнением операции обновления существующей записи. Процедура триггера может в т.ч. проверять текущее содержимое таблиц БД и на этом основании блокировать выполнение операции.

3 МП-целостность при использовании суррогатных ключей

При обеспечении МП-целостности в реляционной БД на основе *SK* необходимо учитывать две особенности: неидентифицирующий характер связей между таблицами; неизменность значений ключей – идентификаторов строк таблицы.

Первая особенность выражается в том, что экземпляры дочерних сущностей идентифицируются независимо от своих родителей. На концептуальной модели это обстоятельство отражается наличием светлых квадратиков в символах связи (рисунок 2а). В этом примере все сущности, кроме сущности Сдача, самоидентифицирующиеся, т.е. идентификация их экземпляров не зависит от того, с какими экземплярами других сущностей они связаны.

Сущность Сдача не является самоидентифицирующейся, каждый её экземпляр соответствует некоторой паре экземпляров сущностей Студент и Предмет, т.е. является полным «иждивенцем» своих «кормильцев» Студент и Предмет. Это обстоятельство отражено в модели наличием темных квадратиков у символов связи.

При переходе к РМ сущности становятся таблицами (см. рисунок 2б). В каждой таблице, если она не является полным иждивенцем, размещается *SK* (*SK* обозначены звёздами, а имена содержат префикс «Ид» – идентификатор). Связи преобразуются в *FK* (*FK* обозначены треугольниками, указывающими на соответствующий ключ родительской таблицы).

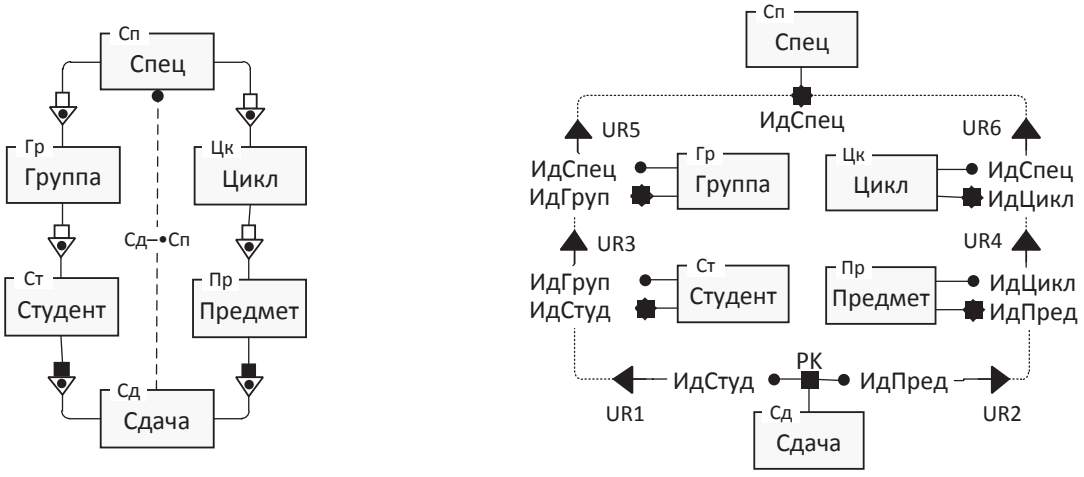


Рисунок 2 – Пример моделей с неидентифицирующими связями:
а) – СС-модель; б) – реляционная модель на основе суррогатных ключей

Таблицы, являющиеся полными иждивенцами (таблица Сдача), получают составной *PK* (отмечен тёмным квадратом), компонентами которого являются *FK*, ссылающиеся на родителей-кормильцев. Для *FK* задаются *RI*-правила (поскольку значения *SK* не изменяются в течение жизненного цикла (ЖЦ), это RESTRICT-правило). На рисунке 2а МП-отношение Сд—●Сп запрещает студенту сдачу «чужих» предметов и содержит две ЛВР, заданные цепочками ссылок *FK*:

$$\text{Сд}\text{—}\bullet\text{Сп} = \left\{ \text{Сд} \xrightarrow{\text{UR1}} \text{Ст} \xrightarrow{\text{UR2}} \text{Гр} \xrightarrow{\text{UR3}} \text{Сп} \;\middle|\; \text{Сд} \xrightarrow{\text{UR2}} \text{Пр} \xrightarrow{\text{UR4}} \text{Цк} \xrightarrow{\text{UR6}} \text{Сп} \right\}.$$

Здесь стрелка обозначает *RI*-ограничение; слева от стрелки – таблица, в которой определено это ограничение; справа – родительская таблица, на которую ссылается ограничение; над стрелкой указано имя ограничения.

Операции, которые потенциально могут привести к нарушению МП-целостности, связаны с операциями работы с данными *FK* во всех таблицах МП-отношения, за исключением таблицы МП-предка (таблицы Спец).

- Для таблицы, являющейся МП-потомком, – это операции вставки новой строки (INSERT) или изменения (UPDATE) в существующей строке значений *FK*, являющихся компонентами *PK*. В примере на рисунке 2б это относится к таблице Сдача, в которую может быть добавлена новая или изменена имеющаяся строка так, что появляется сдача студентом «чужого» предмета. Такие операции возможны без нарушения *RI*-целостности: для этого достаточно, чтобы новые значения *FK* соответствовали какому-нибудь студенту из таблицы Студент и какому-нибудь предмету из таблицы Предмет.
- Для промежуточных таблиц МП-отношения – это изменение в существующей строке значения *FK*. Например, при переводе некоторого студента в группу, обучающуюся на другой специальности, путём изменения значения *FK* UR3 *RI*-целостность сохранится, но МП-целостность будет нарушена.

Предлагаются два подхода к обеспечению МП-целостности в этом случае: с помощью табличных триггеров или с помощью избыточных *FK*.

3.1 Обеспечение МП-целостности на основе триггеров базы данных

Этот подход предполагает создание БД-триггеров, контролирующих ситуации нарушения МП-целостности и блокирующих соответствующие операции обработки данных. Это BEFORE-триггеры, срабатывающие перед исполнением операции.

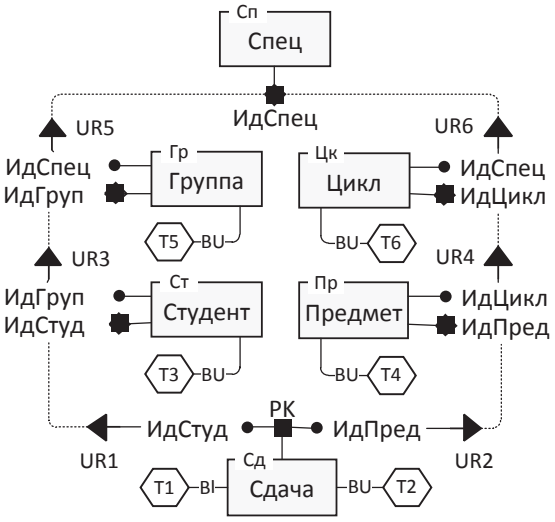


Рисунок 3 – Пример использования суррогатных ключей и триггеров для обеспечения МП-целостности

На рисунке 3 приведён пример РМ на основе *SK* (см. рисунок 2б), дополненной триггерами для обеспечения МП-целостности. Триггеры обозначены шестиугольниками, тип триггера указан в разрыве линии, соединяющей его с родительской таблицей:

- BI – триггер T1 BEFORE INSERT, привязан к таблице Сдача;
- BU – триггеры T2–T6 BEFORE UPDATE, привязаны к таблицам Сдача, Студент, Предмет, Группа и Цикл соответственно.

Триггер T1 срабатывает перед вставкой новой строки в таблицу Сд (Сдача), соответствующую МП-потомку. Процедура триггера, основываясь на вставляемых значениях столбцов-атрибутов сдачи ИдСтуд и ИдПред, должна определить значения атрибутов ИдСпец по разным ЛВР (т.е. соответствующие сдававшему студенту и сданному предмету) и в случае их несовпадения выдать команду блокирования операции вставки.

Триггер T2 срабатывает перед изменением строки в МП-потомке в таблице Сд. Процедура триггера должна выполнить те же действия, что и в триггере T1, т.е. для изменяемых значений атрибутов сдачи ИдСтуд и ИдПред определить значения атрибутов ИдСпец сдававшего студента и сданного предмета и при несовпадении блокировать операцию обновления.

Триггеры T3–T6 срабатывают перед изменением строки в промежуточных таблицах МП-отношения (в таблицах Ст, Пр, Гр и Цк соответственно). Особенность этих таблиц в том, что модифицируемой строке в общем случае может соответствовать ноль, одна или несколько дочерних строк в МП-потомке (в таблице Сд). Поэтому процедуры этих триггеров должны

отыскивать строки, являющиеся МП-потомками модифицируемой строки, и проверять для них МП-целостность (отсутствие сдачи студентами «чужих» предметов), в противном случае блокировать модификацию.

Языки процедур для триггеров различаются в различных реляционных СУБД. В данном случае используется язык *PL/SQL* (диалект *MySQL* 8.0 / *MariaDB* 10).

Программный код триггера T1 (см. листинг 1) создаёт триггер BEFORE INSERT с именем T1, привязанный к таблице Сдача (строка 1). Опция FOR EACH ROW (строка 2) задаёт выполнение процедуры триггера для каждой вставляемой строки («строчный» триггер). Объявляются две целочисленные переменные (строка 3). В переменную ИдСпецСтуд заносится значение идентификатора специальности сдавшего студента, извлечённое по ЛВР Сд → Ст → Гр → Сп командой SELECT (строки 4–7), а в переменную ИдСпецПред – значение идентификатора специальности сданного предмета, извлечённое аналогичным образом по ЛВР Сд → Пр → Цк → Сп (строки 8–11). Значения этих переменных сравниваются (строка 11) и в случае неравенства возбуждается исключительное состояние SQLSTATE '45000', которое отменяет вставку новой строки в таблицу и выдаёт сообщение MESSAGE_TEXT (строка 13).

Листинг 1 – Программный код INSERT-триггера для контроля вставки в таблицу Сдача

```
 1  CREATE TRIGGER T1 BEFORE INSERT ON Сдача
 2    FOR EACH ROW BEGIN
 3      DECLARE ИдСпецСтуд, ИдСпецПред INT;
 4      SET ИдСпецСтуд = (
 5          SELECT Спец.ИдСпец FROM Группа NATURAL JOIN Студент
 6          WHERE Студент.ИдСтуд = NEW.ИдСтуд );
 7      SET ИдСпецПред = (
 8          SELECT Спец.ИдСпец
 9          FROM Спец NATURAL JOIN Цикл NATURAL JOIN Предмет
10          WHERE Предмет.ИдПред = NEW.ИдПред );
11      IF  ИдСпецСтуд != ИдСпецПред THEN
12        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
13          "Триггер T1: ОТМЕНА вставки — нарушение МР-целостности «Сдача–•Спец»";
14      END IF;
15    END;
```

Программный код триггера T2 имеет аналогичный вид с той разницей, что опция INSERT заменена на опцию UPDATE, «T1» – на «T2», «вставки» – на «обновления».

Программный код триггера T3 (см. листинг 2) содержит две числовые переменные (ИдСпецСтуд и ИдСпецПред, строка 3). Дальнейшая обработка происходит при условии, что изменению в строке таблицы Студент подвергается значение *FK*, ссылающегося на таблицу Группа (строка 4), причём у соответствующего студента имеются дочерние строки в таблице Сдача (строка 5). В случае выполнения этого условия в переменную ИдСпецСтуд заносится значение идентификатора специальности сдавшего студента, извлечённое по ЛВР Ст → Гр → Сп командой SELECT (строки 7–8), а в переменную ИдСпецПред – значение идентификатора специальности сданного предмета, извлечённое аналогичным образом по пути Ст →» Сд → Пр → Цк → Сп (строки 9–11). Поскольку участок Ст →» Сд может содержать расщепление, команда SELECT (строка 10) содержит опцию DISTINCT, устраняющую дубликаты результирующего идентификатора специальности. Значения переменных ИдСпецСтуд и ИдСпецПред сравниваются (строка 12), и в случае неравенства возбуждается исключительное состояние, которое отменяет обновление, и выдаётся соответствующее сообщение (строка 13).

Листинг 2 – Программный код UPDATE-триггера для промежуточной таблицы

```
 1  CREATE TRIGGER T3 BEFORE UPDATE ON Студент
 2    FOR EACH ROW BEGIN
 3      DECLARE ИдСпецСтуд, ИдСпецПред INT;
 4      IF  NEW.ИдГруп != OLD.ИдГруп AND EXISTS (
 5        SELECT * FROM Студент NATURAL JOIN Сдача WHERE Студент.ИдСтуд = NEW.ИдСтуд )
 6      THEN
 7        SET ИдСпецСтуд = (
 8          SELECT Группа.ИдСпец FROM Группа WHERE Группа.ИдГруп = NEW.ИдГруп );
 9        SET ИдСпецПред = (
10          SELECT DISTINCT Цикл.ИдСпец FROM Сдача NATURAL JOIN Предмет NATURAL JOIN Цикл
11          WHERE Сдача.ИдСтуд = NEW.ИдСтуд );
12        IF  ИдСпецСтуд != ИдСпецПред  THEN
13          SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
14            "Триггер T3: ОТМЕНА обновления — нарушение МР-целостности «Сдача—•Спец»";
15        END IF;
16      END IF;
17    END;
```

3.2 Обеспечение МП-целостности на основе избыточных внешних ключей

Этот подход предполагает использование системных возможностей по контролю *RI*-целостности для того, чтобы обеспечить МП-целостность. Для этого требуется внесения избыточных атрибутов в состав *FK* так, чтобы иметь возможность проследить *PK* МП-предка вдоль ЛВР до МП-потомка. СУБД поддерживает *RI*-целостность с помощью скрытых системных триггеров, поэтому в данном случае можно воспользоваться этим механизмом, чтобы не вводить пользовательские триггеры. Для этого необходимо продублировать идентификатор МП-предка в составе всех *FK* вдоль ЛВР. Это позволит контролировать идентификатор МП-предка вдоль всей ЛВР вплоть до МП-потомка. Важно, чтобы для таких *FK* действовало каскадное поддержание ссылочной целостности: любое допустимое изменение идентификатора МП-предка в промежуточных таблицах должно передаваться в дочерние таблицы. В результате для проверки МП-целостности остаётся проконтролировать совпадение этих идентификаторов по разным ЛВР у МП-потомка.

На рисунке 4 приведён пример РМ на основе *SK* (см. рисунок 2б). Идентификатор МП-предка ИдСпец продублирован во всех таблицах, причём в таблице МП-потомка он сделан общим для обеих ЛВР.

FK, задающие цепочки ЛВР, содержат идентификатор МП-предка, для чего в промежуточных таблицах предусмотрены соответствующие *UK* (обозначены тёмными ромбами). Например, уникальный ключ UK1 в таблице Студент задан как

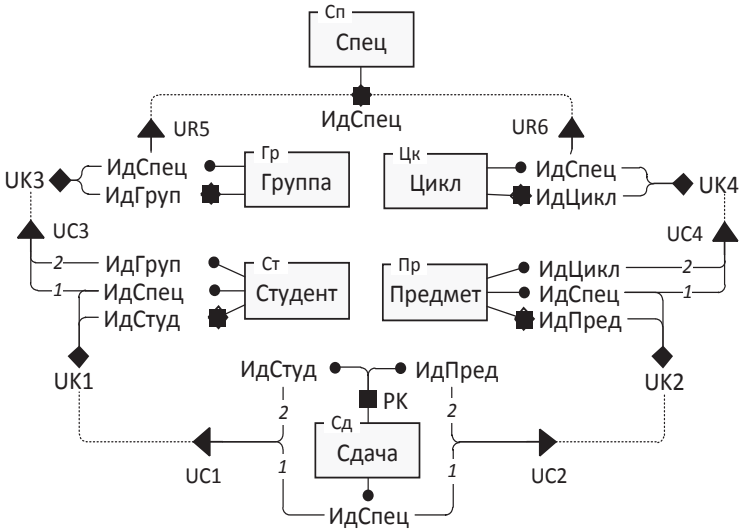


Рисунок 4 – Пример использования суррогатных ключей и избыточных внешних ключей для обеспечения МП-целостности

CONSTRAINT UK1 UNIQUE (ИдСпец, ИдСтуд).

На него из таблицы Сдача ссылается *FK* UC1, заданный как

CONSTRAINT UC1 FOREIGN KEY (ИдСпец, ИдГруп) REFERENCES Группа (ИдСпец, ИдГруп) ON UPDATE CASCADE.

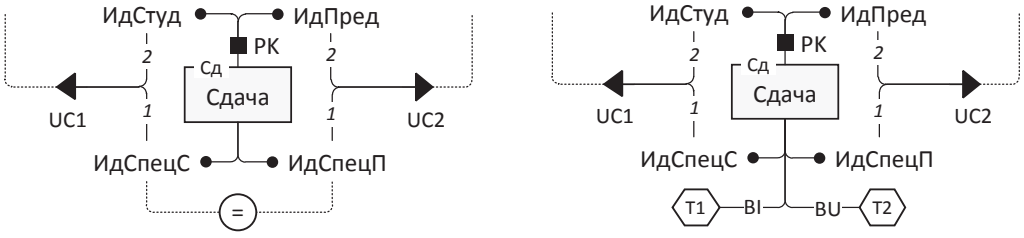
Здесь *RI*-правило ON UPDATE CASCADE обеспечивает автоматическую коррекцию значения ИдСпец в таблице Сдача при изменении ИдСпец в таблице Студент. В результате этого изменённое значение ИдСпец в таблице Группа (например, при переводе студента в группу, которая соответствует другой специальности) корректируется в строках таблицы Студент, которые соответствуют данной группе. Это изменение аналогичным образом переходит в таблицу Сдача, где блокируется вследствие нарушения ссылочной целостности по *FK* UC2 (поскольку изменение не касается ЛВР Сд → Пр → Цк → Сп).

В рассмотренном случае при нарушении МП-целостности происходит нарушение ссылочной целостности, о чём сообщается пользователю. Это может дезориентировать пользователя и затруднить понимание ситуации. Кроме того, этот подход неприменим для случая отрицательной МП-целостности, когда требуется обеспечить различие МП-предков для разных ЛВР. Более удобно использовать в МП-потомке разные имена идентификаторов МП-предка для различных ЛВР с последующим явным сравнением их значений.

На рисунке 5 этот подход иллюстрируется на примере рассмотренной модели (см. рисунок 4). Здесь в таблице МП-потомка Сдача предусмотрено два идентификатора МП-предка: ИдСпецС и ИдСпецП. Значение ИдСпецС каскадно наследуется через ЛВР Сд→Ст→Гр→Сп, а значение ИдСпецП – через ЛВР Сд→Пр→Цк→Сп. Сравнение этих значений может быть выполнено двумя способами: с помощью ограничения CHECK, заданного в таблице Сдача (рисунок 5а), или с помощью BI/BU-триггеров, прикреплённых к таблице (рисунок 5б). При использовании первого способа в определение таблицы вводится ограничение целостности

CONSTRAINT CH1 CHECK (ИдСпецС = ИдСтудП).

Сообщение о нарушении данного ограничения однозначно укажет на нарушение МП-целостности. Этот способ работает не во всех СУБД. Например, если для *ProgreSQL* 15 он успешно применим, то текущие версии *MySQL* 8 / *MariaDB* 10 не допускают использование *FK* в условии действия ограничения CHECK.



а) б)

Рисунок 5 – Пример использования различных идентификаторов МП-предка в МП-потомке: а) – на основе условия CHECK; б) – на основе триггеров

В подобных случаях проверку совпадения / несовпадения идентификаторов МП-предка можно возложить на триггеры. В листинге 3 представлен программный код INSERT-триггера T1, который сравнивает новые значения ИдСпецС и ИдСпецП и блокирует операцию в случае несовпадения. Программный код UPDATE-триггера имеет аналогичный вид.

Листинг 3 – Программный код INSERT-триггера для МП-потомка Сдача

```
1   CREATE TRIGGER T1 BEFORE INSERT ON Студент
2     FOR EACH ROW BEGIN
```

```
3          IF  NEW.ИдСпецС != NEW.ИдСпецП  THEN
4            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
5              "Триггер T1: ОТМЕНА вставки — нарушение МР-целостности «Сдача–•Спец»";
6          END IF;
7      END;
```

4 МП-целостность при использовании натуральных ключей

Реляционная БД, основанная на *NK*, в отличие от БД, основанной на *SK*, может иметь идентифицирующие связи между таблицами, и значения ключей могут изменяться в течение ЖЦ строк таблицы. Идентифицирующие связи означают, что экземпляры дочерних сущностей идентифицируются в контексте своих родителей. На концептуальной модели это обстоятельство отражено с помощью тёмных квадратов в символах связи. На рисунке 6а приведён пример СС-модели, в которой все связи идентифицирующие, т.е. все сущности (кроме сущности Спец) идентифицируются в контексте своих родителей (студенческие группы локально идентифицируются в пределах своей специальности, студенты – в пределах группы и т.д.). На рисунке 6б приведена соответствующая РМ, где имена натуральных ключевых атрибутов имеют префикс «Код». *PK* (кроме МП-предка) являются составными и включают в качестве компонент *PK* таблиц-родителей. *FK* (кроме ссылки на МП-предка) также являются составными. В результате идентификатор МП-предка присутствует в составе *PK* своих потомков, что позволяет использовать подход к МП-целостности, аналогичный подходу с избыточными *FK* (см. рисунок 4).

Изменчивость *NK* в течение ЖЦ порождает необходимость поддержания каскадной ссылочной целостности, но не создаёт непосредственной угрозы для МП-целостности. Например, изменение идентификатора КодГруппы в таблице Группа (см. рисунок 6б) приводит к необходимости каскадного обновления одноимённых атрибутов в *FK* дочерних таблиц Студент и Сдача, что сохраняет МП-целостность. Нарушение МП-целостности может возникнуть при изменении *FK*, например, при переводе студента на другую специальность. Таким образом, в этом случае ситуация та же, что и при неизменных *SK*.

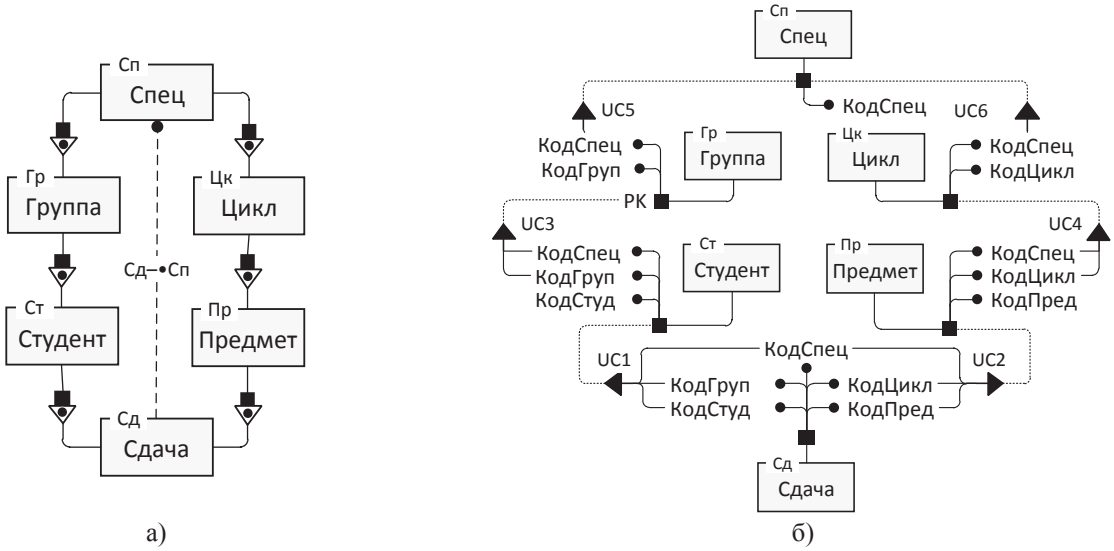


Рисунок 6 – Пример моделей с идентифицирующими связями:
а) – СС-модель; б) – реляционная модель на основе натуральных ключей

Для обеспечения каскадной ссылочной целостности в этих случаях удобно использовать *FK* с опцией ON UPDATE CASCADE. Например, если *PK* в таблице Студент задан как CONSTRAINT PK_Ст PRIMARY KEY (КодСпец, КодГруп, КодСтуд), то на него из таблицы Сдача ссылается *FK* UC1, заданный как CONSTRAINT UC1 FOREIGN KEY (КодСпец, КодГруп, КодСтуд) REFERENCES Группа (КодСпец, КодГруп, КодСтуд) ON UPDATE CASCADE. В результате этого все изменения компонентов *PK* в таблице Студент автоматически передаются в *FK* UC1 таблицы Сдача.

Каскадное обновление ключей от МП-предка к МП-потомку по параллельным ЛВР может быть затруднено из-за технических ограничений СУБД. Так, модель на основе общего идентификатора МП-предка у МП-потомка, представленная на рисунке 6б, успешно работает в среде *PostgreSQL* 10. В среде *MySQL* 8 / *MariaDB* 10 таким способом не удаётся выполнить каскадное обновление *PK* КодСпец МП-предка (в таблице Спец) из-за несинхронного обновления по разным ЛВР. В среде *MS SQL Server* 2022 запрещено каскадное обновление при наличии параллельных ЛВР – возникает ошибка на этапе компиляции. В среде *Oracle Database* автоматическое каскадное обновление не предусмотрено, и его необходимо поддерживать с помощью триггеров. В этих условиях можно перейти к схеме с раздельными идентификаторами МП-предка в МП-потомке, что позволяет выполнить каскадное обновление идентификатора МП-предка. Для контроля МП-целостности в этом случае потребуется использование триггеров (см. рисунок 5б).

Применительно к СУБД *MySQL* 8 / *MariaDB* 10 (см. рисунок 7) МП-потомок Сдача содержит два идентификатора МП-предка Спец (КодСпецС и КодСпецП), унаследованные по разным ЛВР (см. рисунок 5б). Это обеспечивает их корректное каскадное обновление ON UPDATE CASCADE через *FK* UC1–UC6.

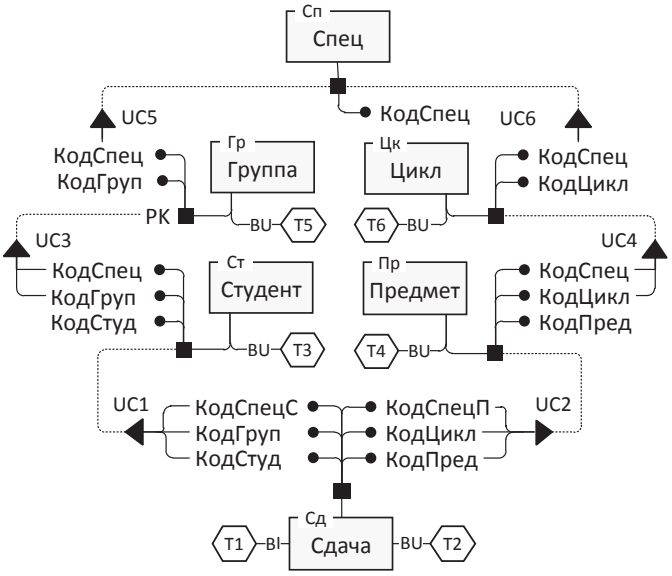


Рисунок 7 – Различные идентификаторы МР-предка в МР-потомке в случае натуральных ключей и идентифицирующих связей

Триггеры Т1–Т6 обеспечивают контроль МП-целостности. Триггеры Т1 и Т2 контролируют равенство значений КодСпецС и КодСпецП в таблице Сдача при вставке и обновлении строк (см. листинг 3). Триггеры Т3–Т6 контролируют то же самое при обновлении строк в промежуточных таблицах ЛВР (их необходимость вызвана тем, что в СУБД *MySQL*/*MariaDB* каскадное обновление таблиц не запускает триггеры, привязанные к этим таблицам).

В качестве примера рассмотрен триггер Т3 (см. листинг 4), который проверяет, не приведёт ли изменение в таблице Студент в результате последующего каскадного обновления к нарушению равенства значений КодСпецС и КодСпецП в таблице Сдача (строки 3–7), и в этом случае отменяет операцию обновления с выдачей соответствующего сообщения (строки 8–10). Наличие в таблице Сдача ключевых атрибутов КодГруп и КодСтуд упрощает эту проверку с помощью оператора SELECT, который выполняет выборку из таблицы Сдача строк, являющихся дочерними для обновляемой строки таблицы и содержащими при этом различающиеся значения КодСпецС и КодСпецП. Триггеры Т4–Т6 построены аналогичным образом.

Листинг 4 – Программный код триггера Т3 (см. рисунок 7) в среде *MySQL/MariaDB*

```
CREATE TRIGGER Т3 BEFORE UPDATE ON Студент
  FOR EACH ROW BEGIN
    IF  ( NEW.КодСпец != OLD.КодСпец OR NEW.КодГруп != OLD.КодГруп
        OR NEW.КодСтуд != OLD.КодСтуд ) AND EXISTS (
          SELECT * FROM Сдача WHERE Сдача.КодСпецС = OLD.КодСпец
            AND Сдача.КодГруп = OLD.КодГруп AND Сдача.КодСтуд = OLD.КодСтуд
            AND NEW.КодСпец != Сдача.КодСпецП )
    THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT =
      "Триггер Т3: ОТМЕНА обновления — нарушение МР-целостности «Сдача—•Спец»";
    END IF;
  END;
```

Заключение

Понятие МП-целостности распространено здесь на РМ логического уровня с целью практического применения в реляционных БД. В реляционной среде универсальным подходом к обеспечению МП-целостности является подход на основе триггеров БД, обнаруживающих и блокирующих операции вставки и обновления строк в таблицах, которые ведут к нарушению МП-целостности. Недостатком этого подхода является необходимость процедурного программирования триггеров, а также сложность переноса БД между СУБД разного вида из-за различий языков процедурного программирования.

В ряде случаев использование стандартных возможностей СУБД по поддержанию ссылочной целостности позволяет дополнительно обеспечить МП-целостность и решить задачу на декларативном уровне без использования триггеров (работают скрытые системные триггеры, с помощью которых СУБД поддерживает ссылочную целостность). Недостатком этого является зависимость конкретных решений от использования простых/составных, натуральных/суррогатных ключей, а также от технических ограничений, имеющихся у СУБД. Кроме того, может потребоваться введение избыточных компонентов в состав ключей для отслеживания экземпляра МП-предка вдоль линий восходящего родства.

Приведённые РМ и программный код созданы в виде тестовых примеров и проверены в средах СУБД (полностью в *MySQL* и *MariaDB*, частично, где это позволяла функциональность, в *PostgreSQL*, *MS SQL Server*, *Oracle Database*).

Список источников

[1] ***Storey V.C. et al***. An ontology-based expert system for database design. *Data and Knowledge Engineering*, 1998. vol.28, no.1. P.31-46. DOI: 10.1016/S0169-023X(98)00012-3. EDN: ABKHUB.

[2] ***Sugumaran V., Storey V.*** Supporting database designers in entity-relationship modeling: An ontology-based approach. *ICIS 2003 Proceedings*. 2003. P.59-71. https://aisel.aisnet.org/icis2003/6.

[3] ***Chujai P., Kerdprasop N., Kerdprasop K.*** On transforming the ER model to ontology using Protégé OWL tool. *Int. J. Computer Theory and Engineering*. 2014. Vol.6, no.6. P.484-489. DOI: 10.7763/IJCTE.2014.V6.914.

[4] ***Benjamin P.C. et al.*** IDEF5 Method Report. Prepared for Armstrong Laboratory AL/HRGA. Knowledge Based Systems Inc. Sept. 21, 1994. 187 p. https://online-pmo.com/wp-content/Education/Idef5.pdf.

[5] ***Миронов В.В., Миронов К.В.*** Концептуально-онтологические аспекты множественности предка в информационных моделях «сущность-связь». *Онтология проектирования*. 2024. Т.14, № 4(54). С.493-503. DOI: 10.18287/2223-9537-2024-14-4-493-503. EDN: HHPGHD.

[6] ***Jardine D.A.*** The ANSI/SPARC DBMS Model. North-Holland Pub. Co., 1977. 225 p.

[7] ***Peter Pin-Shan Chen.*** The Entity-Relationship Model – toward a unified view of data. *ACM Transactions on Database Systems*. 1976. Vol.1. P.9–36. DOI: 10.1145/320434.320440.

[8] ***Мартин Дж.*** Организация баз данных в вычислительных системах. Пер. с англ. М.: Мир, 1978. 616 с.

[9] ***Дейт К.Дж.*** Введение в системы баз данных = Introduction to Database Systems. 8-е изд. М.: Вильямс, 2006. 1328 с. ISBN 5-8459-0788-8.
[10] ***Wildman W.J.*** An introduction to relational ontology. *The Trinity and an Entangled World: Relationality in Physical Science and Theology*. 2010. P.55-73.
[11] ***Zhao T.*** Introduction to Hiromatsu Wataru's "relational ontology". *Region – Educational Research and Reviews*. 2024. 6. 111. DOI: 10.32629/rerr.v6i5.2152.
[12] ***Боргест Н.М.*** Системный и онтологический анализы: схожесть и различие понятий. *Онтология проектирования*. 2024. Т.14, № 1(51). С.9-28. DOI: 10.18287/2223-9537-2024-14-1-9-28. EDN: KRGWSR.
[13] ***Семенова В.А., Смирнов С.В.*** Модели и методы онтологического анализа данных в задаче структурного анализа и синтеза технических решений. *Онтология проектирования*. 2023. Т.13, №4(50). С.531-547. DOI: 10.18287/2223-9537-2023-13-4-531-547. EDN: UADWCT.
[14] ***Семенова В.А., Смирнов С.В.*** Механизм нормализации эмпирического контекста в онтологическом анализе данных. *СИИТ*. 2021. Т.3, №3(7). С.45-52. DOI: 10.54708/26585014_2021_33745. EDN: QXRTXB.
[15] ***Богданова Д.Р., Шахмаметова Г.Р., Ниязгулов А.М.*** Структура информационного хранилища системы поддержки принятия клинических решений. *Онтология проектирования*. 2024. Т.14, № 2(52). С.270-278. DOI: 10.18287/2223-9537-2024-14-2-270-278. EDN: CAYCVK.
[16] ***Смирнов С.В.*** Онтологии как смысловые модели. *Онтология проектирования*. 2013. № 2(8). С.13-19.

Сведения об авторах

Миронов Валерий Викторович, 1952 г.р., профессор кафедры автоматизированных систем управления УУНиТ. Диплом радиофизика (Воронежский государственный университет, 1975). Д.т.н. (1995). Труды в области ситуационного управления, иерархических моделей и баз данных. ORCID: 0000-0002-0550-4676. AuthorID (РИНЦ): 691759. Author ID (Scopus): 57192962687. Researcher ID (WoS): AAB-9377-2022. *mironov@list.ru* ✉

Миронов Константин Валерьевич, 1991 г.р., доцент кафедры вычислительной техники и защиты информации УУНиТ. Диплом специалиста по защите информации (Уфимский государственный авиационный технический университет, 2012). Степень PhD Технического университета Вены (2015). Труды в области робототехники и искусственного интеллекта, обработки и защиты информации. ORCID: 0000-0002-4828-1345. AuthorID (РИНЦ): 939814. Author ID (Scopus): 56732791500. *mironovconst@gmail.com.*

Поступила в редакцию 08.07.2025, после рецензирования 16.08.2025. Принята к публикации 25.08.2025.

Scientific article DOI: 10.18287/2223-9537-2025-15-4-552-565

AM-integrity in designing relational database models

© 2025, V.V. Mironov ✉, K.V. Mironov

Ufa University of Science and Technologies, Ufa, Russia

Abstract

The article discusses a special type of integrity in databases – ancestral multiplicity integrity (AM-integrity), a concept introduced by the authors at the conceptual–ontological level within "entity–relationship" models. In this paper, this concept is extended to the relational model for the purpose of practical application in creating databases. Related notions are examined, including the lineage of ascending ancestry, the relation of ancestral multiplicity, and the corresponding integrity constraint. Transitioning to the relational model requires consideration of primary and foreign keys, referential integrity, and table triggers. As a universal means of ensuring integrity in the relational environment, the paper proposes an approach based on database triggers. These triggers detect and block insert and update operations that would lead to violations of integrity. The need for procedural programming of triggers is noted, along with the challenges of transferring data between different database management systems due to variations in procedural programming languages. An alternative approach is suggested, based on leveraging built-in mechanisms for maintaining referential integrity, in which ancestral multiplicity integrity is ensured as part of referential integrity. The specific solution depends on the use of simple or composite, natural or surrogate keys, as well as on the technical limitations of the database implementation environment. In some cases, redundant key components may need to be introduced to track ancestor instances along ancestral lineages. Test examples of relational models are provided for different relational systems (complete implementations in MySQL and MariaDB, and partial implementations in PostgreSQL, MS SQL Server, and Oracle Database).

Keywords: *entity-relationship model, relational model, ancestral multiplicity, lineage of ascending ancestry, database triggers, foreign keys, referential integrity.*

For citation: *Mironov VV, Mironov KV.* AM-integrity in designing relational database models [In Russian]. *Ontology of designing*. 2025; 15(4): 552-565. DOI: 10.18287/2223-9537-2025-15-4-552-565.

Authors' contributions: *Mironov V.V.* – idea, conceptual framework, formalization; *Mironov K.V.* – models, algorithms, programming, and debugging across different environments.

Conflict of interest: The authors declare no conflict of interest.

List of figures and listings

References

[1] ***Storey VC. et al***. An ontology-based expert system for database design. *Data and Knowledge Engineering*, 1998; 28(1): 31-46. DOI: 10.1016/S0169-023X(98)00012-3. EDN: ABKHUB.

[2] ***Sugumaran V, Storey V.*** Supporting database designers in entity-relationship modeling: An ontology-based approach. In: *ICIS 2003 Proceedings*. 2003. P.59-71. https://aisel.aisnet.org/icis2003/6.
[3] ***Chujai P, Kerdprasop N, Kerdprasop K.*** On transforming the ER model to ontology using Protégé OWL tool. *Int. J. Computer Theory and Engineering*. 2014; 6(6): 484-489. DOI: 10.7763/IJCTE.2014.V6.914.
[4] ***Benjamin PC. et al.*** IDEF5 Method Report. Prepared for Armstrong Laboratory AL/HRGA. Knowledge Based Systems Inc. Sept. 21, 1994. 187 p. https://online-pmo.com/wp-content/Education/Idef5.pdf.
[5] ***Mironov VV, Mironov KV.*** Conceptual and ontological aspects of the plurality of ancestors in "entity-relational" information models [In Russian]. *Ontology of Designing*. 2024; 14(4): 493-503. DOI: 10.18287/2223-9537-2024-14-4-493-503. EDN: HHPGHD.
[6] ***Jardine DA.*** The ANSI/SPARC DBMS Model. North-Holland Pub. Co., 1977. 225 p.
[7] ***Peter Pin-Shan Chen.*** The Entity-Relationship Model — toward a unified view of data. *ACM Transactions on Database Systems*. 1976; 1: 9–36. DOI: 10.1145/320434.320440.
[8] ***Martin J.*** Organization of Databases in Computing Systems [In Russian]. Moscow: Mir, 1978. 616 p.
[9] ***Date KJ.*** Introduction to Database Systems [In Russian]. 8th ed. Moscow: Williams, 2006. 1328 p.
[10] ***Wildman WJ.*** An introduction to relational ontology. *The Trinity and an Entangled World: Relationality in Physical Science and Theology*. 2010. P.55-73.
[11] ***Zhao T.*** Introduction to Hiromatsu Wataru's "relational ontology". *Region – Educational Research and Reviews*. 2024. 6. 111. DOI: 10.32629/rerr.v6i5.2152.
[12] ***Borgest NM.*** Systems and ontological analysis: similarities and differences between the concepts [In Russian]. *Ontology of Designing*. 2024; 14(1): 9-28. DOI: 10.18287/2223-9537-2024-14-1-9-28. EDN: KRGWSR.
[13] ***Semenova VA, Smirnov SV.*** Models and methods of ontological data analysis in the problem of structural analysis and synthesis of technical solutions [In Russian]. *Ontology of Designing*. 2023; 13(4): 531-547. DOI: 10.18287/2223-9537-2023-13-4-531-547. EDN: UADWCT.
[14] ***Semenova VA, Smirnov SV.*** The mechanism of normalization of empirical context in the ontological analysis of data. *SIIT*. 2021; 3(3): 45-52. DOI: 10.54708/26585014_2021_33745. EDN: QXRTXB.
[15] ***Bogdanova DR, Shakhmametova GR, Niyazgulov AM.*** The structure of the CDSS information repository based on the ontological approach [In Russian]. *Ontology of designing*. 2024; 14(2): 270-278. DOI: 10.18287/2223-9537-2024-14-2-270-278. EDN: CAYCVK.
[16] ***Smirnov SV.*** Ontologies as semantic models [In Russian]. *Ontology of designing*. 2013; 2(8): 13-19.

About the authors

Valeriy Viktorovich Mironov (b. 1952). Professor of the Department of Automated Control Systems at UUST. Diploma in Radiophysics (Voronezh State University, 1975), Dr. Tech. Sci. (Ufa State Aviation Technical University, 1995). Works in the field of situational management, hierarchical models and databases. ORCID: 0000-0002-0550-4676. Author ID (RSCI): 691759. Author ID (Scopus): 57192962687. ResearcherID (WoS): AAB-9377-2022. *mironov@list.ru* ✉

Konstantin Valerievich Mironov (b. 1991). Associate Professor of the Department of Computer Science and Information Security at UUST, Certified Specialist in Information Security (Ufa State Aviation Technical University, 2012), PhD (Technical University of Vienna, 2015). Works in the field of robotics and artificial intelligence, information processing and security. ORCID: 0000-0002-4828-1345. AuthorID (RSCI): 939814. Author ID (Scopus): 56732791500. *mironovconst@gmail.com*

Received July 8, 2025. Revised August 16, 2025. Accepted August 25, 2025.