Оценка распространения искажений в Data-продуктах реляционной архитектуры Data Mesh

С.Е. Духовенский

***Аннотация* — При децентрализованном подходе к управлению данными Data Mesh взаимодействие с данными осуществляется на уровне отдельных продуктов. Если при подготовке Data-продуктов возникли ошибки или были использованы некачественные данные, то их распространение может существенно увеличиться в репликах, витринах, сервисах и др. Работа посвящена оценке качества Data-продукта в реляционной архитектуре Data Mesh в условиях выявленных искажений входных данных. Предложен подход к построению такой оценки, опирающийся на представление трансформаций данных внутри Data-продукта в виде цепочки элементарных операций реляционной алгебры, таких как расширенная проекция, фильтрация и декартово произведение, и анализа распространения искажений входных данных в результате применения указанных операций. Представлена концепция «разметки» исходного отношения, позволяющая разделить кортежи на типы в зависимости от характера выявленных в них искажений, а также алгоритмы, направленные на отслеживание преобразования кортежей разных типов в результате применения перечисленных выше элементарных операций.**

Разработан комплексный алгоритм, реализующий решение рассматриваемой задачи оценки качества Data-продукта. В рамках тестирования подготовлен экспериментальный стенд на наборе открытых данных Open University и реализован представленный алгоритм. Результаты эксперимента подтвердили, что алгоритм оценки качества Data-продукта может быть успешно использован в контексте реляционной архитектуры Data Mesh. В заключении предложено несколько вариантов практического применения такой оценки для эффективного управления качеством данных в рамках концепции Data Mesh.

***Ключевые слова* — качество данных, Data Mesh, Data-продукт, реляционная алгебра.**

I. Введение

Ответом на рост объема цифровых данных является внедрение практик эффективного управления данными многими современными организации. Традиционные централизованные архитектуры данных, основанные на хранилищах и озерах данных, имеют ряд ограничений [1, 2]. В частности, доменные команды, обладающие наибольшей компетенцией в предметной области, вынуждены полагаться на центральную группу в части сбора, обработки и управления данными. Как следствие, центральная группа становится «бутылочным горлышком», препятствующим раскрытию ценности данных предметной области. Кроме того, в отсутствие четкого понимания, кто является владельцем данных, появляется неопределенность в распределении ответственности за их качество, что приводит к потенциальным проблемам в данных.

Data Mesh – это децентрализованный социотехнический подход к управлению данными в больших организациях, призванный решить проблемы, свойственные традиционным архитектурам данных [3]. Автор подхода выделяет четыре основных принципа реализации Data Mesh: данные как продукт, владение данными в рамках домена, платформа самообслуживания данных и федеративное управление. Более детально, доменные команды создают и управляют Data-продуктами, позволяющими потребителям данных эффективно обнаруживать и использовать данные домена. Домены внутри Data Mesh могут также потреблять Data-продукты других доменов, причём за обеспечение совместимости Data-продуктов и их соответствие внутренним и внешним правилам отвечает федеративная команда управления. Наконец, платформа самообслуживания данных поддерживает домены («узлы»), предлагая инфраструктурные и платформенные услуги, необходимые для создания, поддержки, управления Data-продуктами и связывания их в единую «сеть».

Важно отметить, что вопрос эффективного управления качеством данных пронизывает всю концепцию Data Mesh. С одной стороны, запрос на повышение качества данных является одним из ключевых мотивационных аспектов для перехода к Data Mesh [1]. С другой стороны, элементы управления качеством данных встречаются во всех четырех принципах реализации Data Mesh [2]. В частности, разработка методологии для выявления и исправления некачественных данных является одной из ключевых задач федеративной команды управления Data Mesh, в то время как доменные команды, ответственные за обеспечение высокого качества своих Data-продуктов, реализуют меры в соответствии с разработанной методологией.

С учётом вышесказанного задача оценки качества Data-продуктов имеет особенную прикладную ценность как для федеративной команды управления Data Mesh, так и для отдельных доменных команд. На практике для решения указанной задачи рекомендуется использование Data-контрактов, суть которых заключается в описании требований к части данных Data-продукта, ис-

Статья получена 21 апреля 2026.

Духовенский С.Е., аспирант РТУ МИРЭА, Москва, Россия (dukhovenskiy.s.e@edu.mirea.ru).

пользуемой другими доменами [4]. Такой подход хорошо согласуется с принципом «fitness-for-use», согласно которому качество данных непосредственно связывается с их назначением и способами использования [5], однако может быть сильно ограничен краткосрочными потребностями доменов и не учитывать расширение Data Mesh в долгосрочной перспективе. Кроме того, оценка качества данных на основе Data-контрактов не принимает во внимание знания доменной команды относительно искажений входных данных.

Целью исследования является выработка методики оценки качества Data-продуктов в реляционной архитектуре Data Mesh с учётом наличия сведений об искажениях входных данных, а также разработка соответствующего алгоритма отслеживания распространения искажений в условиях, когда набор данных в выходных портах можно представить в виде SQL трансформаций входного набора, состоящих из базовых операций вида: расширенная проекция, фильтрация и декартово произведение. Работа опирается на исследования [6, 7] в области оценки качества данных в реляционных базах данных, а также обобщает результаты, полученные в [8]. В частности, получены детерминированные правила отслеживания искажений в случае фильтрации по булевому атрибуту, декартова произведения и удаления атрибута, а для операции добавления атрибута представлены формулы, позволяющие оценить вероятность наличия искажений в новых значениях при наличии выявленных мониторингом качества данных искажений в исходных атрибутах.

Для достижения указанной цели исследования решались следующие задачи:

- математическая формализация задачи оценки качества Data-продукта в реляционной структуре Data Mesh с учётом наличия сведений об искажениях входных данных;
- разработка программно-алгоритмического обеспечения, позволяющего определить распространение различных видов искажений в базовых SQL операциях и рассчитать оценку качества выходного набора данных;
- реализация экспериментального стенда на основе открытого набора данных Open University [9] для тестирования предложенного алгоритма;
- анализ результатов эксперимента.

Научная новизна работы заключается в приложении исследованных ранее механизмов распространения искажений в реляционных базах данных к задаче определения качества данных в концепции Data Mesh. В отличие от основной части актуальных исследований по соответствующей тематике, работа сфокусирована на математической формализации задачи оценки качества Data-продуктов и базируется на декомпозиции процессов обработки данных до уровня элементарных операций реляционной алгебры, что позволяет отследить динамику искажений в процессе их трансформации от входных портов до выходных. Это даёт возможность перейти от статического мониторинга выходных данных с помощью Data-контрактов, на который ориентировано большинство современных инструментов качества данных, к аналитическому подходу оценки качества. Такой подход может быть полезен организациям, использующим концепцию Data Mesh, поскольку позволяет доменным командам гарантировать требуемый уровень качества на основе прозрачного отслеживания распространения искажений, что в целом способствует превентивному управлению качеством в единой «сети».

II. Постановка задачи

A. *Основные определения*

Для дальнейшего изложения в работе использована система обозначений, предлагаемая авторами [10] и [11] для описания отношений в реляционной модели данных и операций реляционной алгебры. Для формулирования задачи исследования и описания алгоритмов указанная нотация была дополнена следующими элементами: SQL-запрос в виде цепочки операций реляционной алгебры [8], частные случаи операции проекции: исключение атрибута, добавление атрибута вида инъекция и добавление булевого атрибута, а также проверки качества данных и множества выявляемых с их помощью ячеек [12]. Дополненная система обозначений представлена в таблице I.

Для численного определения качества данных в реляционной модели данных используется идея, основанная на сравнении эталонного и искаженного экземпляров некоторого отношения [7]. Пусть дан некоторый материальный объект, информацию о котором можно представить в виде отношения в реляционной модели данных с заголовком (схемой), описываемой набором атрибутов A. Тогда под $R(A)$ будет обозначен *эталонный экземпляр* – непустое реляционное отношение, точно описывающее целевой объект. Также за $\hat{R}(A)$ будет обозначен *искаженный экземпляр* – реляционное отношение, описывающее целевой объект с искажениями, возникающими в результате обработки данных (например, вследствие ошибок ручного ввода, некорректного сбора данных и т.п.). Пусть также для отношения $R(A)$ можно выделить уникальный ключ $A_{key} \subseteq A$, а $A_{nkey} = A$, A_{key}. Тогда с помощью сравнения эталонного и искаженного экземпляров на основе ключа A_{key} кортежи эталона $R(A)$ можно разбить на три множества:

- $T_{similar} = \{t_r \mid \ t_r \in r \wedge \ \exists t_d \in \hat{r} : v(t_r, A) = v(t_d, A)\}$ – множество кортежей, для которых в искаженном экземпляре существуют полностью совпадающие кортежи;
- $T_{different} = \{t_r \mid \ t_r \in r \wedge \ \exists t_d \in \hat{r} :$ $v(t_r, A_{key}) = v(t_d, A_{key}) \wedge v(t_r, A_{nkey}) \neq v(t_d, A_{nkey})\}$ – множество кортежей, для которых в искаженном экземпляре существуют кортежи, совпадающие полностью только по ключу;
- $T_{missed} = \{t_r \mid t_r \in r \wedge \forall t_d \in \hat{r} : v(t_r, A_{key}) = v(t_d, A_{key})\}$ – множество кортежей, для которых в искаженном экземпляре не существует совпадающих по ключу кортежей.

В свою очередь все кортежи искаженного экземпляра $\hat{R}(A)$ также можно разбить на три множества:

- Для каждого эталонного кортежа $t_r \in T_{similar}$ необходимо определить **один** любой кортеж $t_d \in \hat{r}$, для которого выполняется условие $v(t_r, A) = v(t_d, A)$ – множество таких кортежей будет обозначаться как $\hat{T}_{similar}$;
- Для каждого эталонного кортежа $t_r \in T_{different}$ необходимо определить **один** любой кортеж $t_d \in \hat{r}$, для которого выполняется условие $v(t_r, A_{key}) = v(t_d, A_{key}) \wedge v(t_r, A_{nkey}) \neq v(t_d, A_{nkey})$ – множество таких кортежей будет обозначаться как $\hat{T}_{different}$;
- $\hat{T}_{extra} = \left\{ t_d \mid \ t_d \in \hat{r} \wedge \ t_d \notin (\hat{T}_{similar} \bigcup \hat{T}_{different}) \right\}$ – множество кортежей, для которых не нашлось соответствия в эталонном экземпляре.

Обозначив $N_s = \left|T_{similar}\right| = \left|\hat{T}_{similar}\right|$, $N_d = \left|T_{different}\right| = \left|\hat{T}_{different}\right|$, $N_m = \left|T_{missed}\right|$ и $N_e = \left|\hat{T}_{extra}\right|$ (здесь и далее под $\left|M\right|$ будет обозначена мощность множества), можно ввести следующее определение *качества данных искаженного экземпляра* $\hat{R}(A)$ *в сравнении с эталоном* $R(A)$:

$$DQ(\hat{R}, R) = \frac{N_s}{N_s + N_d + N_m + N_e} \quad (1)$$

Очевидно, что $DQ(\hat{R}, R) \in [0;1]$, причём нулевое значение качества данных достигается в случае полного отсутствия в искаженном экземпляре кортежей $\hat{T}_{similar}$ – т.е. все эталонные кортежи были тем или иным образом искажены; при этом значение 1 достигается тогда и только тогда, когда одновременно пусты множества $\hat{T}_{different}$, $\hat{T}_{extra}$ и T_{missed}, т.е. искаженный экземпляр полностью совпадает с эталоном.

Таблица I. Система обозначений

Обозначение	**Расшифровка**
a и $A = a_1, a_2 \ldots a_n$	Соответственно, наименование атрибута и набор уникальных наименований атрибутов
$dom(a)$ и $dom(A)$	Соответственно, множество значений атрибута a и множество значений набора A
$R(A)$ и $r(A)$	Отношение $R(A)$, состоящее из заголовка (схемы) в виде набора атрибутов A и тела $r(A)$
DS	Набор данных, состоящий из нескольких отношений $R_1(A_1), R_2(A_2) \ldots R_m(A_m)$
t	Элемент $r(A)$, являющийся кортежем
(t, a)	Ячейка, получаемая на пересечении кортежа t и атрибута a
$v(t, a)$ и $v(t, A)$	Соответственно, значение ячейки (t, a) и набор значений в кортеже t по набору атрибутов $A = a_1, a_2 \ldots a_n$
$f(v)$ и $f(V)$	Соответственно, функция, применяемая к значению ячейки v, и функция, применяемая к набору значений в нескольких ячейках V внутри одного кортежа
$expr(A)$	Выражение, включающее наименования атрибутов A, константы, арифметические и строковые операторы
$R_1 \times R_2$	Операция декартова произведения отношений R_1 и R_2
$\sigma_C(R)$	Операция фильтрации (выборки) над отношением R по логическому условию C
$\pi_L(R)$	Операция расширенной проекции над отношением R по списку L, где элементы списка представляются в виде $expr(A) \to z$ и z – наименование атрибута, получаемого вследствие вычисления выражения
$\pi_{a \to \varnothing}(R)$	Операция исключения из отношения R атрибута a
$Q(DS)$ и $O_1, O_2 \ldots O_k$	SQL-запрос, выполняемый над набором данных DS и представляющий собой цепочку $O_1, O_2 \ldots O_k$, где каждый элемент O_i означает операцию реляционной алгебры, выполняемую над множеством отношений $DS_{ext} = DS \bigcup \{R \mid R - результат\ O_j \ \forall j < i\}$
$\mu^{\mathrm{F}} = (R, a)$ и $\Xi(\mu^{\mathrm{F}})$	Проверка качества данных для показателя *«Наполненность обязательного атрибута»* (далее – fullness-проверка), которая определяется парой параметров: отношение $R(A)$ и атрибут $a \in A$, – и выявляет соответствующее множество искаженных ячеек $\Xi(\mu^{\mathrm{F}}) = \{(t, a) \mid \ t \in r(A) \wedge v(t, a) = \mathrm{null}\}$
$\mu^{\mathrm{D}} = (R, a, Dom)$ и $\Xi(\mu^{\mathrm{D}})$	Проверка качества данных для показателя *«Семантическая аккуратность данных»* (далее – domain-проверка), которая определяется тройкой параметров: отношение $R(A)$, атрибут $a \in A$ и Dom – множество семантически корректных значений атрибута a, и выявляет соответствующее множество искаженных ячеек $\Xi(\mu^{D}) = \{(t, a) \mid \ t \in r(A) \wedge v(t, a) \notin Dom\}$
$\mu^{\mathrm{C}} = (R, a, A^{sub}, f)$ и $\Xi(\mu^{\mathrm{C}})$	Проверка качества данных для показателя *«Семантическая согласованность»* (далее – consistency-проверка), которая определяется четвёркой параметров: отношение $R(A)$, атрибут $a \in A$, подмножество атрибутов $A^{sub} \subseteq A$ и условие согласованности $f : dom(A^{sub}) \to boolean$, – и выявляет соответствующее множество искаженных ячеек $\Xi(\mu^{C}) = \{(t, a) \mid \ t \in r(A) \wedge f(v(t, A^{sub})) = \mathrm{False}\}$

B. Задача оценки качества Data-продукта

Процесс обработки данных внутри типового Data-продукта в концепции Data Mesh можно разделить на три составляющих [3]:

1. забор (ingest) данных из «входных» портов Data-продукта, которые могут представлять собой массивы данных во внешних информационных системах, данные ручного ввода внутри домена, а также подключения к другим Data-продуктам;
2. трансформация (process) данных, включающее обогащение, соединение, агрегацию и пр.;
3. выдача (serve) данных другим доменам через «выходные» порты Data-продукта.

В случае наличия полной информации об эталонных и искаженных экземплярах входных данных, предложенный выше метод определения качества данных можно было бы применить к Data-продукту, построенному на основе реляционной базы данных, следующим образом. Пусть входные порты Data-продукта представлены некоторым набором искаженных экземпляров $DS^{in} = \hat{R}_1^{in} ... \hat{R}_n^{in}$ и соответствующим ему набором эталонных экземпляров $DS^{in} = R_1^{in} ... R_n^{in}$. Пусть также процесс трансформации данных внутри Data-продукта представлен в виде перечня SQL-запросов $Q_1(DS^{in}) ... Q_k(DS^{in})$, каждый из которых возвращает данные для одного выходного порта. Тогда i-выходной порт будет представлен искаженным экземпляром $\hat{R}_i^{out} = Q_i(DS^{in})$ и эталонным экземпляром $R_i^{out} = Q_i(DS^{in})$, на основе которых можно рассчитать качество искаженного экземпляра $DQ(\hat{R}_i^{out}, R_i^{out})$ по формуле (1). Следуя принципу «fitness-for-use» [5], целесообразно определить *качество Data-продукта* как среднее арифметическое от значений $DQ(\hat{R}_i^{out}, R_i^{out})$, взвешенное по критичности данных в каждом выходном порту для других доменов Data Mesh.

На практике доменные команды в большинстве случаев вынуждены оперировать только искаженными экземплярами, поскольку информация об эталонных экземплярах не доступна, – следовательно, непосредственный расчёт $DQ(\hat{R}_i^{out}, R_i^{out})$ по формуле (1) невозможен. Важно отметить, что современные методы мониторинга данных на основе различных характеристик и связанных с ними показателей качества данных [12] в комбинации с экспертными знаниями предметной области позволяют доменным командам эффективно выявлять искажения во входных данных и выполнять частичную очистку/исправление данных [13]. С точки зрения обнаружения расхождений между искаженным и эталонным экземплярами наиболее перспективным представляется использование внутренне присущих характеристик качества данных, таких как аккуратность, наполненность и согласованность. На рисунке 1 представлена схема процесса обработки данных внутри Data-продукта, обогащенная процессом управления качеством данных, который включает мониторинг входных данных на основе различных показателей качества данных и последующее частичное исправление данных от выявленных искажений (красным обозначены искаженные ячейки, а зелёным – исправленные).

Разные типы выявленных искажений имеют разный потенциал к очистке: например, значения в неверном формате можно легко исправить с помощью методов преобразования типов, в то время как заполнение пропущенных значений в ячейке требует знаний об эталонных значениях. Целесообразно рассмотреть три ключевых вида искаженных значений (см. таблицу 1), точное исправление которых невозможно без информации об эталонных значениях:

- пропущенные значения, выявляемые fullness-проверками μ^F на основе показателей «Наполненность обязательного атрибута»;
- значения вне допустимого диапазона, выявляемые domain-проверками μ^D на основе показателей «Семантическая аккуратность данных»;
- несогласующиеся значения, выявляемые consistency-проверками μ^C на основе показателей «Семантическая согласованность».

Поскольку в случае наличия искажений указанных видов полное исправление данных на практике является трудоемкой, а иногда и невыполнимой задачей, качество данных отдельных выходных портов и всего Data-продукта будет отличаться от 1, а задача определения качества Data-продукта в указанных условиях приобретает практическую ценность. Идея решения такой задачи базируется на том, что информация о выявленных в рамках мониторинга качества данных искажениях во входных данных может быть использована для анализа распространения этих искажений в результате выполнения SQL-запросов, описывающих трансформацию данных внутри Data-продукта и, как следствие, вычисления оценок параметров N_s, N_d, N_m, N_e для выходных портов. Полученные таким образом оценочные значения N_s, N_d, N_m, N_e можно использовать для вычисления оценки качества данных i-выходного порта DQ_i и расчёта итоговой оценки качества Data-продукта DQ^{DP}.

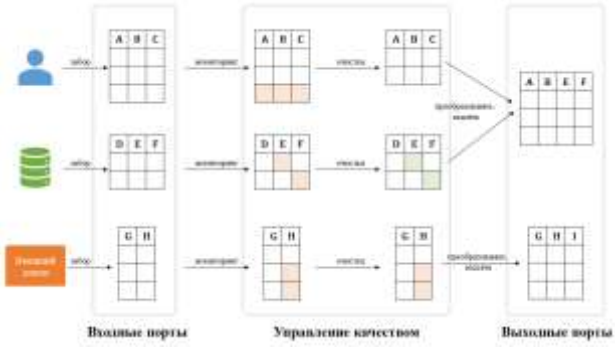


Рис. 1. Обработка данных в Data-продукте

В указанных условиях можно сформулировать следующую **задачу оценки качества Data-продукта в реляционной архитектуре Data Mesh**.

Дано:

- $DS^{in} = \hat{R}_1^{in} ... \hat{R}_n^{in}$ – набор искаженных экземпляров во входных портах Data-продукта после применения релевантной очистки данных;
- $\mathrm{M}^{\mathrm{F}} = \left\{ \mu_{i,j}^{\mathrm{F}} \mid \ i = 1..n, j = 1..m_i^{\mathrm{F}} \right\}$ – набор fullness-проверок качества DS^{in}, $\mathrm{M}^{\mathrm{D}} = \left\{ \mu_{i,j}^{\mathrm{D}} \mid \ i = 1..n, j = 1..m_i^{\mathrm{D}} \right\}$ – набор domain-проверок качества DS^{in}, $\mathrm{M}^{\mathrm{C}} = \left\{ \mu_{i,j}^{\mathrm{C}} \mid \ i = 1..n, j = 1..m_i^{\mathrm{C}} \right\}$ – набор consistency-проверок качества DS^{in}, где i-проверки применяются к $\hat{R}_i^{in}$, а неотрицательные параметры $m_i^{\mathrm{F}}, m_i^{D}, m_i^{\mathrm{C}}$ обозначают количество соответствующих проверок для i-входного экземпляра;
- $Q_1(DS^{in}) ... Q_k(DS^{in})$ – набор SQL-запросов, результаты которых формируют данные в выходных портах Data-продукта;
- $w_1 ... w_k$ – веса, обозначающие степень критичности данных соответствующих выходных портов Data-продукта для использования другими, причём $w_i > 0$ и $\sum_i w_i = 1, \ i = 1..k.$

Требуется: для каждого i-выходного порта рассчитать оценочные значения $N_s^i, N_d^i, N_m^i, N_e^i$ и определить качество Data-продукта по формуле:

$$DQ^{DP} = \sum_{i=1}^{k} w_i \cdot DQ_i, \qquad (2)$$

где значения DQ_i вычисляются по формуле (1) путём подстановки $N_s^i, N_d^i, N_m^i, N_e^i$ вместо N_s, N_d, N_m, N_e. □

III. Методика оценки

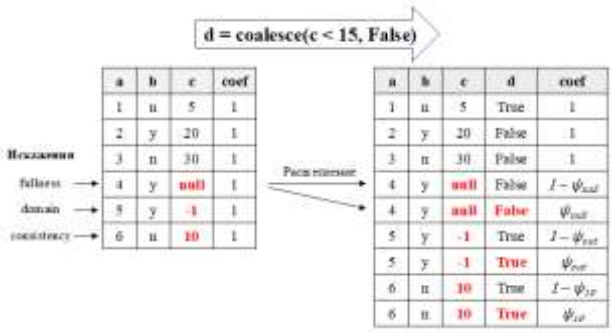
Опираясь на описанную выше идею решения сформулированной задачи, в настоящем исследовании предлагается методика оценки качества Data-продукта, состоящая из следующих шагов:

1. получение данных из входных портов после релевантной очистки данных;
2. выявление в полученных данных ошибок с помощью выполнения fullness-, domain- и consistency-проверок, т.е. разметка ячеек с искажениями;
3. декомпозиция каждого SQL-запроса, описывающего трансформации данных для каждого выходного порта, на цепочку простых операций реляционной алгебры;
4. трансформация «размеченных» входных данных с одновременным отслеживанием распространения искаженных ячеек (т.е. трекингом разметки);
5. подсчёт размеченных ячеек с искажениями для набора данных, полученного на последних шагах каждой трансформации, с целью формирования оценочных значений $N_s^i, N_d^i, N_m^i, N_e^i$.

В рамках данного исследования были рассмотрены SQL-запросы, которые можно декомпозировать на следующие базовые операции: проекция с добавлением атрибута, проекция с удалением атрибута, фильтрация по булевому атрибуту и декартово произведение. Несмотря на то, что рассматриваемый набор содержит всего несколько базовых операций, он позволяет описать существенный класс SQL-запросов, выполняемых современными СУБД, поскольку может быть расширен комбинациями указанных операций для представления ключевых манипуляций с реляционными данными, таких как фильтрация по любому условию (комбинация из добавления булевого атрибута и фильтрации по нему), внутреннее соединение (декартово произведение и фильтрация), оператор IN (аналог внутреннего соединения), операторы EXISTS и ANY (соединение и проекция), а также использования выражений CTE и подзапросов. Далее в работе проводится анализ распространения искажений в результате применения перечисленных базовых операций.

A. Операция проекции: добавление атрибута на основе одной переменной

Для анализа распространения искажений при выполнении данной операции необходимо учесть, что в кортежах, внутри которых вычисление нового значения выполняется над ячейкой с искаженными значениями, возникает неопределенность, является ли вычисленное значение искаженным или нет. Для примера на рисунке 2 представлен искаженный экземпляр с ключевым атрибутом a, строковым атрибутом b со значениями ‘y’ и ’n’ и числовым атрибутом c с семантически корректными значениями 5, 10, 20 и 30. В результате выполнения проверок качества данных были выявлены следующие искажения (выделены на рисунке красным): null-значение в кортеже $a = 4$ (fullness-проверка), значение -1 вне набора семантически корректных в кортеже $a = 5$ (domain-проверка) и семантически несогласованное по условию $f(b,c) = (b = \text{'n'} \ \text{AND} \ c \in \{5,30\})$ OR $(b = \text{'y'})$ значение 10 в кортеже $a = 6$ (consistency-проверка). В рамках операции проекции выполняется добавление булевого столбца d по условию $c < 15$, т.е. применяется операция проекции $\pi_{c<15 \to d}$.



a	b	c	coef
1	n	5	1
2	y	20	1
3	n	30	1
4	y	null	1
5	y	-1	1
6	n	10	1

a	b	c	d	coef
1	n	5	True	1
2	y	20	False	1
3	n	30	False	1
4	y	null	False	$1 - \psi_{null}$
4	y	null	False	ψ_{null}
5	y	-1	True	$1 - \psi_{out}$
5	y	-1	True	ψ_{out}
6	n	10	True	$1 - \psi_{10}$
6	n	10	True	ψ_{10}

Рис. 2. Пример добавления атрибута

В результате выполнения проекции в кортежах с искажениями нельзя однозначно определить, отличаются ли полученные значения d от эталонных и стоит ли помечать соответствующие ячейки как «искаженные». Например, для кортежа $a = 6$ столбцу d будет присвоено значение True и в случае, если эталонное значение c было бы равно 5, присвоенное значение не было искажено; при этом если эталонное значение было бы равно 30, то присвоенное значение было бы, напротив, искажено – отсюда и возникает неопределенность с разметкой но-

вой ячейки. Для решения возникшей неопределенности предлагается подход, основанный на «расщеплении» с помощью параметра $\psi \in [0;1]$ каждого такого кортежа на пару, состоящую из следующих элементов:

1. копия исходного кортежа с меткой **об отсутствии** искажения в новой ячейке и коэффициентом $coef = 1 - \psi$;
2. копия исходного кортежа с меткой **о наличии** искажения в новой ячейке и коэффициентом $coef = \psi$.

Для реализации расщепления необходимо дополнить исходное отношение техническим атрибутом $coef$: $dom(coef) \in [0;1]$, обладающим следующим смыслом: значение атрибута указывает на степень уверенности в том, что указанный кортеж действительно содержит искажения относительно эталона в ячейках, помеченных как «искаженные»; чем ближе значение $coef$ к 1, тем выше эта уверенность. Все кортежи во входных портах необходимо инициировать $coef = 1$, что соответствует полной уверенности в наличии ошибок в ячейках, помеченных как «искаженные». В результате расщепления кортежей с искажениями эта единица может быть в неравных долях поделена между расщепленными копиями, чтобы отделить варианты с наличием искажений в новом атрибуте от вариантов без искажений. Далее каждый кортеж из расщепленной пары может поразному трансформироваться в дальнейших операциях.

В свою очередь указанный выше параметр расщепления ψ можно интерпретировать как показатель того, является ли полученное в новом столбце значение искаженным относительно эталона. Для определения значения ψ в рамках данного исследования рассматривался подход на основе предположений о вероятностной природе значений атрибутов и искажений. В этом случае параметры расщепления можно определить как вероятность события, что значения нового атрибута в искаженном экземпляре отличаются от значения этого атрибута в эталонном экземпляре при условии наличия искажений в исходных атрибутах.

Дальнейшие рассуждения базируются на следующих предположениях:

1. значения атрибута a^{cont} вещественного типа внутри всех кортежей эталонного экземпляра являются реализацией **непрерывной** случайной величины, закон распределения которой заранее неизвестен;
2. выявленные в атрибуте a^{cont} вещественного типа искаженные значения также являются реализацией **непрерывной** случайной величины, распределенной независимо от эталонной (возможно, с иным законом распределения);
3. значения атрибута a^{disc} строкового или целочисленного типа (включая сводимые к ним, например тип дата/время) внутри всех кортежей эталонного экземпляра являются реализацией **дискретной** случайной величины;
4. искаженные значения атрибута a^{disc} строкового или целочисленного типа, принадлежащие $dom(a^{disc})$ и выявленные для всех кортежей с условием $v(t, a^{disc}) = \lambda$, также являются реализацией **дискретной** случайной величины над множеством $dom(a^{disc})$, λ и вероятностями, пропорциональными вероятностям значений a^{disc};
5. искаженные значения атрибута a^{disc} строкового или целочисленного типа, не принадлежащие $dom(a^{disc})$, являются реализацией дискретной случайной величины, распределенной независимо от эталонной.

Основываясь на указанных предположениях, для вычисления параметра ψ можно предложить следующий подход. Пусть в рамках выполнения операции проекции над искаженным экземпляром добавляется новый атрибут a^{new}, значения которого вычисляются на основе искаженного атрибута a^{dis} с помощью функции f. Значения атрибутов в искаженном экземпляре $\hat{\nu} = v(\hat{t}, a^{new})$ и $\hat{\alpha} = v(\hat{t}, a^{dis})$ являются случайными величинами, где t – кортежи искаженного экземпляра, причём соответствующие значения в эталонном экземпляре $\nu = v(t, a^{new})$ и $\alpha = v(t, a^{dis})$ также будут случайными величинами. Тогда коэффициент расщепления будет определяться через условную вероятность $\psi = P(\nu \neq \hat{\nu} \mid \alpha \neq \hat{\alpha})$. Далее рассмотрены два варианта оценки этого значения в зависимости от типа искаженного атрибута.

1. Вещественный тип a^{dis} – непрерывная СВ

Для определения параметра расщепления необходимо зафиксировать значение $\hat{\nu} = y$:

$$\begin{aligned}\psi_{cont}(y) &= P(\nu \neq \hat{\nu} \mid \alpha \neq \hat{\alpha};\ \hat{\nu} = y) \\ &= P(f(\alpha) \neq y \mid \alpha \neq \hat{\alpha};\ f(\hat{\alpha}) = y) \\ &= \frac{P(f(\alpha) \neq y;\ f(\hat{\alpha}) = y;\ \alpha \neq \hat{\alpha})}{P(f(\hat{\alpha}) = y;\ \alpha \neq \hat{\alpha})} \\ &= \frac{P(f(\alpha) \neq y;\ f(\hat{\alpha}) = y)}{P(f(\hat{\alpha}) = y)} = P(f(\alpha) \neq y)\end{aligned} \tag{3}$$

Здесь второе равенство получается из определения условной вероятности, третье – из предположения 1 и 2 о непрерывности распределения эталонных и искаженных значений, а четвертое – также из предположения 2 о независимости величин.

Далее из предположения 1 можно заключить, что вероятность $P(f(\alpha) \neq y)$ для множества кортежей с искажениями в a^{dis} равна вероятности такого же события для T_{clean} – множества кортежей, в которых в атрибуте a^{dis} не было выявлено искажений. Тогда вероятность $P(f(\alpha) \neq y)$ для кортежей из T_{clean} можно оценить через соответствующую долю:

$$\psi_{cont}(y) = P(f(\alpha) \neq y) \approx \frac{\left|\{t \in T_{clean} : f(\alpha) \neq y\}\right|}{\left|T_{clean}\right|} \tag{4}$$

Важно отметить, что параметр расщепления не зависит от типа искажения, то есть все ошибки, выявленные с помощью fullness-, domain- и consistency-проверок, необходимо обрабатывать одинаково.

В качестве частного случая применения формулы (4) для дальнейшего анализа важно рассмотреть ситуацию,

когда новый атрибут вычисляется как результат условия последующей фильтрации, т.е. в терминах SQL вычисляется выражение $a^{new} = coalesce(a^{dis} > L,$ False). Тогда для кортежей, где искаженное значение $v(\hat{t}, a^{dis}) > L$, параметр расщепления ψ необходимо вычислять как долю кортежей $v(\hat{t}, a^{dis}) \le L$ в T_{clean}, а для кортежей с пустым значением a^{dis} или $v(\hat{t}, a^{dis}) \le L$ параметр ψ будет равен доли кортежей $v(\hat{t}, a^{dis}) > L$ в T_{clean}.

Вторым важным частным случаем является ситуация, когда функция f является инъективным отображением. В этом случае для фиксированного значения $\hat{v} = y$ найдётся единственный элемент $\alpha_y : f(\alpha_y) = y$, откуда $\psi = P\big(f(\alpha) \ne y\big) = P(\alpha \ne \alpha_y) = 1$, т.е. значения $\hat{v} = v(\hat{t}, a^{new})$ точно будут искажены относительно эталона и расщепления не происходит.

2. Невещественный тип a^{dis} – дискретная СВ

В данном случае необходимо ввести обозначения $\alpha_1 ... \alpha_m \in dom(a^{dis})$ – возможные реализации случайной величины α, и $p_i = P(\alpha = \alpha_i)$ – вероятности их реализации. Тогда из предположения 4 можно получить, что для $i \ne j, i = 1..m$: $P(\hat{\alpha} = \alpha_i \mid \alpha = \alpha_j) = p_i / \sum_{i \ne j} p_i = p_i / (1 - p_j)$, а также $P(\hat{\alpha} = \alpha_i \mid \alpha = \alpha_i) = 0$.

Для определения параметра расщепления целесообразно зафиксировать значение $\hat{\alpha} = x$:

$$\begin{aligned}
\psi_{disc}(x) &\triangleq P\big(\nu \ne \hat{\nu} \mid \alpha \ne \hat{\alpha}; \hat{\alpha} = x\big) \\
&= P\big(f(\alpha) \ne f(x) \mid \alpha \ne x; \hat{\alpha} = x\big) \\
&= \frac{P\big(f(\alpha) \ne f(x); \alpha \ne x; \hat{\alpha} = x\big)}{P\big(\alpha \ne x; \hat{\alpha} = x\big)} \\
&= \frac{P\big(f(\alpha) \ne f(x); \alpha \ne x \mid \hat{\alpha} = x\big) P(\hat{\alpha} = x)}{P\big(\alpha \ne x \mid \hat{\alpha} = x\big) P(\hat{\alpha} = x)} \\
&= P\big(f(\alpha) \ne f(x); \alpha \ne x \mid \hat{\alpha} = x\big) \\
&= P\big(f(\alpha) \ne f(x) \mid \hat{\alpha} = x\big) = \sum_{\alpha_k \in \mathrm{A}(x)} p_C(\alpha_k, x),
\end{aligned} \tag{5}$$

$$p_C(\alpha_k, x) = P(\alpha = \alpha_k \mid \hat{\alpha} = x),\ \mathrm{A}(x) = \{\alpha_i \mid f(\alpha_i) \ne f(x)\}$$

В формуле (5) учтено, что для рассматриваемых кортежей эталонные и искаженные значения не совпадают, т.е. $P\big(\alpha \ne x; \hat{\alpha} = x\big) = 1$, а также равенство вероятностей $P\big(f(\alpha) \ne f(x); \alpha \ne x\big) = P\big(f(\alpha) \ne f(x)\big)$, которое следует из условия $f(\alpha) \ne f(x) \Rightarrow \alpha \ne x$.

Далее, для определения параметра расщепления необходимо определить величины $p_C(\alpha_k, x)$. Если $x \notin dom(a^{dis})$, то из предположения 5 следует, что $p_C(\alpha_k, x) = P(\alpha = \alpha_k) = p_k$. В случае, если $x \in dom(a^{dis})$, тогда найдётся такой индекс z: $\alpha_z = x$, откуда:

$$\begin{aligned}
p_C(\alpha_k, x) &= p_C(\alpha_k, \alpha_z) \triangleq P(\alpha = \alpha_k \mid \hat{\alpha} = \alpha_z) \\
&= \frac{P\big(\alpha = \alpha_k\big) P\big(\hat{\alpha} = \alpha_z \mid \alpha = \alpha_k\big)}{\sum_i P\big(\alpha = \alpha_i\big) P\big(\hat{\alpha} = \alpha_z \mid \alpha = \alpha_i\big)} \\
&= \frac{p_k \dfrac{p_z}{1 - p_k}}{\sum_{i \ne z} p_i \dfrac{p_z}{1 - p_i}} = \frac{q_k}{\sum_{i \ne z} q_i}, \quad q_i = p_i / (1 - p_i)
\end{aligned} \tag{6}$$

Здесь была использована расширенная формула Байеса и определенные выше условные вероятности для $\hat{\alpha}$.

Аналогично первому случаю, значения вероятности p_i можно оценить через доли кортежей со значением α_i в атрибуте a^{dis} во множестве T_{clean}, т.е.

$$p_i \approx \frac{\big|\{t \in T_{clean} \mid \alpha = \alpha_i\}\big|}{\big|T_{clean}\big|} \tag{7}$$

Далее, выполнив переход от p_i к q_i, с помощью формулы (6) можно определить оценку значений $p_C(\alpha_k, x)$ и далее с помощью (5) оценку значения $\psi_{disc}(x)$ для каждого кортежа.

В качестве частного случая применения формулы (5) целесообразно рассмотреть ситуацию, когда новый атрибут в терминах SQL вычисляется как выражение $a^{new} = coalesce(a^{dis} = L,$ False), причем $L \in dom(a^{dis})$. Тогда в зависимости от характера искажения можно заключить следующее:

- если искаженное значение было выявлено с помощью fullness-проверки, т.е. $x = \mathrm{null} \notin dom(a^{dis})$, то $f(x) = f(\mathrm{null}) = \mathrm{False}$, $\mathrm{A}(x) = \{\alpha_i \mid f(\alpha_i) \ne \mathrm{False}\} = \{L\}$ и $p_C(\alpha_k, x) = p_k$, откуда $\psi_{disc} = p_l,\ l : \alpha_l = L$;
- искаженное значение было выявлено с помощью domain-проверки, т.е. $x \notin dom(a^{dis})$, тогда аналогично $f(x) = \mathrm{False}$, $\mathrm{A}(x) = \{\alpha_i \mid f(\alpha_i) \ne \mathrm{False}\} = \{L\}$ и $\psi_{disc} = p_l,\ l : \alpha_l = L$;
- искаженное значение было выявлено с помощью consistency-проверки и равняется L, т.е. $x = L$, тогда $\mathrm{A}(L) = \{\alpha_i \mid f(\alpha_i) \ne \mathrm{True}\} = \{\alpha_i \mid \alpha_i \ne L \vee \alpha_i = \mathrm{null}\}$, что эквивалентно множеству $\{\alpha_i \mid \alpha_i \ne x\}$, откуда следует $\psi_{disc}(L) = \sum_{\alpha_k \ne x} P\big(\alpha = \alpha_k \mid \hat{\alpha} = x\big) = 1$;
- искаженное значение было выявлено с помощью consistency-проверки и не равно L, т.е. $x \in dom(a^{dis})$, L, тогда $\mathrm{A}(x) = \{\alpha_i \mid f(\alpha_i) \ne \mathrm{False}\} = \{L\}$, откуда следует $\psi_{disc}(x) = \sum_{\alpha_k = L} p_C\big(\alpha_k, x\big) = p_C(\alpha_l, \alpha_z)$, где $l : \alpha_l = L$ и $z : \alpha_z = x$.

Вторым важным частным случаем, как и для варианта атрибута вещественного типа, является ситуация, когда функция f является инъективным отображением. В этом случае для фиксированного значения $\hat{\alpha} = x$ не найдётся ни одного $\alpha_i \ne x : f(\alpha_i) = f(x)$, т.е. $\mathrm{A}(x) = \{\alpha_i \mid \alpha_i \ne x\}$, откуда получаем $\psi = \sum_{\alpha_k \ne x} P\big(\alpha = \alpha_k \mid \hat{\alpha} = x\big) = 1$, т.е. здесь также расщепления не происходит.

B. Операция проекции: добавление атрибута на основе булевой функции

Для анализа распространения искажений данных в случае выполнения операций фильтраций необходимо дополнить результаты предыдущего подраздела исследованием распространения искажений в случае добавления булевого атрибута на базе двух переменных с использованием операций AND и OR. Продолжая рассмотрение описанного выше примера, на рисунке 3 представлен искаженный экземпляр, дополненный булевым атрибутом e, значения которого вычисляются из условию $b = \text{'y'}$. В рамках операции проекции выполняется добавление булевого столбца $f = (d \text{ AND } e)$, т.е. применяется операция проекции $\pi_{d \wedge e \to f}$.

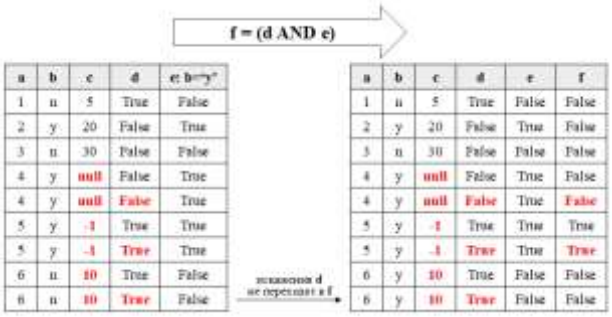


Рис. 3. Пример добавления булевого атрибута

В результате выполнения такой проекции во время расчёта значения нового атрибута в отдельных кортежах происходит сочетание искаженных значений d и неискаженных в e. При этом, в частности, наличие False в атрибуте e в кортежах $a = 6$ позволяет нивелировать наличие искажений в d, поскольку в любом случае искаженный и эталонный результаты примут значения False.

Обобщая эту идею, необходимо рассмотреть операцию проекции $\pi_{f(a_1,a_2) \to a^{new}}$, в рамках которой на основе атрибутов a_1, a_2 вычисляется новый атрибут a^{new}, причём функция $f \in \{\wedge, \vee\}$ принимает и возвращает булевые значения, т.е. $f(a_1, a_2) : boolean \times boolean \to boolean$. Для того чтобы определить, является ли результирующее значение атрибута a^{new} искаженным, достаточно смоделировать всевозможные комбинации эталонных значений и проследить, каким образом искажения входных значений влияют на наличие искажений на выходе операции. В таблице II представлены различные комбинаций значений атрибутов a_1, a_2 и значения функций $f(a_1, a_2) = a_1 \wedge a_2$ и $f(a_1, a_2) = a_1 \vee a_2$ в эталонном экземпляре данных. Далее моделируются искажения значений атрибута a_1, а результат выполнения операций над искаженными значениями сравнивается с эталонным – все искажения выделены красным.

Таблица II. Добавление атрибута на основе операций AND и OR (одно искажение)

Эталонный экземпляр				**Искаженный экземпляр**			
a_1	a_2	$a_1 \wedge a_2$	$a_1 \vee a_2$	a_1	a_2	$a_1 \wedge a_2$	$a_1 \vee a_2$
T	T	T	T	**F**	T	**F**	T
T	F	F	T	**F**	F	F	**F**
F	T	F	T	**T**	T	**T**	T
F	F	F	F	**T**	F	F	**T**

Аналогичным образом в таблице III приведён анализ для случая, когда в искаженном экземпляре значения обоих атрибутов a_1, a_2 заданы с ошибкой. Здесь также пара значений a_1, a_2 в искаженном экземпляре однозначно определяет наличие ошибки в новом атрибуте. Легко видеть, что пара значений в атрибутах, участвующих в операции, и информация о наличии искажений в одном или двух значениях однозначно определяет, будет ли значение в новом атрибуте содержать искажения или нет – это позволяет отследить распространение искажений в результате выполнения операции.

Таблица III. Добавление атрибута на основе операций AND и OR (два искажения)

Эталонный экземпляр				**Искаженный экземпляр**			
a_1	a_2	$a_1 \wedge a_2$	$a_1 \vee a_2$	a_1	a_2	$a_1 \wedge a_2$	$a_1 \vee a_2$
T	T	T	T	**F**	**F**	**F**	**F**
T	F	F	T	**F**	**T**	F	T
F	T	F	T	**T**	**F**	F	T
F	F	F	F	**T**	**T**	**T**	**T**

C. Операция фильтрации по булевому атрибуту

Вслед за анализом операции добавления атрибута необходимо рассмотреть влияние выявленных искажений на результат применения операции фильтрации. Далее приводится результат исследования операции фильтрации по условию равенства непустого булевого атрибута значению True, что не ограничивает общности, поскольку любую операции фильтрации σ_C можно заменить последовательностью из двух операций: расчёт нового атрибута с условием $a^{new} = coalesce(C, \text{False})$, комбинируя вычисления на базе одной переменной с помощью AND/OR; выполнение операции $\sigma_{a^{new}=\text{True}}$.

Продолжая рассмотрение описанного выше примера, на рисунке 4 представлена операция фильтрация $\sigma_{f=\text{True}}$ на основе булевого столбца f.

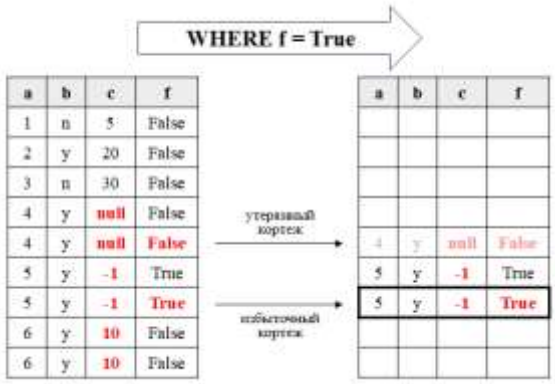


Рис. 4. Пример фильтрации по булевому атрибуту

Большая часть кортежей будет отброшена в результате фильтрации и в выходном отношении останутся только кортежи $a=5$, однако наличие искажений в атрибуте f оказывает существенный эффект на отличие выходного отношения от эталонного результата:

- В кортеже $a=4$ искаженное значение f равно False и в результате применения фильтрации этот кортеж будет **отброшен**. Ввиду того, что в булевом столбце могут содержаться только значения True/False, в эталонном экземпляре значение f могло быть равно только True и данный кортеж был бы **сохранен**. Другими словами, в результате искажения кортеж после фильтрации оказался «утерянным» в сравнении с эталонным экземпляром, т.е. дополнил множество T_{missed}.
- В кортеже $a=5$ искаженное значение f равно True и наблюдается обратная ситуация: в результате применения фильтрации этот кортеж будет **сохранен**, в то время как в эталонном экземпляре он был бы **отброшен**. Таким образом, в результате искажения кортеж после фильтрации оказался «избыточным» в сравнении с эталонным экземпляром, т.е. дополнил множество $\hat{T}_{extra}$.

Обобщая этот пример, можно заключить, что наличие искаженных ячеек и применение операции фильтрации приводит к смене типа кортежа: по определению, кортеж с искажением изначально входит во множество $\hat{T}_{different}$, но после фильтрации должен быть сохранён и перемещен во множество T_{missed} или $\hat{T}_{extra}$, что необходимо отслеживать наряду с распространением искаженных ячеек. Для полноты картины необходимо также рассмотреть влияние искаженных ячеек на результаты фильтрации в случаях, когда входные кортежи были отнесены ко множествам T_{missed} и $\hat{T}_{extra}$ в рамках предыдущих шагов цепочки SQL-запроса. Результаты анализа, аналогичного представленному для примера выше, сведены в таблицу IV: для каждого типа входного кортежа указаны варианты значений булевого атрибута (эталонное и искаженное), наличие или отсутствие кортежа на выходе фильтрации (+ или –) и действия по отслеживанию типа выходного кортежа.

Таблица IV. Фильтрация по булевому атрибуту

Тип кортежа	**Эталонный экземпляр**		**Искаженный экземпляр**		**Действие над кортежем**
	Вход	Выход	Вход	Выход	
$\hat{T}_{different}$	T	+	**F**	–	Смена типа T_{missed}
$\hat{T}_{different}$	F	–	**T**	+	Смена типа $\hat{T}_{extra}$
T_{missed}	T	+	**F**	–	Удержание
T_{missed}	F	–	**T**	–	Исключение
$\hat{T}_{extra}$	T	–	**F**	–	Исключение
$\hat{T}_{extra}$	F	–	**T**	+	Удержание

D. Операция проекции: удаление атрибута

Для анализа операции удаления атрибута необходимо рассмотреть два случая: удаляемый атрибут не включен в состав ключа или, напротив, включён. На рисунке 5 представлен пример искаженного экземпляра с ключевым атрибутом a и атрибутами b,c, к которому применяется операция проекции с удалением двух атрибутов $\pi_{a,\,b\to\varnothing}$, в результате которого новым ключом становится атрибут c.

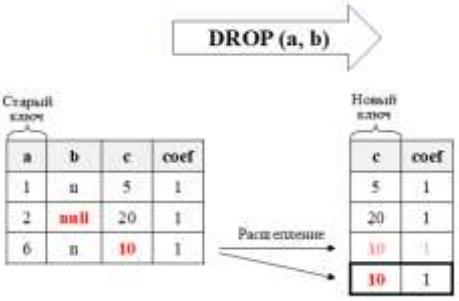


Рис. 5. Пример удаления ключевого и неключевого атрибутов

В кортеже $a=2$ удаление атрибута b приводит к исключению ячейки с искажением без особенностей. В то же время, в кортеже $a=6$ удаление ключевого атрибута a приводит к тому, что значение нового ключевого атрибута $c=10$ будет искажено и, следовательно, соответствующий ему кортеж в эталонном экземпляре будет иметь отличающийся ключ. Это приведёт к двойному эффекту: с одной стороны, искаженный кортеж с ключом $c=10$ нельзя будет найти в эталонном экземпляре, т.е. он дополнит множество $\hat{T}_{extra}$; с другой стороны, эталонный кортеж с корректным ключом нельзя будет найти в искаженном экземпляре, т.е. он дополнит множество T_{missed}.

Описанный выше пример можно обобщить следующим образом: корректное отслеживание искажений в результате операции удаления атрибута сводится к прекращению трекинга искажений в удаляемом атрибуте и, в случае смены ключа, к расщеплению кортежа с искажениями в новом ключе на две копии, причём первую копию необходимо включить во множество $\hat{T}_{extra}$, а вторую – во множество T_{missed}.

E. Операции декартова произведения

Поскольку в результате операции декартова произведения не происходит удаления или добавления новых атрибутов, то искаженные ячейки не распространяются «в ширину» и потому не требуют дополнительного отслеживания. Однако, как и в случае операции фильтрации, корректный анализ распространения искажений должен учитывать типы кортежей, участвующих в сочетаниях декартова произведения.

Для примера можно рассмотреть сочетание, в котором один входной кортеж принадлежит множеству $\hat{T}_{extra}$, а второй – множеству $\hat{T}_{different}$. В этом случае искаженный экземпляр будет включать оба кортежа и в выходном искаженном экземпляре образуется их сочетание. При этом по определению $\hat{T}_{extra}$ первый кортеж отсутствовал бы в эталонном экземпляре, потому в выходном эталонном экземпляре сочетания кортежей не образовалось бы. Сравнивая искаженный и эталонный результаты, легко видеть, что такие сочетания необходимо сохранить в результате применения операции над искаженным экземпляром и переместить во множество $\hat{T}_{extra}$.

В таблице V приведён результат анализа для всевозможных сочетаний кортежей $\hat{t}_1$ и $\hat{t}_2$ искаженных экземпляров. Для каждой пары входных кортежей указаны признаки их наличия или отсутствия в экземпляре (+ или –), признак наличия/отсутствия результирующего кортежа $\hat{t}_r$ и действия по отслеживанию типа выходного кортежа.

Таблица V. Сочетания декартова произведения

Тип кортежей	**Эталонный экземпляр**			**Искаженный экземпляре**			**Действие над сочетанием**
	t_1	t_2	t_r	$\hat{t}_1$	$\hat{t}_2$	$\hat{t}_r$	
$T_{missed}, \hat{T}_{extra}$	+	–	–	–	+	–	Исключение
$T_{missed}, \hat{T}_{different}$	+	+	+	–	+	–	Удержание с типом T_{missed}
$T_{missed}, \hat{T}_{similar}$	+	+	+	–	+	–	Удержание с типом T_{missed}
$\hat{T}_{extra}, \hat{T}_{different}$	–	+	–	+	+	+	Удержание с типом $\hat{T}_{extra}$
$\hat{T}_{extra}, \hat{T}_{similar}$	–	+	–	+	+	+	Удержание с типом $\hat{T}_{extra}$
$\hat{T}_{different}, \hat{T}_{similar}$	+	+	+	+	+	+	Удержание с типом $\hat{T}_{different}$

IV. Программное обеспечение

Для реализации представленной выше методики оценки качества Data-продукта предлагается на основе искаженного экземпляра $\hat{R}(A)$ сконструировать отношение $\tilde{R}(\tilde{A})$, именуемое далее как *размеченный экземпляр,* по следующему правилу:

1. $\tilde{r} = \hat{r}$ – тело является копией тела искаженного экземпляра;
2. $\tilde{A} = A \bigcup \{type, track, coef\}$ – набор атрибутов искаженного экземпляра, дополненный техническими атрибутами;
3. Атрибут *type* обозначает тип кортежа и содержит значения из списка: ‘found’, ‘missed’ и ‘extra’;
4. Атрибут *track* содержит метки ячеек с искаженными значениями и заполняется множеством неключевых атрибутов вида $v(t, track) = S : S \subseteq A_{nkey}$;
5. Атрибут *coef*, определённый ранее в работе и содержащий значения [0;1].

Таким образом, размеченный экземпляр $\tilde{R}(\tilde{A})$ представляет собой копию искаженного экземпляра, дополненную техническими атрибутами $type, track, coef$ для отслеживания искажений.

Реализация методики оценки качества Data-продукта на базе размеченных экземпляров сводится к следующим действиями. На втором шаге для каждого входного порта необходимо подготовить размеченные экземпляры следующим образом: атрибут *type* инициализируется значением ‘found’, атрибут *coef* – значением 1, а в значение атрибута *track* для каждого кортежа записывается набор атрибутов с искаженными ячейками, выявленными в результате проверок качества. Далее на четвертом шаге происходит последовательная модификация размеченных экземпляров в результате выполнения элементарных операций, включающая: изменения значений *coef* при расщеплении кортежей и их сочетании в декартовом произведении; дополнение/сокращение списка атрибутов в *track* в результате добавления/удаления атрибутов; замена значения *type* на ‘missed’ в случае включения кортежа во множество T_{missed} и ‘extra’ при включении в $\hat{T}_{extra}$. На пятом шаге для каждого размеченного экземпляра в i-выходном порту происходит подсчёт оценочных значений:

- $N_s^i = \{v(t, coef) |\ t \in \tilde{r} \wedge v(t, type) = \text{'found'} \wedge v(t, track) = \varnothing\}$;
- $N_d^i = \{v(t, coef) |\ t \in \tilde{r} \wedge v(t, type) = \text{'found'} \wedge v(t, track) \neq \varnothing\}$;
- $N_m^i = \{v(t, coef) |\ \ t \in \tilde{r} \wedge v(t, type) = \text{'missed'}\}$;
- $N_e^i = \{v(t, coef) |\ \ t \in \tilde{r} \wedge v(t, type) = \text{'extra'}\}$.

Далее представлен алгоритм оценки качества Data-продукта в реляционной архитектуре Data Mesh, позволяющий решить сформулированную в разделе II задачу и реализующий описанную в разделе III методику (алгоритм 1). Алгоритм использует вычисление пяти функций с наименованиями TRANSFORM*, каждая из которых предназначена для отслеживания распространения искажений в соответствующих элементарных операциях реляционной алгебры (подразделы A-E раздела III), получает на вход размеченный экземпляр и возвращает трансформированный размеченный экземпляр. Результатом работы алгоритма являются оценочные параметры выходных портов $N_s^i, N_d^i, N_m^i, N_e^i$ и итоговое оценочное значение качества данных Data-продукта DQ^{DP}.

Алгоритм 1. Оценка качества Data-продукта в реляционной архитектуре Data Mesh

ESTIMATEDATAPRODUCTQUALITY (DS^{in}, M^{F}, M^{D}, M^{C}, Q, w, A^{key})

Вход:

1. $DS^{in} = \hat{R}_1^{in} ... \hat{R}_n^{in}$ – набор искаженных экземпляров;
2. $\mathrm{M}^{\mathrm{F}} = \left\{ \mu_{i,j}^{\mathrm{F}} \mid \ i = 1..n, j = 1..m_i^{\mathrm{F}} \right\}$, $\mathrm{M}^{\mathrm{D}} = \left\{ \mu_{i,j}^{\mathrm{D}} \mid \ i = 1..n, j = 1..m_i^{\mathrm{D}} \right\}$ и $\mathrm{M}^{\mathrm{C}} = \left\{ \mu_{i,j}^{\mathrm{C}} \mid \ i = 1..n, j = 1..m_i^{\mathrm{C}} \right\}$ – набор проверок качества данных над DS^{in};
3. $Q = Q_1(DS^{in}) ... Q_k(DS^{in})$ – набор SQL-запросов, разложенных на цепочки $Q_i = O_i^1 \ldots O_i^l$, состоящие из $l(i)$ элементарных операций вида $\left\{ \pi_{f(a) \to a^{new}}, \sigma_{a=True}, \pi_{a \to \varnothing}, \times \right\}$;
4. $w = w_1 ... w_k$ – веса критичности, $w_i > 0$ и $\sum w_i = 1, \ i = 1..k$;
5. $A^{key} = A_1^{key} ... A_k^{key}$ – набор ключей для каждого выходного порта, где для i –выходного порта набор ключей $A_i^{key} = A_{i,1}^{key}, ... A_{i,l}^{key}$ объединяет $A_{i,j}^{key}$ – массивы ключей для результата операции O_i^j.

Выход: оценка качества Data-продукта DQ_{est}

for $i \leftarrow 1...n$ **do**
  $\Xi_i^{\mathrm{F}} \leftarrow \bigcup_{j=1..m_i^{\mathrm{F}}} \Xi\left(\mu_{i,j}^{\mathrm{F}}\right)$; $\Xi_i^{\mathrm{D}} \leftarrow \bigcup_{j=1..m_i^{\mathrm{F}}} \Xi\left(\mu_{i,j}^{\mathrm{D}}\right)$; $\Xi_i^{\mathrm{C}} \leftarrow \bigcup_{j=1..m_i^{\mathrm{F}}} \Xi\left(\mu_{i,j}^{\mathrm{C}}\right)$
  $\Xi_i \leftarrow \Xi_i^{\mathrm{F}} \bigcup \Xi_i^{\mathrm{D}} \bigcup \Xi_i^{\mathrm{C}}$
  $\tilde{R}_i^{in}(\tilde{A}_i) \leftarrow \hat{R}_i^{in}(A_i), \ \tilde{A}_i = A_i \bigcup \{type, track, coef\}$
  for all $t \in \tilde{r}_i^{in}$ **do**
    $v(t, coef) \leftarrow 1; \ v(t, type) \leftarrow$ 'found'; $v(t, track) \leftarrow \{a \mid \ a \in A_i \wedge v(t, a) \in \Xi_i\}$
  end for
end for
for $i \leftarrow 1...k$ **do**
  $DS^{temp} \leftarrow$ **copy** DS; $l \leftarrow$ длина цепочки Q_i; $QL \leftarrow \varnothing$
  for $j \leftarrow 1...l$ **do**
    определить $\tilde{R}^{op} \subseteq DS^{temp}$ – множество отношений, участвующих в операции O_i^j
    case $type(O_i^j)$ **of**
      $\pi_{f(a) \to a^{new}}$: $\tilde{R}^{last} \leftarrow$ TRANSFORMPROJECTIONADDSIMPLE ($f(a)$, a^{new}, $\tilde{R}_1^{op}$)
      $\pi_{f(a_1, a_2) \to a^{bool}}$, $f \in \{\wedge, \vee\}$: $\tilde{R}^{last} \leftarrow$ TRANSFORMPROJECTIONADDBOOLEAN ($f(a_1, a_2)$, a^{bool}, $\tilde{R}_1^{op}$)
      $\sigma_{a=True}$: $\tilde{R}^{last} \leftarrow$ TRANSFORMFILTERBOOLEAN (a, $\tilde{R}_1^{op}$)
      $\pi_{a \to \varnothing}$: $\tilde{R}^{last} \leftarrow$ TRANSFORMPROJECTIONDROPATTRIBUTE (a, $\tilde{R}_1^{op}$, $A_{i,j}^{key}$)
      $\times$: $\tilde{R}^{last} \leftarrow$ TRANSFORMCROSSJOIN ($\tilde{R}_1^{op}$, $\tilde{R}_2^{op}$)
    end case
    $DS^{temp} \leftarrow DS^{temp} \bigcup \{\tilde{R}^{last}\}$
  end for
  $N_s^i \leftarrow \{v(t, coef) \mid \ t \in \tilde{r}^{last} \wedge v(t, type) = \text{'found'} \wedge v(t, track) = \varnothing\}$
  $N_d^i \leftarrow \{v(t, coef) \mid \ t \in \tilde{r}^{last} \wedge v(t, type) = \text{'found'} \wedge v(t, track) \neq \varnothing\}$
  $N_m^i \leftarrow \{v(t, coef) \mid \ t \in \tilde{r}^{last} \wedge v(t, type) = \text{'missed'}\}$
  $N_e^i \leftarrow \{v(t, coef) \mid \ t \in \tilde{r}^{last} \wedge v(t, type) = \text{'extra'}\}$
  $QL \leftarrow QL \bigcup \left\{ w_i \cdot \frac{N_s^i}{N_s^i + N_d^i + N_m^i + N_e^i} \right\}$
end for
$DQ_{est} \leftarrow \sum QL_i, \ i = 1..k$
return $N_s^i, N_d^i, N_m^i, N_e^i, DQ_{est}$

TRANSFORMPROJECTIONADDSIMPLE ($f(a)$, a^{new}, $\tilde{R}^{in}$)

$\tilde{R}^{out} \leftarrow$ **copy** $\pi_{f(a) \to a^{new}}(\tilde{R}^{in})$
$T_{clean} = \{t \mid \ t \in \tilde{R}^{in} \wedge a \notin v(t, track)\}$
for all $t \in \tilde{r}^{out}$ **do**

if $a \in v(t, track)$ **then**
  $\psi \leftarrow$ GETSPLITTINGPSI ($f(a)$, $v(t,a)$, T_{clean})
  $t_e \leftarrow t$; $v(t_e, track) \leftarrow v(t_e, track) \bigcup \{a^{bool}\}$; $v(t_e, coef) \leftarrow v(t_e, coef) \cdot \psi$
  добавить t_e в $\tilde{r}^{out}$
  $v(t, coef) \leftarrow v(t, coef) \cdot (1 - \psi)$
end if
end for
return $\tilde{R}^{out}$

GETSPLITTINGPSI ($f(a)$, $\hat{\alpha}$, T_{clean})

if $typeof(a) = $ 'real' **then** $\psi \leftarrow \left|\{t \in T_{clean} : f(v(t,a)) \neq f(\hat{\alpha})\right| / \left|T_{clean}\right|$
else
  for all $\alpha_i \in dom(a)$ **do**
    $p_i \leftarrow \left|\{t \in T_{clean} \mid v(t,a) = \alpha_i\}\right| / \left|T_{clean}\right|$; $q_i \leftarrow p_i / (1 - p_i)$
  end for
  $p_S \leftarrow 0$; $z \leftarrow$ индекс, для которого $\alpha_i = \hat{\alpha}$, $\alpha_i \in dom(a)$
  for all $\alpha_k \in dom(a) : f(\alpha_k) \neq f(\hat{\alpha})$ **do** $p_S \leftarrow p_S + \dfrac{q_k}{\sum_{i \neq z} q_i}$ **end for**
  $\psi \leftarrow p_S$
end if
return ψ

TRANSFORMPROJECTIONADDBOOLEAN ($f(a_1, a_2)$, a^{bool}, $\tilde{R}^{in}$)

$\tilde{R}^{out} \leftarrow$ **copy** $\pi_{f(a_1, a_2) \Rightarrow a^{bool}}(\tilde{R}^{in})$
for all $t \in \tilde{r}^{out}$ **do**
  $ftype \leftarrow typeof(f)$
  if $\left|v(t, track)\right| = 2 \ \wedge \ v(t, a_1) = v(t, a_2)$ **then** $v(t, track) \leftarrow v(t, track) \bigcup \{a^{bool}\}$
  elif $\left|v(t, track)\right| = 1 \ \wedge \ ftype = $ 'AND' $\wedge \ v(t, a_2) = $ True **then** $v(t, track) \leftarrow v(t, track) \bigcup \{a^{bool}\}$
  elif $\left|v(t, track)\right| = 1 \ \wedge \ ftype = $ 'OR' $\wedge \ v(t, a_2) = $ False **then** $v(t, track) \leftarrow v(t, track) \bigcup \{a^{bool}\}$
  end if
end for
return $\tilde{R}^{out}$

TRANSFORMFILTERBOOLEAN (a, $\tilde{R}^{in}$)

$\tilde{R}^{out} \leftarrow$ **copy** $\tilde{R}^{in}$
for all $t \in \tilde{r}^{out}$ **do**
  if $a \notin v(t, track) \wedge v(t, a) = False$ **then** удалить t из $\tilde{r}^{out}$
  elif $a \in v(t, track) \wedge v(t, type) = $ 'found' $\wedge v(t, a) = False$ **then** $v(t, type) \leftarrow$ 'missed'
  elif $a \in v(t, track) \wedge v(t, type) = $ 'found' $\wedge v(t, a) = True$ **then** $v(t, type) \leftarrow$ 'extra'
  elif $a \in v(t, track) \wedge v(t, type) = $ 'missed' $\wedge v(t, a) = True$ **then** удалить t из $\tilde{r}^{out}$
  elif $a \in v(t, track) \wedge v(t, type) = $ 'extra' $\wedge v(t, a) = False$ **then** удалить t из $\tilde{r}^{out}$
  end if
end for
return $\tilde{R}^{out}$

TRANSFORMPROJECTIONDROPATTRIBUTE (a, $\tilde{R}^{in}$, A^{out}_{key})

1 $\tilde{R}^{out} \leftarrow$ **copy** $\pi_{a \to \varnothing}(\tilde{R}^{in})$
2 **for all** $t \in \tilde{r}^{out}$ **do**
3 **if** $a \in v(t, track)$ **then** $v(t, track) \leftarrow v(t, track) \setminus \{a\}$ **end if**
4 **if** $v(t, type) = \text{'found'} \wedge \exists a^{key} \in A^{out}_{key} : \; a^{key} \in v(t, track)$ **then**
5 $t_e \leftarrow t$; $v(t_e, track) \leftarrow v(t_e, track) \setminus A^{out}_{key}$; $v(t_e, type) \leftarrow \text{'extra'}$
6 добавить t_e в $\tilde{r}^{out}$
7 $v(t, type) \leftarrow \text{'missed'}$
8 **end if**
9 **end for**
10 **return** $\tilde{R}^{out}$

TRANSFORMCROSSJOIN ($\tilde{R}^{in}_1$, $\tilde{R}^{in}_2$)

1 $R^{mod}_1 \leftarrow$ **copy** $\tilde{R}^{in}_1$, выполнено переименование атрибутов: $type \to type1$, $track \to track1$, $coef \to coef1$
2 $R^{mod}_2 \leftarrow$ **copy** $\tilde{R}^{in}_2$, выполнено переименование атрибутов: $type \to type2$, $track \to track2$, $coef \to coef2$
3 $\tilde{R}_{out} \leftarrow R^{mod}_1 \times R^{mod}_2$
4 **for all** $t \in \tilde{r}^{out}$ **do**
5 **if** $\left(v(t, type1) = \text{'extra'} \wedge v(t, type2) = \text{'missed'}\right) \vee \left(v(t, type2) = \text{'extra'} \wedge v(t, type1) = \text{'missed'}\right)$ **then**
6 удалить t из $\tilde{r}^{out}$
7 **else**
8 **if** $\text{'missed'} \in \{v(t, type1), v(t, type2)\}$ **then** $ntype \leftarrow \text{'missed'}$
9 **elif** $\text{'extra'} \in \{v(t, type1), v(t, type2)\}$ **then** $ntype \leftarrow \text{'extra'}$
10 **else** $ntype \leftarrow \text{'found'}$
11 **end if**
12 $v(t, type) \leftarrow ntype$; $v(t, track) \leftarrow v(t, track1) \bigcup v(t, track2)$; $v(t, coef) \leftarrow v(t, coef1) \cdot v(t, coef2)$
15 **end if**
16 **end for**
17 удалить атрибуты $type1$, $type2$, $track1$, $track2$, $coef1$, $coef2$ из $\tilde{R}^{out}$
18 **return** $\tilde{R}^{out}$

V. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТАЛЬНОГО ТЕСТИРОВАНИЯ

Для экспериментального тестирования был выбран открытый набор данных, который содержит деперсонализированную статистику онлайн обучения студентов на платформе «Virtual Learning Environment (VLE)» в крупнейшем британском университете Open University за 2013-2014 года [9]. Для представления реляционной архитектуры Data Mesh предметная область, описываемая указанным набором данных, была разделена на 3 домена: домен «Student demographics», управляющий данными из таблицы studentInfo; домен «Activities», управляющий данными из таблиц studentRegistration, studentAssessment и studentVle; домен «Module presentation», управляющий данными из таблиц courses, assessments, vle.

Для тестирования был смоделирован Data-продукт в домене «Student demographics», состоящий из следующих элементов (рисунок 7):

- входной порт 1: таблица student_info_manual, содержащая результаты «ручного» ввода студентами информации о регионе проживания и уровне высшего образования;
- входной порт 2: таблица student_info_auto, выгружаемая из автоматизированной системы с информацией о половой принадлежности студента и наличии инвалидности;
- входной порт 3: таблица student_exam_score, получаемая от внешнего Data-продукта домена «Activities» с данными об оценках на экзаменах;
- выходной порт 1 представлен результатом трансформации SQL-1, состоящей из соединения таблиц 1го и 2го входных портов с последующим инъективным отображением highest_education в education_level и фильтрацией по условию: «*region <> 'East Anglian Region'*»;
- выходной порт 2 представлен результатом трансформации SQL-2, состоящей из соединения таблиц 1го и 2го входных портов, последующей фильтрацией по условию «*disability = True*» и удалением этого атрибута;
- выходной порт 3 представлен результатом трансформации SQL-3, состоящей из фильтрации по вещественному атрибуту «*score < 0.6*» и удалением этого атрибута.

При проведении расчётов для указанных выходных портов были заданы веса критичности (0.3, 0.1, 0.6).

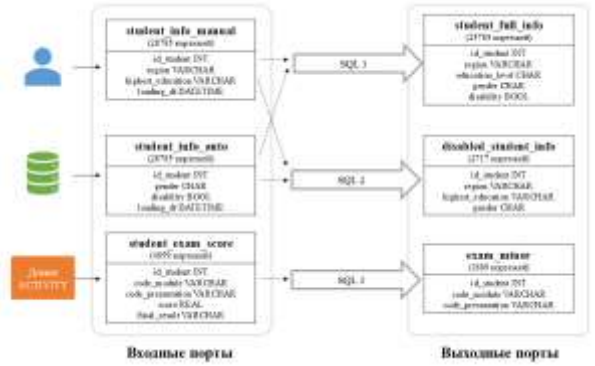


Рис. 6. Data-продукт домена “Student demographics”

В рамках экспериментального тестирования было подготовлено несколько вариантов искажений входных данных, затрагивающих не более 10% записей и выявляемых разными типами проверок. На первом этапе для каждого варианта искажений с помощью прямого вычисления были получены эталонные и искаженные экземпляры выходных портов, что позволило рассчитать точные значения параметров N_s, N_d, N_m, N_e и значения качества данных $DQ(\hat{R}^{out}, R^{out})$. На втором этапе с помощью описанной методики и алгоритма 1 для каждого выходного порта были получены оценки N_s, N_d, N_m, N_e и оценки качества данных DQ. Результаты эксперимента представлены в таблице VI: показатели x, $\tilde{x}$ и Δ обозначают, соответственно, точное значение, оценку и разницу между ними.

Таблица VI. Результаты эксперимента

Вых. порт		N_s	N_d	N_m	N_e	DQ
Искажения student_info_manual (fullness-, domain-проверки), **качество Data-продукта: точное 0,9599, оценка 0,9599**						
№1	x	23311	2474	0	152	0,8988
	$\tilde{x}$	23311	2476,4	0	149,6	0,8988
	Δ		0%		-2%	**0%**
№2	x	2452	265	0	0	0,9025
	$\tilde{x}$	2452	265	0	0	0,9025
	Δ					**0%**
Искажения student_info_auto (fullness-проверки), **качество Data-продукта: точное 0,9617, оценка 0,9618**						
№1	x	23284	2501	0	0	0,9030
	$\tilde{x}$	23284	2501	0	0	0,9030
	Δ					**0%**
№2	x	2466	115	136	0	0,9076
	$\tilde{x}$	2466	115	132,1	0	0,9089
	Δ			-3%		**0,15%**
Искажения student_exam_score (все 3 вида проверок), **качество Data-продукта: точное 0,9312, оценка 0,9303**						
№3	x	1806	0	83	151	0,8853
	$\tilde{x}$	1806,0	0	86,5	151,0	0,8838
	Δ	0%		4%	0%	**-0,17%**
Одновременно все искажения, **качество Data-продукта: точное 0,851, оценка 0,8527**						
№1	x	20976	4809	0	145	0,8089
	$\tilde{x}$	20976	4801,9	0	152,1	0,8089
	Δ		0%		5%	**0%**
№2	x	2204	381	132	0	0,8112
	$\tilde{x}$	2204	381	135,0	0	0,8103
	Δ			2%		**-0,11%**
№3	x	1809	0	80	170	0,8786
	$\tilde{x}$	1817,4	0	82,3	161,6	0,8817
	Δ	0%		3%	-5%	**0,35%**

VI. Заключение

Результаты экспериментального тестирования продемонстрировали, что предложенный в работе алгоритм может быть успешно применён для оценки качества Data-продуктов. Так, несмотря на некоторые отклонения оценок отдельных параметров от точных значений (не превышающих 5%), оценка качества данных таблиц в каждом отдельном выходном порту, полученная в результате работы алгоритма, отличается от точной не более чем на 0,4%, а итоговая оценка качества Data-продукта отличается от точного значения лишь в 3 знаке после запятой.

Важно отметить, что на практике однократный расчёт уровня качества некоторого набора данных зачастую не приносит дополнительной ценности: реальные данные почти всегда содержат искажения и ожидаемое отличие полученного уровня качества данных от 100% само по себе не определяет необходимость управления качеством. Тем не менее, в контексте концепции Data Mesh можно предложить следующие варианты практического применения предложенного выше подхода к оценке качества Data-продукта:

1. Управляя наборами размеченных экземпляров входных данных, доменные команды могут использовать алгоритм оценки для выявления искажений, которые оказывают наибольший вклад в качество Data-продукта. Более низкая оценка качества будет соответствовать большему вкладу, что позволит определить эффективный план по устранению искажений входных данных в зоне ответственности домена и соответствующему повышению качества Data-продукта.
2. Доменные команды также могут проводить регулярный мониторинг качества Data-продукта, результаты которого в динамике можно использовать для анализа результативности действий, предпринимаемых в рамках повышения качества данных.
3. Анализ разных типов искаженных кортежей, полученных для выходного порта в результате работы алгоритма, может быть использован для дополнения или корректировки Data-контрактов. В частности, наличие большого количества кортежей с непустыми значения *track* говорит о частых ошибках в соответствующих атрибутах, наличие большого количества кортежей с типом ‘missed’ указывает на возможную потерю большого объема данных, а большое количество кортежей с типом ‘extra’ – на возможное наличие дубликатов. Эта информация может послужить фундамен-

том для формирования дополнительных требований в Data-контракте.

4. Набор оценок качества, полученных для каждого Data-продукта внутри одной реализации Data Mesh, может быть использован федеративной командой для определения наиболее слабых с точки зрения качества данных узлов общей «сети» Data Mesh, что может быть особенно полезно в случае масштабирования такой «сети».

В рамках потенциального развития предложенного алгоритма можно рассмотреть исследование трансформаций для таких SQL конструкции, как UNION, LEFT/RIGHT JOIN, GROUP BY, оконные функции. Доработка алгоритма в части покрытия указанных случаев позволит существенно расширить область применения рассматриваемого подхода к оценке качества данных.

Библиография

[1] Bode J., Kühl N., Kreuzberger D., Holtmann C. Toward Avoiding the Data Mess: Industry Insights From Data Mesh Implementations // IEEE Access. – 2024. – №12.

[2] Goedegebuure A., Kumara I., Driessen S., Van Den Heuvel W.J., Monsieur G., Tamburri D.A., Nucci D.D. Data Mesh: A Systematic Gray Literature Review // ACM Computing Surveys. – 2024. – №57(1). – С.1–36.

[3] Dehghani Z. How to Move Beyond a Monolithic Data Lake to a Distributed Data Mesh. – 2019. – URL: https://martinfowler.com/articles/data–monolith–to–mesh.html

[4] Dolhopolov A., Castelltort A., Laurent A. Implementing federated governance in data mesh architecture // Future Internet. – 2024. – № 16(4).

[5] Neely M. P. The Product approach to data quality and fitness for use: A Framework for analysis. – 2005. – URL: https://repository.rit.edu/other/555

[6] Ballou D.P, Chengalur–Smith I.N, Wang R.Y. Sample–based quality estimation of query results in relational database environments // IEEE transactions on knowledge and data engineering. – 2006. – № 18(5).

[7] Parssian A., Yeoh W., Ee M.S. Quality–based SQL: specifying information quality in relational database queries // Computer. – 2015. – № 48(9). – С.69–74.

[8] Духовенский С.Е., Никульчев Е.В. Разработка программного и математического обеспечения оценки искажений результатов SQL–запросов в условиях снижения качества данных // Электронный научный журнал «ИТ–Стандарт». – 2025. – № 3. – С. 58–73.

[9] Kuzilek J., Hlosta M., Zdrahal Z. Open University Learning Analytics dataset – Sci. Data 4:170171 – 2017. – doi:10.1038/sdata.2017.171.

[10] Oliveira P., Rodrigues F., Henriques P.R. A formal definition of data quality problems // ICIQ. – 2005.

[11] Гарсиа–Молина Г., Ульман Д., Уидом Д. Реляционная алгебра // Системы баз данных. Полный курс / Пер. с англ. – Москва, Издательский дом "Вильямс" – 2003. – С.203–249.

[12] ГОСТ Р 71484.2-2024 (ИСО/МЭК 5259-2:2024) Искусственный интеллект. Качество данных для аналитики и машинного обучения. Часть 2. Показатели качества данных // Электронный фонд правовых и нормативно-технических документов. – URL: https://docs.cntd.ru/document/1310068315

[13] Wang J., Liu Y., Li P., Lin Z., Sindakis S., Aggarwal S. Overview of data quality: Examining the dimensions, antecedents, and impacts of data quality // Journal of the Knowledge Economy. – 2024. – № 15(1) – С. 1159–1178.

Assessment of the Distortion Propagation in Data Products in the Context of a Relational Data Mesh Architecture

S. E. Dukhovenskiy

***Abstract* — With a decentralized approach to data management named Data Mesh, interaction with data occurs at the level of individual data products. If errors or poor-quality data are used during the preparation of data products, their distribution can significantly increase across replicas, datamarts, services, and etc. This paper examines the quality of a data product in a relational Data Mesh architecture under conditions of detected input data distortions. An approach of constructing such an assessment based on representing data transformations within a data product as a chain of elementary relational algebra operations, such as extended projection, filtering, union, and Cartesian product, and analyzing the propagation of input data distortions as a result of applying these operations is proposed. A concept for "labeling" the source relation is presented, allowing tuples to be divided into types based on the nature of the distortions detected in them, as well as algorithms aimed at tracking the transformation of tuples of different types as a result of applying the above elementary operations.**

A complex algorithm is developed that implements the solution to the considered data product quality assessment problem. As part of the testing, an experimental setup was developed using the Open University open data set and the proposed algorithm was implemented. The experimental results confirmed that the data product quality assessment algorithm can be successfully used within the context of a relational Data Mesh architecture. In conclusion, several practical applications of this assessment for effective data quality management within the Data Mesh concept are proposed.

***Key words* — data quality, Data Mesh, data product, relational algebra.**

REFERENCES

[1] Bode J., Kühl N., Kreuzberger D., Holtmann C. “Toward Avoiding the Data Mess: Industry Insights From Data Mesh Implementations” *IEEE Access*, 2024, no.12.

[2] Goedegebuure A., Kumara I., Driessen S., Van Den Heuvel W.J., Monsieur G., Tamburri D.A., Nucci D.D. “Data Mesh: A Systematic Gray Literature Review” *ACM Computing Surveys*, 2024, no. 57(1), pp.1–36.

[3] Dehghani Z. “How to Move Beyond a Monolithic Data Lake to a Distributed Data Mesh”, 2019, available at: https://martinfowler.com/articles/data–monolith–to–mesh.html

[4] Dolhopolov A., Castelltort A., Laurent A. “Implementing federated governance in data mesh architecture” *Future Internet*, 2024, no. 16(4).

[5] Neely M. P. “The Product approach to data quality and fitness for use: A Framework for analysis”, 2005, available at: https://repository.rit.edu/other/555

[6] Ballou D.P, Chengalur–Smith I.N, Wang R.Y. “Sample–based quality estimation of query results in relational database environments*” IEEE transactions on knowledge and data engineering*, 2006, no. 18(5).

[7] Parssian A., Yeoh W., Ee M.S. “Quality–based SQL: specifying information quality in relational database queries” *Computer*, 2015, no. 48(9), pp.69–74.

[8] Dukhovenskiy S. E., Nikulchev E.V. “Development of software and mathematical framework for estimation of distortions in SQL-query results under conditions of poor-quality data” *IT-Standard*, 2025, no. 3, pp. 58–73.

[9] Kuzilek J., Hlosta M., Zdrahal Z. Open University Learning Analytics dataset, Sci. Data 4:170171, 2017, doi:10.1038/sdata.2017.171.

[10] Oliveira P., Rodrigues F., Henriques P.R. “A formal definition of data quality problems” *ICIQ*, 2005.

[11] Garcia-Molina H., Ullman J., Widom J. *Relation algebra. Database systems: the complete book*, Tr. from Eng., Moscow, Williams Publishing House, 2003, pp. 203–249.

[12] GOST R 71484.2-2024 (ISO/MEK 5259-2:2024) “Artificial intelligence. Data quality for analytics and machine learning. Part 2. Data quality measures” *Electronic fund of legal and regulatory documents*, available at: https://docs.cntd.ru/document/1310068315

[13] Wang J., Liu Y., Li P., Lin Z., Sindakis S., Aggarwal S. “Overview of data quality: Examining the dimensions, antecedents, and impacts of data quality” *Journal of the Knowledge Economy*, 2024, no. 15(1), pp. 1159–1178.

About Authors

S. E. Dukhovenskiy, graduate student, MIREA – Russian Technological University, Moscow, Russia (dukhovenskiy.s.e@edu.mirea.ru).