КОНТУР
CPM-ПЛАТФОРМА

CPM-ПЛАТФОРМА «КОНТУР»

Версия системы: 5.1

Информация о системе

Оглавление

Введение

CPM-платформа «Контур» (далее Платформа) является специализированной программной платформой, обеспечивающей построение CPM-систем для финансово-кредитных организаций на базе корпоративного хранилища данных.

Платформа предназначена для быстрого развертывания хранилищ данных и прикладных решений на его основе.

Платформа может быть использована финансовыми организациями разного масштаба, в том числе многофилиальными банками, финансовыми группами и финансово-промышленными холдингами.

Платформа обеспечивает формирование общего прикладного информационного пространства банка для решения стратегических и тактических задач управления на основе дата-центричного подхода. Объединение данных, поступающих из различных источников в единое корпоративное хранилище данных, происходит на базе предоставляемой Платформой отраслевой модели данных. Структура модели данных соответствует типовым потребностям универсального банка в информационном сопровождении процессов подготовки всех видов регуляторной и управленческой отчетности, бюджетирования, финансового и риск-менеджмента. Модель данных банка аккумулирует 25-летний опыт компании «Интерсофт Лаб» в области построения банковских хранилищ данных, что существенно снижает проектные риски.

CPM-платформа «Контур» предоставляет:

- Инструменты для развертывания корпоративного хранилища данных.
- Модель данных банка.
- Инструменты для настройки бизнес-ориентированных структур Хранилища, обеспечивающих гибкость решений и простоту кастомизации модели данных.
- Единую среду для функционирования и поддержки бизнес-приложений для автоматизации комплекса интегрированных методик управления эффективностью банковской деятельности.

Платформа использует в качестве СУБД Postgres Pro, обеспечивает высокую производительность, масштабируемость и информационную безопасность.

Платформа является базисом для построения полнофункциональной системы риск-ориентированного управления эффективностью банковского бизнеса (RCPM-системы), включающей компоненты для финансовой консолидации, стратегического планирования и прогнозирования, бюджетирования, подготовки финансовой и управленческой отчетности, управления ликвидностью, управления прибыльностью, риск-менеджмента.

В данном документе приводится:

- Общие сведения о Платформе.
- Описание базового функционала Платформы.
- Сведения, необходимые для установки и эксплуатации Платформы.

Термины и определения

CPM\RCPM - платформа «Контур» (Платформа Программный продукт)	Единая инструментально-технологическая среда, предназначенная для построения хранилищ данных и прикладных решений для риск-ориентированного управления бизнесом финансовой организации.
CPM\RCPM-Приложение	Программные компоненты, обеспечивающих решение отдельных категорий бизнес-задач и путем использования данных, накопленных в Хранилище данных «Контур».
ETL (Extraction, Transformation, Load)	Технология загрузки данных в хранилище данных, включающая в себя последовательные этапы извлечения данных из внешних источников, очистку и преобразование данных в соответствии с требованиями бизнес модели, загрузку в хранилище данных
Базовая таблица (БТ)	Метаобъект, обязательный элемент структуры объектов, относящихся к категории «Банк данных», предназначен для организации связей между банками данных
Групповая таблица (ГТ)	Метаобъект, необязательный элемент структуры объектов ряда категорий, предназначен для характеристики записи объекта по его ролевой принадлежности
Иерархия	Метаобъект, необязательный элемент структуры объектов, относящихся к категории «Банк данных», предназначен для организации иерархических связей между записями банка данных
Категория	Разновидность универсальной прикладной структуры в ХД «Контур», состоящая из обязательного набора определенным образом связанных метаобъектов
Контекст	Шаблон отображения объекта в интерфейсах системы и прикладных приложений, формируемый на основе представления объекта в БД.
Метаданные	Структурированные данные, характеризующие объекты-носители информации, и использующиеся для идентификации, оценки и управления этими объектами.
Метаобъект	Типовой элемент структуры базы данных, использующийся для описания матаданных
Навигатор	Индексы базы данных, предназначенные для фильтрации записей по атрибутам
Риск-ориентированное управление эффективностью бизнеса, Risk-based	Технология управления эффективностью бизнеса, набор аналитических и управленческих процессов, которые поддерживаются соответствующими информационными технологиями и помогают интегрировать управление рисками

Corporate Performance Management (RCPM)	с задачами стратегического управления, финансового планирования и управления прибыльностью
Управление эффективностью бизнеса, Corporate Performance Management (CPM)	Технология управления эффективностью бизнеса, набор аналитических и управленческих процессов, которые поддерживаются соответствующими информационными технологиями и включают финансовое и операционное планирование, консолидацию и отчетность, моделирование, анализ и мониторинг ключевых показателей эффективности.
Типовая таблица (ТТ)	Метаобъект, необязательный элемент структуры объектов, относящихся к категории «Банк данных», предназначен для отнесения записей банка данных в определенному прикладному типу
ХД «Контур»	Хранилище данных CPM-платформы «Контур»
Хранилище данных (ХД)	Предметно-ориентированная корпоративная база данных, предназначенная для подготовки отчётов и анализа бизнес-данных с целью поддержки принятия управленческих решений.
Центральная таблица (ЦТ)	Метаобъект, обязательный элемент структуры объектов всех категорий, предназначен для хранения всех записей объекта.

1 Общие сведения о программном продукте

1.1 Логическая архитектура Платформы

Логическая архитектура Платформы представлена следующими подсистемами:

1. Подсистема управления метаданными и метаобъектами;
2. Подсистема управления данными;
3. Подсистема управления импортом данных;
4. Подсистема администрирования и информационной безопасности.

Каждая подсистема представляет собой структуру, предназначенную для выполнения группы взаимосвязанных функций, и предоставляет сервисы, востребованные другими компонентами платформы и\или CPM\RCPM-приложениями.

Каждая подсистема представляет собой набор базовых компонент, обеспечивающих работу ядра платформы, и дополнительных модулей. Базовые компоненты инкапсулируют абстрактную функциональность подсистемы, дополнительные модули, обеспечивают дополнительную прикладную функциональность в рамках подсистемы, или предоставляют интерфейс к функциональности базовых компонент.

Обобщенная схема, представляющая логическую структуру платформы, представлена на Рис. 1.

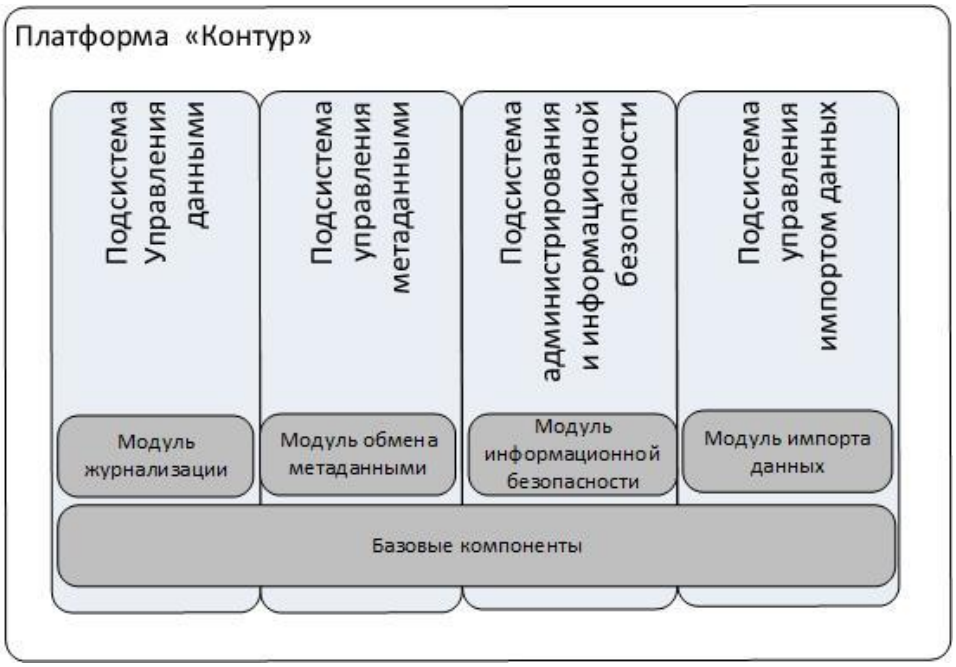

Рис. 1. Подсистемы и компоненты Платформы

Подсистема управления метаданными и метаобъектами предоставляет функционал для работы с моделью данных Хранилища и обеспечивает:

- Работу с метаданными, в том числе создание и поддержку словаря метаданных;
- Работу с прикладными метаобъектами;

- Автоматическую генерацию прикладных структур в Хранилище данных и структур временного хранения в Staging Area на основании метаописания при создании (изменении) метаобъектов;

Компоненты подсистемы позволяют автоматически создавать на базе логической модели данных следующие объекты:

- физические таблицы БД для постоянного и временного хранения данных;
- системные и прикладные связи между объектами;
- аналитические выборки;
- процедуры для отбора, ввода, удаления и изменения данных Хранилища;
- средства навигации, обеспечивающие быстрый поиск и выборку данных в Хранилище;
- форматы XML-файлов, используемых для описания метаданных.

Платформа реализует стандартизованный подход к описанию бизнес-объектов, позволяющий производить настройку метаданных и устанавливать связи между объектами в соответствии с прикладным содержанием задачи.

Подсистема управления метаданными поддерживает также универсальный механизм обогащения данных (механизм классификаций), используемый большинством CPM\RCPM-приложений. Системные инструменты классификации обеспечивают настройку алгоритмов и правил обогащения данных, автоматическую генерацию процедур классификации по заданному метаописанию механизма, запуск процедур классификации.

Платформа предоставляет правила декларативного описания мета-объектов различного назначения, правила обращения к генерированным процедурам и правила взаимодействия с данными временной и постоянной областей хранения. В состав подсистемы включен модуль обмена метаданными, который позволяет импортировать и экспортировать xml-метаописания ХД или его части. Подсистема «Управление метаданными и метаобъектами» проверят импортированные метаописания ХД и сохраняет в словарях системы, а также производит генерацию таблиц, процедур и функций для подсистем управления данными и управления импортом данных.

Технология обмена метаданными позволяет более эффективно решать такие задачи как:

- Создание новых приложений системы.
- Хранение версий словарей метаданных
- Обмен словарями метаданных

Подсистема управления данными обеспечивает работу с данными Хранилища, в том числе:

- Выполнение операций создания, изменения, физического и логического удаления записей в таблицах ХД;
- Предоставление выборки данных по запросу пользователя;
- Выполнения операций по управлению данными в иерархических структурах;
- Журнализацию данных.

Подсистема импорта обеспечивает загрузку данных в ХД из внешних источников.

Импорт данных осуществляется через область временного хранения (Stage), что позволяет организовать двухфазную загрузку:

- 1 фаза – заполнение stage-структур,
- 2 фаза - перенос импортированных данных в область постоянного хранения с выполнением процедур проверки качества, очистки, трансформации и обогащения данных. Все манипуляции с данными на этапе загрузке могут выполняться как стандартными, включенными в пакет поставки процедурами, так и пользовательскими процедурами.

Входящие в подсистему импорта компоненты обеспечивают:

- Выполнение обязательных системных проверок качества данных;
- Возможность подключения любых пользовательских процедур проверки и обработки данных на отдельные этапы загрузки.
- Ведение протоколов загрузки.

Возможность управления процессом загрузки и детальная журнализация всех операций системы позволяет оптимизировать технологию сбора данных в ХД в каждом проекте.

Подсистема информационной безопасности позволяет управлять пользователями, ролями и привилегиями, в соответствии с выбранной политикой безопасности, а также вести аудит действий пользователей.

Права пользователей разграничиваются на уровне объектов БД (таблиц, функций, процедур, пакетов) и объектов Платформы (команд и источников данных).

Платформа предлагает ряд тиражных системных ролей, которые могут быть использованы в процессе администрирования пользователей Системы.

1.2 CPM\RCPM-система на платформе «Контур»

Платформа «Контур» является технологической основой для построения комплекса CPM\RCPM-приложений и обеспечивает формирование общего прикладного информационного пространства, которое используют приложения, входящие в состав полнофункциональной RCPM-Системы.

CPM\RCPM -Система на платформе «Контур» включает следующие компоненты:

- Платформа Хранилищ данных «Контур»;
- Приложения и модули модели данных банка;
- RCPM–приложения «Контур».

Приложения и модули модели данных содержат элементы базового словаря метаданных, востребованные CPM\RCPM -приложениями.

CPM\RCPM-приложения, развернутые на Платформе, предназначены для автоматизации комплекса задач риск-ориентированного управления банком и подготовки всех видов банковской отчетности, в том числе:

- финансового планирования,
- прогнозирования и моделирования,
- подготовки управленческой отчетности,
- аллокации расходов,
- трансфертного управления ресурсами,
- управления рисками,
- подготовки отчетности для Банка России,
- обмена данными с БКИ и др.

Вариант архитектуры CPM\RCPM -Системы на платформе «Контур» представлен на Рис. 2.

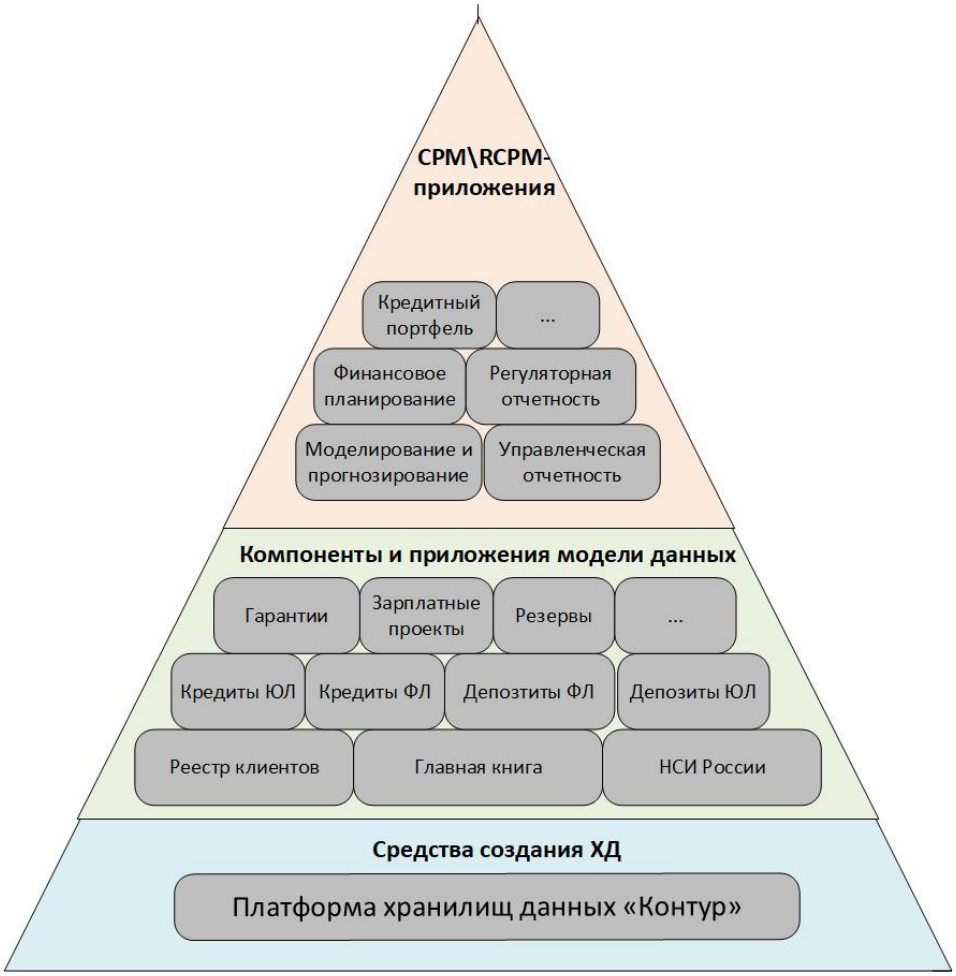

Рис. 2. Архитектура RCPM-системы на платформе «Контур»

CPM\RCPM-Система может включать произвольный набор приложений, достаточный для поддержки выборочных управленческих технологий (в соответствии с потребностями конкретного кредитного учреждения).

Основными категориями задач, решаемыми прикладными решениями Системы, являются:

- Сбор и консолидация финансовой и нефинансовой информации из различных источников в корпоративном хранилище данных.
- Автоматизация управленческих методик с использованием данных Хранилища.
- Формирование регулятивной отчетности в соответствии с различными стандартами учета (национальными, международными, отраслевыми).

Архитектура системы позволяет проводить внедрение полнофункционального решения поэтапно. При этом автоматизация отдельных этапов решается как самостоятельная задача.

2 Базовая функциональность Платформы

2.1 Управление репозиторием метаданных

2.1.1 Принципы описания метаданных

Прикладные структуры ХД создаются на основе единого подхода, позволяющего:

- Описывать произвольные бизнес-объекты, используя ограниченный набор стандартизованных метаобъектов.
- Легко проводить кастомизацию логической модели данных за счет добавления пользовательских атрибутов и установления связей между объектами.
- При необходимости сохранять историю изменения атрибутов метаобъектов.

2.1.2 Категории объектов хранения

В Системе используется понятие категории объектов хранения, которое было введено для унификации структур Хранилища данных, предназначенных для описания бизнес-объектов различной природы.

Используется пять **категорий объектов** хранения:

1. ***Словарь (словарный банк данных)*** – структура, предназначенная для хранения простейших плоских справочников, не требующих ведения истории изменения атрибутов.
2. ***Банк данных (пользовательский банк данных)*** – универсальная структура, предназначенная для хранения объектов любого назначения (планы показателей, сложные справочники, учетные регистры и др.).
3. ***Транзакционная таблица (транзакционный банк данных)*** – структура, предназначена для хранения информации о событиях, произошедших на определенную дату (операции и обороты по счетам, бухгалтерские проводки и др.).
4. ***Интервальная таблица (интервальный банк данных)*** – структура, предназначена для хранения значений, отражающих состояние прикладных объектов на определенном временном интервале (остатки на счетах, значения нормативных показателей, котировки валют и др.).
5. ***Витрина данных –*** структура, предназначенная для хранения больших объемов данных, оптимизирована для быстрого выполнения сложных аналитических запросов.

Любая структура в Хранилище данных принадлежит к одной из вышеперечисленных категорий.

Описание структуры объектов ХД, независимо от их принадлежности к той или иной категории, производится в соответствии со следующими принципами:

- Объекты строятся из набора «атомарных» метаобъектов, являющихся базовыми элементами структуры.
- «Атомарные» метаобъекты представляют собой отдельные таблицы, с определенным прикладным назначением и предопределенным принципом организации. Основные характеристики метаобъектов представлены ниже (Таблица 1).

Для каждой категории предопределена схема связи метаобъектов, а также состав системных атрибутов метаобъектов. В рамках каждой категории определены обязательные элементы структуры и дополнительные элементы, необходимость использования которых определяется прикладным назначением объектов (Гибкость прикладных структур Хранилища данных обеспечивается за счет:

- Кастомизации атрибутов переменной части центральной таблицы,

  Примечание

  Атрибуты переменной части центральной таблицы специфичны для каждого банка данных (словаря, интервальной или транзакционной таблицы или витрины данных) и одинаковы для всех объектов банка данных (словаря, интервальной или транзакционной таблицы).

- Использования неограниченного количества типов (для банка данных) и кастомизации атрибутов переменной части каждой типовой таблицы,

  Примечание

  Для хранения атрибутов каждого типа резервируется отдельная типовая таблица.

  Если при настройке банка данных (интервальной таблицы) для него был сформирован перечень типов, то любой объект этого банка данных может быть отнесен только к одному типу. Отнесение к типу не является обязательным (объект может не иметь типовой принадлежности).

- Использования неограниченного количества групп и кастомизации атрибутов переменной части каждой групповой таблицы.

  Примечание

  Для хранения атрибутов каждой группы резервируется отдельная таблица.

  Если при настройке банка данных (интервальной или транзакционной таблицы) был сформирован перечень групп, то любой объект этого банка данных может быть отнесен к одной или нескольким группам. Отнесение к группе не является обязательным.

- Настройки произвольного количества иерархий (для банков данных и интервальных таблиц).
- Возможности историзации значений атрибутов банков данных.

Разрешить/отменить любой элемент в структуре банка данных (интервальной или транзакционной таблицы) можно независимо от наличия данных в банке данных при наличии соответствующих прав у пользователя.

Примечание

Удаление элементов структуры выполняется только в том случае, если это не приводит к нарушению ссылочной целостности данных.

- Таблица 2).
- При настройке прикладной структуры объявляется вид категории и состав метаобъектов, допустимый в рамках заданной категории, а также описываются пользовательские (прикладные) атрибуты метаобъектов.

В ХД может быть настроено любое количество банков данных, словарей, интервальных и транзакционных таблиц.

Система автоматически формирует необходимые связи между метаобъектами в рамках категории при сохранении банка данных, транзакционной или интервальной таблицы.

Таблица 1. Характеристики «атомарных» метаобъектов

Наименование метаобъекта	Назначение	Примечание
Базовая таблица	Таблица, предназначенная для организации связей между банками данных.	Настраиваемых атрибутов не содержит
Центральная таблица	Таблица, содержащая все записи, относящиеся к заданной категории хранения.	В структуре центральной, типовой и групповой таблиц выделяют **постоянную** и **переменную** части. **Постоянная часть** содержит предустановленные системные атрибуты, в том числе атрибуты для хранения истории изменения атрибутов. **Переменная часть** содержит настраиваемые пользователем прикладные атрибуты. Количество атрибутов переменной части не ограничено.
Типовая таблица	Таблица, содержащая набор атрибутов, характеризующих объект по его принадлежности к определенному прикладному типу.	
Групповая таблица	Таблица, содержащая набор атрибутов, характеризующих объект по его ролевой принадлежности.	
Иерархия	Таблицы, в которых организованы иерархические связи между записями объекта хранения	Для хранения каждой иерархии создается по три таблицы: • таблица с классической организацией хранения списка смежных графов по типу каждый родитель с каждым подчиненным; • таблица, в которой помимо классической организации хранения, организованы связи узлов и всех их подчиненных листьев (для отображения иерархии в интерфейсе системы); • таблица, в которой узлы и листья иерархии развернуты в столбцы (для построения OLAP отчетов).
Навигатор	Индекс БД, позволяющий оптимизировать выполнение запросов по заданным атрибутам.	Допускается построение навигаторов по центральной таблице, типу или группе.

Гибкость прикладных структур Хранилища данных обеспечивается за счет:

- Кастомизации атрибутов переменной части центральной таблицы,

  Примечание

  Атрибуты переменной части центральной таблицы специфичны для каждого банка данных (словаря, интервальной или транзакционной таблицы или витрины данных) и одинаковы для всех объектов банка данных (словаря, интервальной или транзакционной таблицы).

- Использования неограниченного количества типов (для банка данных) и кастомизации атрибутов переменной части каждой типовой таблицы,

  Примечание

  Для хранения атрибутов каждого типа резервируется отдельная типовая таблица.

  Если при настройке банка данных (интервальной таблицы) для него был сформирован перечень типов, то любой объект этого банка данных может быть отнесен только к одному типу. Отнесение к типу не является обязательным (объект может не иметь типовой принадлежности).

- Использования неограниченного количества групп и кастомизации атрибутов переменной части каждой групповой таблицы.

  Примечание

  Для хранения атрибутов каждой группы резервируется отдельная таблица.

  Если при настройке банка данных (интервальной или транзакционной таблицы) был сформирован перечень групп, то любой объект этого банка данных может быть отнесен к одной или нескольким группам. Отнесение к группе не является обязательным.

- Настройки произвольного количества иерархий (для банков данных и интервальных таблиц).
- Возможности историзации значений атрибутов банков данных.

Разрешить/отменить любой элемент в структуре банка данных (интервальной или транзакционной таблицы) можно независимо от наличия данных в банке данных при наличии соответствующих прав у пользователя.

Примечание

Удаление элементов структуры выполняется только в том случае, если это не приводит к нарушению ссылочной целостности данных.

Таблица 2. Структура базовых категорий Системы

Категория	**Характеристики структуры**			
	Элементы структуры	**Обязательность**	**Кол-во**	**Примечание**
Банк данных	Базовая таблица	Да	1	Имеется возможность **хранения истории** изменения атрибутов центральной, типовой и групповой таблиц (опция может быть включена\выключена для каждого банка данных). Имеется возможность **загрузки данных через область временного хранения** (опция загрузки через Staging Area может быть включена\выключена для каждого банка данных).
	Центральная таблица	Да	1	
	Типовая таблица	Нет	Любое	
	Групповая таблица	Нет	Любое	
	Иерархия	Нет	Любое	
	Навигатор	Нет	Любое	
Словарь	Центральная таблица	Да	1	Загрузка данных через область временного хранения **не поддерживается**.
Интервальная таблица	Центральная таблица	Да	1	Хранение истории изменения атрибутов центральной и групповой таблиц является обязательным для данной категории хранения. Имеется возможность загрузки данных через область временного хранения (опция загрузки через Staging Area может быть включена\выключена для каждой таблицы).
	Групповая таблица	Нет	Любое	
	Навигатор	Да	Любое	
Транзакционная таблица	Центральная таблица	Да	1	Имеется возможность загрузки данных через область временного хранения (опция загрузки через Staging Area может быть включена\выключена для каждой таблицы).
	Групповая таблица	Нет	Любое	

Категория	Характеристики структуры			
	Элементы структуры	**Обязательность**	**Кол-во**	**Примечание**
	Навигатор	Да	Любое	
Витрина данных	Центральная таблица	Да	1	Центральная таблица используется для хранения значений аналитических измерений.
	Транзакционная таблица	Нет[1]	Любое	Транзакционная таблица используется для хранения значений факта.
	Интервальная таблица	Нет	Любое	Интервальная таблица используется для хранения значений факта, действующих на определенных временных интервалах.
	Навигатор	Да	N^2+1	Количество навигаторов и состав индексируемых полей строго регламентирован.

2.1.3 Связи между объектами системы

В ходе настройки структур ХД часто возникает необходимость в организации связей между объектами в соответствии с требованиями предметной области.

Атрибуты переменной части центральных, типовых и групповых таблиц одной категории могут быть ссылками на объекты другой категории (т. е. являются ссылочными атрибутами).

Для категорий различных видов существуют собственные требования к организации связей. Допустимость использования ссылок на объекты различных категорий в ссылочных атрибутах банков данных, словарей, интервальных и транзакционных поддерживается системой.

2.1.4 Принципы хранения исторической информации

Одним из ключевых принципов организации ХД является привязка собираемых данных к временной шкале, чем обеспечивается возможность анализа временных зависимостей. Помимо этого, упорядочивание значений атрибутов по времени ускоряет выполнение запросов к БД.

В соответствии с характером собираемой в Хранилище информации возникает необходимость отразить два вида временных изменений:

1. Для объектов, значения атрибутов которых остаются постоянными на определенном временном интервале, необходимо зафиксировать границы временного интервала (последовательные моменты изменения значений атрибута) и значение атрибута на каждом интервале. Примерами объектов такого вида являются остатки лицевых счетов, котировки валют, показатели планов.
2. Если для объекта принципиальное значение имеет регистрация факта изменения значения атрибута, то в ХД достаточно зафиксировать дату изменения и значение атрибута на момент изменения. Примерами объектов такого вида являются обороты по счетам, бухгалтерские документы, операции по счетам.

[1] *Обязательным является наличие хотя бы одной интервальной или транзакционной таблицы*

[2] *Где N – количество прикладных атрибутов центральной таблицы витрины данных*

На основании вышеизложенных требований к организации хранения исторических данных в системе используется два подхода к организации хранения истории значений атрибутов:

1. ***Интервальный принцип*** – основан на построении непрерывного временного ряда, составленного из упорядоченных по времени интервалов, на каждом из которых актуально определенное значение атрибута.
2. ***Событийный принцип*** – основывается на регистрации сведений о событии, произошедшем в определенную календарную дату.

2.1.4.1Интервальный принцип историзации значений атрибутов

Интервальный принцип используется для и объектов, относящихся к следующим категориям:

- Банки данных, для которых возможно хранение истории атрибутов центральной таблицы, типовых и групповых таблиц. Опция хранения истории может быть разрешена или запрещена по каждому конкретному банку данных. Для каждой групповой таблицы банка данных необходимость историзации атрибутов определяются отдельно.
- Интервальные таблицы, для которых хранение истории атрибутов центральной таблицы, типовых и групповых таблиц является обязательным и выполняется по умолчанию.

Для каждого историзируемого атрибута формируется непрерывная временная шкала, составляется из интервалов следующего вида [Дата начала, Дата окончания], где

Дата начала – дата i-того изменения значения атрибута,

Дата окончания – дата (i+1)-го изменения атрибута.

Шкала времени начинается с момента регистрации объекта в системе и заканчивается максимальной системной датой. Если опция хранения истории для банка данных не используется, то атрибут «Дата начала» заполняется системной начальной датой, а атрибут «Дата окончания» - системной конечной датой.

Интервальный принцип хранения истории атрибутов позволяет, например, оптимизировать хранение остатков по лицевым счетам и существенно сократить объем Хранилища данных, поскольку отпадает необходимость в хранении остатков на каждую системную дату.

2.1.4.2Событийный принцип историзации

Событийный принцип хранения информации используется исключительно для объектов, относящихся к категории «транзакционные таблицы».

Для каждого вида событий (для каждой транзакционной таблицы) определяется уникальный набор атрибутов, включающий дату события, который позволяет однозначно идентифицировать каждое отдельное событие. Такой набор атрибутов представляет собой альтернативный ключ для транзакционной таблицы.

Примерами объектов, для которых целесообразно использовать событийных принцип хранения истории являются: бухгалтерские документы, обороты и операции по счетам, учетные и др. журналы (в том числе журналы регистрации действий пользователя в системе, журналы загрузки, журнал классификаций или расчетов и т.п.).

Система поддерживает следующие операции настройки структуры ХД:

- Создание объектов пяти базовых категорий, описание структуры новых объектов.
- Изменение структуры существующих объектов.
- Удаление объектов ХД.

2.1.5 Настройка структуры объектов ХД

Объект создается в соответствующем разделе репозитория метаданных.

В ходе настройки должны быть определены прикладные атрибуты объекта и задана его структура в рамках выбранной категории.

Настройка любого объекта условно разбивается на два этапа:

1. **На первом этапе** производится создание объекта, при этом пользователем указывается:
   1.1. вид категории,
   1.2. значения обязательных атрибутов (код категории, наименование категории, SQL-имя объекта и др.).
   1.3. включается\выключается опция хранения истории атрибутов (только для банков данных),
   1.4. включается\выключается опция использования области временного хранения.

   Примечание

   В процессе сохранения созданного объекта система автоматически генерирует необходимые таблицы и связи между ними в Хранилище данных, а также необходимые структуры в Staging Area (если опция использования области временного хранения была включена) и процедуры.

   Порядок создания объектов одинаков для объектов всех категорий.

2. **На втором этапе** детально настраивается структура объекта:
   2.1. Формируется перечень атрибутов переменной части центральной таблицы,
   2.2. Формируется перечень типов и задаются атрибуты переменной части типовой таблицы для каждого типа.
   2.3. Формируется перечень групп и задаются атрибуты переменной части групповой таблицы для каждой группы.
   2.4. Формируется перечень иерархий и задаются ранги каждой иерархии.
   2.5. Задаются навигаторы.

   Примечание

   Внесение изменений в структуру объекта требует перегенерации структур ХД и структур временного хранения. Эта операция выполняется по команде пользователя (возможен откат внесенных изменений).

   Настройка структуры объектов различных категорий имеет свои особенности.

Состав обязательных и опциональных элементов структуры для объектов различных категорий представлен в разделе 2.1.2.

Операции по настройки структуры объектов включают следующие действия:

- Добавление \ удаление \ редактирование прикладных атрибутов центральной таблицы (ЦТ),
- Добавление \ удаление \ редактирование групп и прикладных групповых атрибутов
- Добавление \ удаление \ редактирование типов и прикладных типовых атрибутов
- Добавление \ удаление \ редактирование иерархий
- Добавление \удаление \ редактирование навигаторов

Возможность добавления \ удаления \ редактирования отдельных элементов структуры определяется состоянием объекта. Объект может находиться либо в стабильном (N) либо в нестабильном состоянии. Причем существует несколько видов нестабильных состояний для каждой категории объектов. Объект переходит из состояния в состояние при выполнении определенных действий по настройке его структуры. Объект переходит в стабильное состояние только после успешного сохранения метаописания и создания физических объектов ХД (после выполнения генерации таблиц и процедур ХД по команде пользователя).

Перечень возможных нестабильных состояний для базовых категорий метаобъектов представлен в таблице.

Категория	Возможные нестабильные состояния
Пользовательский банк данных	CREATEBANK CREATEATTR CREATEOBJECT CREATENAVIGATOR CREATEHIERARHY DELETEBANK
Словарный банк данных	CREATEBANK CREATEATTR CREATENAVIGATOR CREATEHIERARHY DELETEBANK
Интервальный банк данных, Транзакционный банк данных	CREATEBANK NONAVIGATOR CREATEATTR CREATEOBJECT CREATENAVIGATOR DELETEBANK

В Приложении 1 представлены диаграммы состояний для базовых категорий ХД.

Для объектов, находящихся в нестабильном состоянии, доступно добавление \ редактирование прикладных атрибутов центральной таблицы, а также добавление \ редактирование иерархий.

Для объектов, находящихся в стабильном состоянии доступно выполнение настройки всех элементов структуры, допустимых для соответствующей категории.

В Системе поддерживается единая технология работы с объектами всех категорий.

Настройка матаописания банка данных включает следующие этапы:

1. Описание атрибутов переменной части центральной таблицы;
2. Описание типов и прикладных атрибутов каждой типовой таблицы;
3. Описание групп и прикладных атрибутов каждой групповой таблицы;
4. Настройка иерархий и создание уровней для каждой иерархии;
5. Настройка навигаторов и свойств каждого навигатора.

Настройка метаописания словарей сводится к выполнению следующих действий:

1. Настройка прикладных атрибутов центральной таблицы (обязательный этап).
2. Настройка иерархий (не обязательный этап).

3. Настройка навигаторов (не обязательный этап).

Настройка метаописания транзакционной таблицы включает следующие этапы:

1. Описание прикладных атрибутов центральной таблицы (обязательный этап),
2. Настройка групп и описание прикладных атрибутов для каждой групповой таблицы (не обязательный этап),
3. Настройка навигаторов (обязательный этап).

Особенностью настройки транзакционных таблиц является обязательное создание уникального навигатора, используемого в качестве альтернативного ключа таблицы.

Внимание

Транзакционный банк данных, не имеющий в своем составе уникального навигатора, или имеющий навигатор, не удовлетворяющий правилам построения навигаторов интервальных банков данных для ХД «Контур», имеет статус NoNavigator (находится в нестабильном состоянии) и не может быть использован для заполнения данными.

Построение уникального навигатора выполняется в соответствии со следующими правилами:

- Для каждого транзакционного банка данных может быть определен только один навигатор - альтернативный ключ.
- Навигатор строится по системным и\или прикладным атрибутам центральной таблицы.
- Атрибуты, включенные в навигатор, должны являться обязательными (не могут принимать значение NULL).
- В состав навигатора может быть включен единственный системный атрибут «Дата начала», но его участие в составе ключа не является обязательным.
- Уникальный навигатор интервального банка данных всегда имеет sql-имя AK1.

Важно

Ссылка на транзакционную таблицу из других категорий может быть организована только в том случае, если уникальный навигатор построен по одному атрибуту.

Помимо уникального навигатора в структуре транзакционной таблицы может быть настроено любое количество навигаторов по системным и прикладным атрибутам центральной и групповых таблиц.

Настройка метаописания интервальной таблицы включает следующие этапы:

1. Описание прикладных атрибутов центральной таблицы,
2. Настройка групп и описание прикладных атрибутов для каждой групповой таблицы, (не обязательный этап).
3. Настройка навигаторов.

Особенностью настройки интервальных таблиц является обязательное создание уникального навигатора, используемого в качестве альтернативного ключа таблицы.

Внимание

Интервальный банк данных, не имеющий в своем составе уникального навигатора, или имеющий навигатор, не удовлетворяющий правилам построения навигаторов интервальных банков данных для ХД «Контур», имеет статус NoNavigator (находится в нестабильном состоянии) и не может быть использован для заполнения данными.

Конструкция уникального навигатора определяется интервальным принципом хранения данных, принятым для объектов данной категории. Альтернативный ключ состоит из двух логических составляющих:

- Аналитическая составляющая – набор атрибутов центральной таблицы, необходимых для однозначной идентификации состояния объекта на временном интервале.
- Временная составляющая – атрибут «Дата начала», необходимый для фиксации даты начала действия состояния.

Построение уникального навигатора выполняется в соответствии со следующими правилами:

- Для каждого интервального банка данных должен быть определен единственный уникальный навигатор - альтернативный ключ.
- Навигатор строится по системным и\или прикладным атрибутам центральной таблицы.
- Атрибуты, включенные в навигатор, должны являться обязательными (не могут принимать значение NULL).
- В состав уникального навигатора обязательно должен быть включен системный атрибут «Дата начала».
- Уникальный навигатор интервального банка данных всегда имеет sql-имя AK1.

Помимо уникального навигатора в структуре интервальной таблицы может быть настроено произвольное количество навигаторов по системным и прикладным атрибутам центральной и групповых таблиц.

Например, для каждого интервала центральной таблицы в групповой таблице может соответствовать несколько записей, разбитых на собственные интервалы.

2.2 Организация импорта данных

2.2.1 Структура хранилища данных

Особенностью архитектуры северной части системы является выделение в Хранилище данных двух физически изолированных областей: области временного хранения (Staging Area) и области постоянного хранения (DWH).

Область временного хранения выполняет роль шлюза при загрузке в ХД данных из любых внешних источников (АБС, локальных баз данных, файлов, систем автоматизации) и предназначена для верификации и очистки собираемых в Хранилище данных. Данные, загружаемые в Хранилище, размещаются в таблицах временного хранения, откуда после необходимых системных и бизнес-проверок перемещаются в DWH.

В область постоянного хранения переносятся только корректные данные из Staging Area, именно эти данные доступны для последующего аналитического использования.

Физически единая область постоянного хранения (DWH) может быть разбита на три логические части:

- *Хранилище данных первичного учета (Data Warehouse)* – предназначено для хранения транзакционных данных, импортированных из систем первичного учета. Здесь собираются данные бухгалтерского учета, данные позиционного учета, бухгалтерские корректировки и т.п.
- *Область аналитических данных (Master Data Area)* – область, предназначенная для хранения аналитических измерений.
- *Витрины данных (Data Marts)* – область, предназначенная для хранения информации, подготовленной для построения отчетов и выполнения анализа деятельности банка.

2.2.2 Область временного хранения

Массовая загрузка данных в Хранилище всегда осуществляется через область временного хранения, что обеспечивает дополнительный контроль качества данных, предназначенных для анализа.

Область временного хранения содержит специализированные структуры временного хранения, которые генерируются автоматически при настройке Хранилища данных. Структура таблиц временного хранения формируется на основании метаописания соответствующего банка данных (интервальной или транзакционной таблицы) и дополняется набором служебных атрибутов. Помимо этого, для Stage-таблицы создается отдельная таблица регистрации ошибок (таблица протокола качества данных).

В области временного хранения выполняется ряд процедур обработки данных, обеспечивающих контроль их качества, а также перенос корректных данных в область постоянного хранения, постзагрузочная обработка данных, восстановление отложенных ссылок и др. технологические процессы.

Процедуры генерации таблиц временного хранения и таблиц ошибок запускаются каждый раз после успешного сохранения структуры объекта базовой категории хранения. При этом возможны следующие действия:

- Создание области временного хранения (процедура запускается для вновь созданного объекта Хранилища данных или для объекта, не имевшего ранее признака «Загрузка через Staging Area»).
- Перегенерация таблиц области временного хранения и таблиц ошибок (процедура запускается при изменении настроек объекта базовой категории, имевшего ранее признак «Загрузка через Staging Area»).
- Удаление таблиц области временного хранения и таблиц ошибок (процедура запускается в случае отмены признака «Загрузка через Staging Area» для объекта, имевшего ранее этот признак).

  Примечание

  В этом случае удаляются сведения о процедурах контроля, а также сведения о банке данных из порядка загрузки и общего регламента процесса загрузки.

  Удаление структур временного хранения не может быть выполнено, если запущен или активен, хотя бы один процесс загрузки.

В области временного хранения находятся физически несвязанные между собой таблицы временного хранения (Stage-таблицы), таблицы регистрации ошибок и другие вспомогательные таблицы.

Отдельные Stage-таблицы генерируются для базовых, центральных, типовых, групповых таблиц и иерархий. Структура таблиц временного хранения полностью воспроизводит структуру переменной части таблиц соответствующего объекта Хранилища данных (содержат все настроенные в ХД прикладные атрибуты метаобъекта). Помимо этого, Stage-таблицы содержат обязательные системные атрибуты, обеспечивающие поддержку технологии загрузки (идентификатор сеанса загрузки, флаг загрузки, код загружаемой записи и др.) Исключение составляют Stage-таблицы иерархий, которые имеют фиксированную структуру и не содержат прикладных атрибутов.

Таблицы временного хранения заполняются данными из внешних источников с помощью соответствующих системных утилит или специализированных ETL-инструментов.

Каждой таблице временного хранения соответствует своя таблица регистрации ошибок, которая заполняется процедурами проверки качества данных.

В таблицу регистрации ошибок заносится информация об идентификаторе ошибки, сеансе загрузки, статусе ошибки, загружаемой категории, коде ошибочной записи, процедуре обработки, обнаружившей ошибку, а также текст сообщения об ошибке. Система диагностирует два типа ошибок: предупреждения (Warnings) и ошибки (Errors).

Записи с ошибками, считаются не прошедшими контроль качества, загрузка таких записей в область постоянного хранения запрещена. Записи с предупреждениями, загружаются после загрузки основного массива данных и используются для корректной обработки отложенных ссылок.

2.2.3 Фазы и этапы импорта данных

В подсистеме импорта предопределен перечень этапов обработки и контроля данных[3], выполняемых в процессе импорта данных из внешних источников, и зафиксирована последовательность выполнения этапов в рамках стандартного сеанса загрузки.

В процессе загрузки выполняется ряд обязательных системных проверок качества данных, в том числе:

- Контроль дублирования кодов,
- Контроль заполнения обязательных полей,
- Контроль неоднозначности типовых объектов,
- Контроль даты актуальности,
- Контроль несоответствия рангов иерархии.

Для обеспечения целостности и непротиворечивости данных подсистема импорта проводится анализ ссылочных атрибутов:

- При обнаружении записи, значение ссылочного атрибута в которой ссылается на несуществующую запись в соответствующем банке данных в DWH (за исключением банка данных типа «Словарь»), системой формируется мнимый объект для несуществующей записи. Запись, ссылающаяся на мнимый объект, загружается в DWH.
- При обнаружении в типовой, групповой таблицах или в таблицах иерархий записи с кодом, отсутствующим в центральной таблице в областях Stage и DWH, загрузка такой записи в DWH не производится.
- При обнаружении записи, значение ссылочного атрибута для которой равны NULL, для таких атрибутов устанавливается значение неопределенного объекта с ID=-1.

Перечень этапов загрузки и последовательность их выполнения представлены в Табл. 1.

Все этапы обработки данных делятся на две группы, соответствующие двум фазам загрузки данных в ХД:

- Этапы, выполняемые в процессе загрузки данных **из внешних источников в Staging Area** (этапы 2.1-2.4),
- Этапы, выполняемые в процессе переноса данных **из Staging Area в DWH** (этапы 3.1-3.6).

Этапы выполняются последовательно в порядке, указанном в таблице.

Следует различать:

- **Системные этапы** – этапы, на которых выполняются регламентные технологические операции по загрузке данных, а также обязательные системные проверки.
- **Пользовательские этапы** – это этапы загрузки, на которых могут быть подключены любые прикладные (пользовательские) процедуры обработки данных. Использование прикладных процедур обработки данных не является обязательным, и определяется бизнес-логикой конкретного решения. **По**

[3] Перечень этапов и последовательность их выполнения могут быть изменены только специалистами системной разработки компании «Интресофт Лаб».

умолчанию никаких действий с данными на пользовательских этапах система не производит.

Табл. 1. Перечень этапов загрузки данных

Очередность выполнения этап\подэтапов	Код этапа\подэтапа	Наименование этапа \ подэтапа	Системный этап (Да\Нет)	Описание этапа
1.	**SESSION**	**Старт сеанса импорта данных**	Да	Выполняется контроль основных и **дополнительных параметров сеанса и регистрация нового сеанса импорта в журнале сеансов им**порта.
2.	**STG_LOAD**	**Загрузка данных в Staging Area**		Этап выделяется для определения статистики загрузки данных в Staging Area, этап состоит их четырех подэтапов (2.1, 2.2, 2.3 и 2.4).
2.1.	ENROLL_PARAMS	Регистрация параметров прикладных процедур	Да	Выполняются проверки параметров прикладных процедур обработки данных на соответствие системным требованиям, и формируется запись в журнале процесса.
2.2.	ENROLL_IMPOBJ	Регистрация импортируемых объектов	Да	Выполняется контроль атрибутов каждого загружаемого в рамках сеанса метаобъекта и производится регистрация метаобъекта в журнале импортируемых метаобъектов банков данных
2.3.	CHECK_IMPOBJ	Проверки достаточности структур и взаимосвязей импортируемых объектов	Да	Выполняются системные проверки корректности описания структур импортируемых объектов . Основным назначением этапа является выполнение определенных логических проверок, которые разрешают загрузку данных объектов текущего сеанса.
2.4.	LOADOBJ_TO_STAGE	Загрузка данных в stage-таблицы	Да	Выполняется загрузка данных внешних источников в структуры временного хранения. В журналах системы регистрируется итоговая статистика процессов загрузки в в stage-таблицы по каждому метаобъекту в рамках сеанса, а также итоговая статистика по этапу загрузки данных в Stage.
3.	**DWH_LOAD**	**Процесс обработки и загрузки в DWH**		Старт процесса обработки и загрузки данных в DWH из Staging Area. Состоит из шести подэтапов
3.1.	GLOBAL_CHECK	Общая предварительная обработка данных	Нет	Пользовательский этап, предназначенный для выполнения прикладных процедур предобработки данных в Stage- области, предшествующих системной проверке.
3.2.	GLOBAL_SYSCHK	Глобальная системная проверка	Да	Выполнение общих системных проверок.
3.3.	GLOBAL_BEFORE	Общая дозагрузочная обработка	Нет	Пользовательский этап, предназначенный для выполнения прикладных процедур обработки данных до загрузки в DWH.
3.4.	START_LOAD_DATA	Подготовка к загрузке данных в DWH	Да	Подготовка к загрузке данных из области Stage в область хранилища данных, регистрация начала процесса переноса данных в DWH в журнале процесса импорта данных. Состоит из четырех последовательно выполняемых этапов контроля данных (3.4.1, 3.4.2, 3.4.3 и 3.4.4)
3.4.1.	DATABANK _BEFORE	Дозагрузочная обработка отдельного банка данных	Нет	Пользовательский этап. Предназначен для выполнения прикладных процедур обработки отдельных банков данных. Этап выполняется для каждого банка данных, загружаемого в рамках сеанса.
3.4.2.	DATABANK_SYSCHK	Системные проверки для определенного банка данных	Да	Выполняется ряд обязательных системных процедур контроля качества данных, в том числе:

Очередность выполнения этап\подэтапов	Код этапа\подэтапа	Наименование этапа \ подэтапа	Системный этап (Да\Нет)	Описание этапа
				- Контроль заполнения обязательных полей; - Контроль дублирования кодов; - Контроль неоднозначности типовых объектов; - Контроль даты актуальности; - Контроль соответствия рангов иерархии.
3.4.3.	DATABANK_LOAD	Загрузка данных отдельного банка данных в DWH	Да	В ходе загрузки выполняется ряд обязательных системных процедур контроля данных: - системная проверка целостности объекта (проверка присутствия записи в центральной таблицы банка данных для групповых, типовых и иерархических таблиц того же банка данных); - системная проверка ссылки объекта к транзакционному банку данных; - системная проверка ссылки объекта к словарному банку данных; - системная проверка ссылки объекта к пользовательскому банку данных; - системная проверка иерархической структуры. Этап выполняется для каждого банка данных, загружаемого в рамках сеанса.
3.4.4.	DATABANK _AFTER	Постзагрузочная обработка отдельного банка данных	Нет	Постзагрузочная обработка отдельного банка данных
3.5.	GLOBAL_AFTER	Общая постзагрузочная обработка	Нет	Выполнение постзагрузочных обработок в DWH
3.6.	END_LOAD_DATA	Завершение загрузки данных в DWH	Да	Завершение загрузки данных в DWH, в том числе очистка stage-таблиц и формирование статистического отчета о загрузке данных в DWH

2.2.4 Процедуры этапов импорта

Краткое описание процедур, выполняемых на этапах импорта данных представлено в Табл. 2

Табл. 2 Процедуры этапов импорта данных

Код этапа	Процедура	Краткое описание процедуры	Примечание
SESSION	SESSION_REGISTER	Процедура получает: При использовании сервера файлового обмена - путь и имя файла данных, параметры сеанса. При использовании внешнего ETL - код сеанса, параметры сеанса. Процедура выполняет добавление записи о новом сеансе в журнал сеансов импорта (STG_LOG_SESSIONS) и генерирует ID сеанса.	Процедура выполняет обработку ряда программных исключений, включающих проверку задания параметров сеанса. В случае обнаружения ошибок старт сеанса импорта не выполняется. В случае, если сеанс с заданным кодом уже зарегистрирован в журнале сеансов и был прерван с ошибкой, выполняется перезапуск сеанса. В случае, если сеанс с заданным кодом уже зарегистрирован в журнале сеансов и был успешно завершен, старт сеанса не выполняется.
STG_LOAD	SESSION_REGISTER	Выполняется процедура REG_TO_PROCESS_LOG, добавляющая запись о начале этапа импорта в журнал процессов (STG_LOG_PROCESS) После завершения всех подэтапов выполняется процедура завершения этапа FIX_TO_PROCESS_LOG	
ENROLL_IMPOBJ	LOADOBJ_REGISTER	Выполняется добавление записи об импорте нового метаобъекта в журнал импортируемых метаобъектов (STG_LOG_LOADOBJS)	
LOADOBJ_TO_STAGE	LOADOBJ_STG_LOAD	Для каждого метаобъекта выполняется процедура LOADOBJ_STG_INIT, предназначенная для создания партиций stage-таблиц метаобъекта в рамках текущего сеанса.	В случае успешного создания таблиц, в журнал процессов (STG_LOG_PROCESS) добавляется запись об успешном завершении процедуры P_RESULT=-0. В случае возникновения ошибок (отсутствие имени банка данных или имени метаобъекта, несоответствие атрибутного состава таблиц и т.д..) выполнение процедуры прерывается по программному исключению с

			ошибкой или критической ошибкой. Выходному параметру процедуры присваивается значение P_RESULT=-1
DWH_LOAD	PROCESS_SESSION	Выполняет управление процессом переноса данных из Stage-таблиц в DWH. Последовательно вызывает процедуры обработки, контроля и загрузки данных в DWH Определяет общий результат сеанса, Обновляет запись в журнале сеансов.	
GLOBAL_CHECK	run_registered_procedures	Запускает пользовательские процедуры контроля качества данных, зарегистрированные на этапе GLOBAL_CHECK	
GLOBAL_SYSCHK	run_registered_procedures	Запускает системные процедуры контроля качества данных, выполняемые на этапе GLOBAL_SYSCHK	
GLOBAL_BEFORE	run_registered_procedures	Запускает пользовательские процедуры дозагрузочной обработки, зарегистрированные на этапе GLOBAL_BEFORE	
START_LOAD_DATA	LOAD_DATABANKS	Организует цикл загрузки данных в DWH по каждому банку данных	
DATABANK_BEFORE	run_registered_procedures	Запускает пользовательские процедуры предзагрузочного контроля качества данных отдельного банка данных, зарегистрированные на этапе DATABANK_BEFORE .	
DATABANK_SYSCHK	syschk_databank	Запускает системные процедуры контроля качества данных отдельного банка данных, зарегистрированные на этапе DATABANK_SYSCHK.	
DATABANK_LOAD	LOAD_DATABANK	Организует перенос записей метаобъектов отдельного банка данных в таблицы DWH	
DATABANK_AFTER	run_registered_procedures	Запускает пользовательские процедуры постзагрузочного контроля качества данных отдельного банка данных, зарегистрированные на этапе DATABANK_AFTER.	

GLOBAL_AFTER	run_registered_procedures	Запускает пользовательские процедуры общего постзагрузочного контроля качества данных, зарегистрированные на этапе GLOBAL_AFTER	
END_LOAD_DATA	CLEAN_STAGE	В случае успешного завершения сеансы выполняет очистку Stage-таблиц	

2.2.4.1Управление процессом загрузки

Данная реализация системы позволяет не только управлять порядком обработки данных в Stage Area, но и использовать произвольные пользовательские правила бизнес-контроля и дополнительной обработки данных.

В ходе настройки системы определяется:

1. Очередность загрузки объектов из Stage Area в Хранилище данных;
2. Последовательность выполнения процедур контроля качества данных для каждого загружаемого объекта,
3. Последовательность процедур предзагрузочной обработки данных для каждого загружаемого объекта.
4. Последовательность выполнения процедур постзагрузочной обработки для каждого загружаемого объекта.

Изменение порядка загрузки может быть особенно полезно на этапе внедрения системы, в процессе загрузки архивных данных, поскольку общая скорость загрузки лимитируется скоростью перемещения данных из Stage Area в хранилище, и возрастает пропорционально числу бизнес-проверок. Выполнив первоочередную загрузку только корректных с точки зрения бизнес-логики данных можно обеспечить возможность работы с данными хранилища до завершения процесса полной загрузки.

Для любого объекта может быть определена последовательность бизнес-проверок, предшествующих процедуре загрузки в Хранилище данных. Настройка этапа обработки и проверки данных выполняется путем подключения соответствующих хранимых процедур и определения очередности их выполнения.

Для любого загружаемого из Stage Area объекта может быть определена последовательность процедур обработки данных, выполняемых до и после их загрузки в Хранилище. Примерами процедур постзагрузочной обработки могут являться классификация лицевых счетов, проводимая по различным алгоритмам, процедуры расчета агрегатов по времени или по статьям плана, процедуры расчета показателей и др. Настройка этапов дозагрузочной и постзагрузочной обработки данных выполняется путем подключения соответствующих хранимых процедур и определения очередности их выполнения.

2.2.4.2Журнализация процессов загрузки

В системе ведутся следующие журналы:

1. **Журнал сеансов импорта данных** – в журнале регистрируется информация о завершенных сеансах импорта. По каждому сеансу сохраняется код сеанса, дата и время начала и окончания сеанса, даты начала и окончания периода, за который были выгружены данные из учетной системы, код учетной системы, имя архивного файла, количество файлов данных в архиве, признак завершения загрузки, количество ошибочных файлов и др. характеристики сеанса. По каждому завершенному сеансу импорта в журнале формируется одна запись. Запись в журнале формируется также и в случае не штатного (аварийного) завершения сеанса.
2. **Журнал импорта данных о метаобъектах** – в журнале регистрируется информация о результатах загрузки данных в разрезе категорий и метаобъектов по каждому сеансу импорта. Для каждого загруженного метаобъекта в протокол записывается: код банка днных, дата и время начала и окончания импорта метаобъекта, длительность загрузки метаобъекта в рамках сеанса, код возврата утилиты загрузки данных в Stage, признак завершения загрузки в Stage, количество записей, загруженных в Stage-таблицу, признак завершения загрузки в DWH, количество записей добавленных в таблицу DWH и др.
3. **Журнал процесса импорта** – в журнале регистрируется информация о результатах исполнения отдельных этапов процесса загрузки по каждому сеансу импорта в разрезе метаобъектов. Для каждого этапа сеанса загрузки в журнал записывается код категории, код метаобъекта, дата и время начала и завершения этапа, SQL-имя исполняемой процедуры, признак завершения, текст сообщения для этапа и др..

4. **Журнал исполнения процедур обработки и контроля –** в журнале собирается статистика исполнения системных и прикладных процедур обработки и контроля качества данных. Для каждой системной или пользовательской процедуры, выполненной в ходе сеанса загрузки в журнал записывается: код сеанса, код этапа обработки, код правила обработки, дата и время начала и окончания процедуры обработки данных, признак завершения процедуры, количество ошибок и предупреждений, протокол процедуры качества данных.
5. **Журнал протокола контроля качества данных –** предназначен для регистрации сообщений об ошибках или предупреждениях, формируемых процедурами обработки данных в процессе контроля качества данных.
6. **Журнал регистрации значений параметров -** предназначен для регистрации значений параметров процедур обработки и контроля качества данных, выполняемых на соответствующем этапе сеанса загрузки для загружаемого объекта или сеанса в целом.

2.2.5 Статусы завершения этапов

Для индикации результатов выполнения этапов импорта данных в подсистеме импорта используются **статусы завершения этапов**, являющиеся интегральными показателями выполнения процесса загрузки.

Отдельно определяются:

- Статус переноса данных в Staging Area,
- Статус переноса данных в DWH.

2.2.5.1 Статус завершения этапа переноса данных в Staging Area

Обработка данных на этапе переноса данных в Stage-таблицы выполняется только системными процедурами. Системные процедуры проверки выполняют обработку каждой записи, загружаемой в Stage-таблицы. В результате проверок выявляются записи, содержащие критические ошибки, ошибки и предупреждения. Для каждой записи проставляется признак разрешения загрузки. Загрузка записей с ошибками запрещена, загрузка данных с предупреждениями разрешена. При обнаружении критических ошибок на этапах до переноса в Stage-таблицы сеанс загрузки прерывается.

Ошибочные данные, а также данные с предупреждениями переносятся в таблицы отложенных записей N#*. Таблицы отложенных записей не очищаются после завершения сеанса импорта. Администратор процессов загрузки имеет возможность выполнить запросы к таблицам отложенных записей, с целью просмотра ошибочных записей и записей, имеющих предупреждения.

Данные с предупреждениями загружаются и в Stage-таблицы и копируются в таблицы отложенных записей.

Статус завершения этапа переноса данных в Stage-таблицы определяется на основании обработки записей журнала качества данных.

Статус завершения этапа \ подэтапа переноса данных из внешних источников данных в Staging Area может принимать следующие значения:

- «Успешно» - данные успешно загружены, все записи метаобъектов, загружаемых в рамках сеанса перенесены в Stage-таблицы,
- «Есть ошибки» - не удалось загрузить данные в штатном режиме, хотя бы одна запись по какому-либо метаобъекту не была перенесена в Stage-таблицы,
- «Завершено с предупреждениями» - все данные загружены, но есть предупреждение внешнего процесса. На этапе переноса данных в Stage-таблицы обнаружена хотя бы одна запись, с предупреждениями.
- «Не определен» - если процедура PROCESS_SESSION по каким-либо причинам не вернула статус загрузки.

2.2.5.2 Статус завершения этапа переноса данных из Staging Area в DWH

Статус завершения этапа \ подэтапа переноса данных из Stage-таблиц в DWH может принимать следующие значения:

- «Есть критические ошибки»,
- «Есть ошибки»,
- «Есть предупреждения»,
- «Есть информационные сообщения»,
- «Успешно выполнен»,
- «Не определен».

Статус завершения этапа \ подэтапа импорта зависит от статуса исполнения процедур обработки и контроля данных, выполняемых на этапе импорта, который в свою очередь определяется алгоритмом обработки программных исключений и логикой работы каждой конкретной процедуры.

Определение статуса исполнения процедуры выполняется следующим образом:

1. Для каждой процедуры задается три пороговых уровня критичности: Критический (R), Ошибка (E) и Предупреждение (W)*1*. Задание пороговых уровней критичности выполняется:
   - для прикладных процедур обработки и контроля – при подключении прикладных процедур на пользовательские этапы импорта данных в интерфейсе подсистемы импорта.
   - для системных процедур пороговые уровни критичности устанавливаются программно.
2. Каждая процедура обработки и контроля данных должна иметь выходной параметр p_severity (уровень критичности).
3. Возвращаемое процедурой значение параметра p_severity сравнивается с пороговыми значениями уровней критичности, на основании результатов сравнения определяется статус завершения процедуры:
   - Если p_severity>9 или p_severity<=0, то статус завершения процедуры «Не определен»,
   - Если R<=p_severity<=9, то процедура получает статус исполнения «Есть критические ошибки»,
   - Если E<=p_severity<R, то процедура получает статус исполнения «Есть ошибки»,
   - Если W<=p_severity<E, то процедура получает статус исполнения «Есть предупреждения»,
   - Если 0<p_severity<W, то процедура получает статус исполнения «Есть информационные сообщения».

Статус этапов импорта из Stage-таблиц в DWH определяется статусами завершения процедур обработки и контроля данных, выполняемых на этапе. Описание порядка присвоения статусов этапам импорта представлено в таблице:

Табл. 3. Статусы этапов импорта данных из Stage в DWH

[1] *Уровни критичности задаются целыми числами в интервале от 0 до 9, подробнее см. в разделе 3.4.*

№	Статус этапа загрузки	Порядок присвоения статуса этапу	Выполнение сеанса загрузки
1	Есть критические ошибки	Статус присваивается этапу, если хотя бы одна процедура обработки и контроля качества данных на этапе завершена с критической ошибкой. Если процедура выполняет позаписную обработку данных, статус присваивается, если обработка хотя бы одной записи завершена с критической ошибкой.	Сеанс загрузки **прерывается**. Данные, загруженные в Stage-таблицы, автоматически не удаляются.
2	Есть ошибки	Статус присваивается этапу, если хотя бы одна процедура обработки и контроля качества данных на этапе завершена с ошибкой, и нет критических ошибок. Если процедура выполняет позаписную обработку данных, статус присваивается, если обработка хотя бы одной записи завершена с ошибкой и нет критических ошибок на уровне записей.	Сеанс загрузки **не прерывается**, загрузка ошибочной записи не выполняется.
3	Есть предупреждения	Статус присваивается этапу, если хотя бы одна процедура обработки и контроля качества данных на этапе завершена с предупреждением, и нет критических ошибок и ошибок. Если процедура выполняет позаписную обработку данных, статус присваивается, если обработка хотя бы одной записи завершена с предупреждением, и нет критических ошибок и ошибок на уровне записей.	Сеанс загрузки **не прерывается, осуществляется загрузка в DWH всех записей,** в журнале (протоколе ошибок и предупреждений) регистрируются предупреждения
4	Есть информационные сообщения	Статус присваивается этапу, если хотя бы одна процедура обработки и контроля качества данных на этапе завершена с информационными сообщениями, и нет критических ошибок, ошибок и предупреждений.	Сеанс загрузки **не прерывается, осуществляется загрузка в DWH всех записей**
5	Не определен	Статус присваивается этапу, если хотя бы одна процедура обработки и контроля качества данных на этапе завершена с неопределенным статусом, и нет критических ошибок, ошибок, предупреждений и информационных сообщений.	Загрузка в DWH не прерывается
6	Успешно выполнен	Статус присваивается этапу, если все процедуры обработки и контроля качества данных были завершены успешно.	Все записи успешно загружаются из Stage в DWH.

В Приложении 4 приведены уровни критичности, установленные для системных процедур контроля и проверки качества данных.

В случае прерывания сеанса импорта, произошедшего из-за критической ошибки на этапе переноса данных в DWH, данные из Stage-таблиц автоматически не удаляются. Процедура, завершенная с критической ошибкой, возвращает код ошибки, который может быть использован ETL-инструментом для обновления ошибочных записей в Stage (без повторной загрузки данных всего сеанса импорта). Это позволяет исключить затратную по времени операцию очистки и повторного заполнения Stage-таблиц.

3 Требования к программному и аппаратному обеспечению

СУБД Postgres Pro. Версия ядра СУБД не ниже 15.5. Требования к оборудованию и программному обеспечению для установки в соответствии с рекомендациями для эксплуатации СУБД (https://postgrespro.ru/).

Примечание

В случае использовании [PgBouncer](https://www.pgbouncer.org/features.html) необходимо выбирать сессионный режим работы.

Рабочая станция - любая рабочая станция, работающая под управлением Alt Linux или Astra Linux.

Приложение 1. Правила формирования SQL-имен метаобъектов

Формирование SQL-имен всех видов метаобъектов производится по следующим правилам:

- [PREFIX]_[NAME] - для центральных таблиц;
- [PREFIX]_T_[NAME] - для типовых таблиц;
- [PREFIX]_G_[NAME] - для групповых таблиц;
- [PREFIX]_GC_[NAME] - для групповых таблиц, содержащих результаты классификации.

PREFIX	Описание
SO	Системные объекты (System Objects)
ENM	Перечислители (Enumerators)
DCT	Справочник (Dictionary)
GC	Общероссийские или общеупотребительные классификаторы (Global Classifier)
PA	Планы счетов и регламентированных показателей (Plan of Accounts)
PI	Планы показателей (Plan of Indicators)
PP	Стандарты учета и справочник планов
PN	Планы нормативов
SB	Контрагенты АБС (Subjects from OLTP-System)
SBE	Расширения для контрагентов (Subjects Extensions)
RS	Реестр контрагентов (Register of Subjects)
OS	Организационная структура (Organizational Structure)
OSE	Расширения для оргструктуры (Organizational Structure Extensions)
FA	Бухгалтерский учет
RF	Регламентные формы отчетности (Regulation Forms)
MA	Управленческий учет

Общая длина SQL-имени метаобъекта или атрибута метаобъекта не может превышать 20 символов.

Приложение 2. Диаграммы состояний базовых категорий ХД

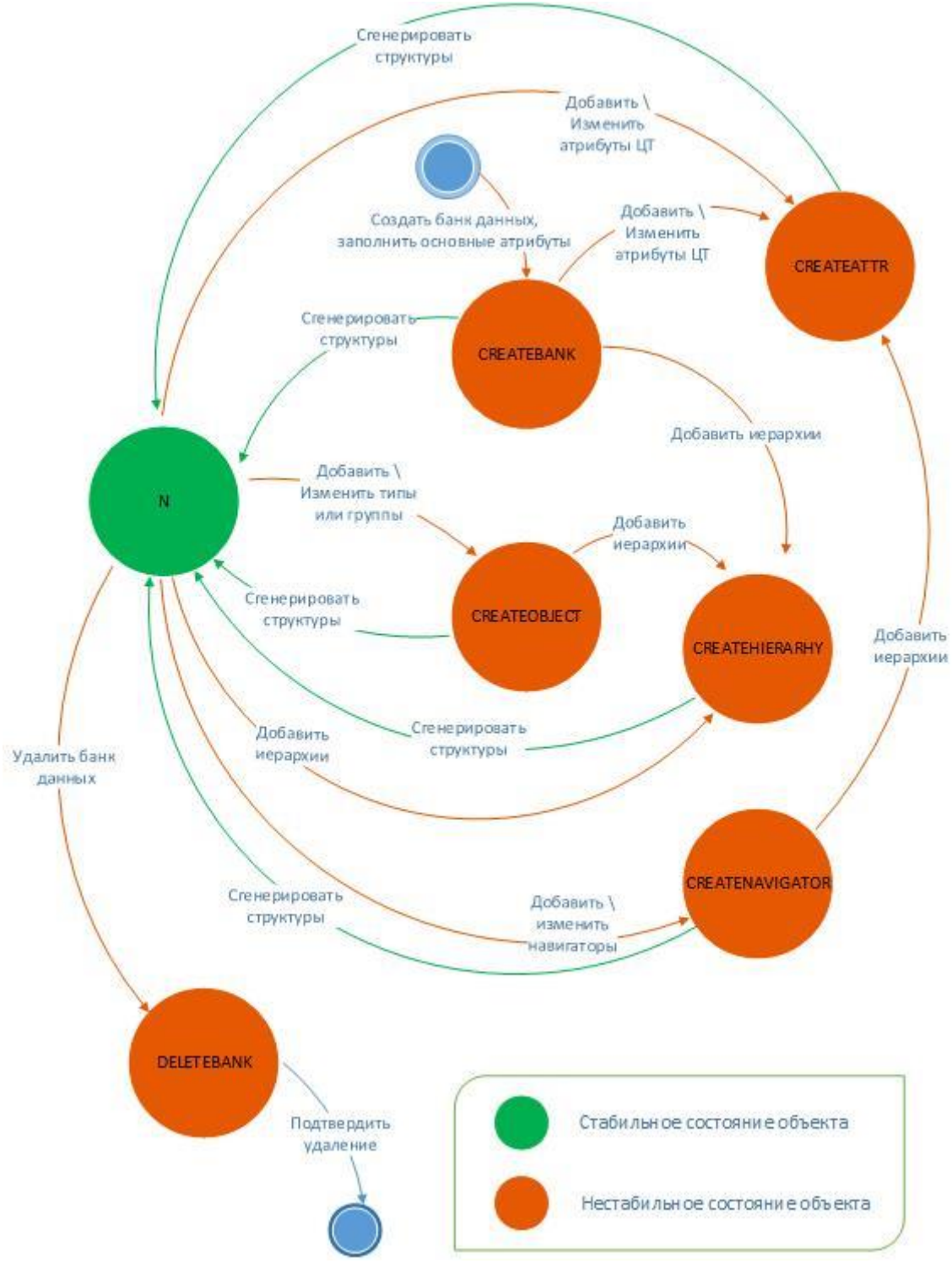

Рис. 3. Диаграмма состояний категории «Пользовательский банк данных»

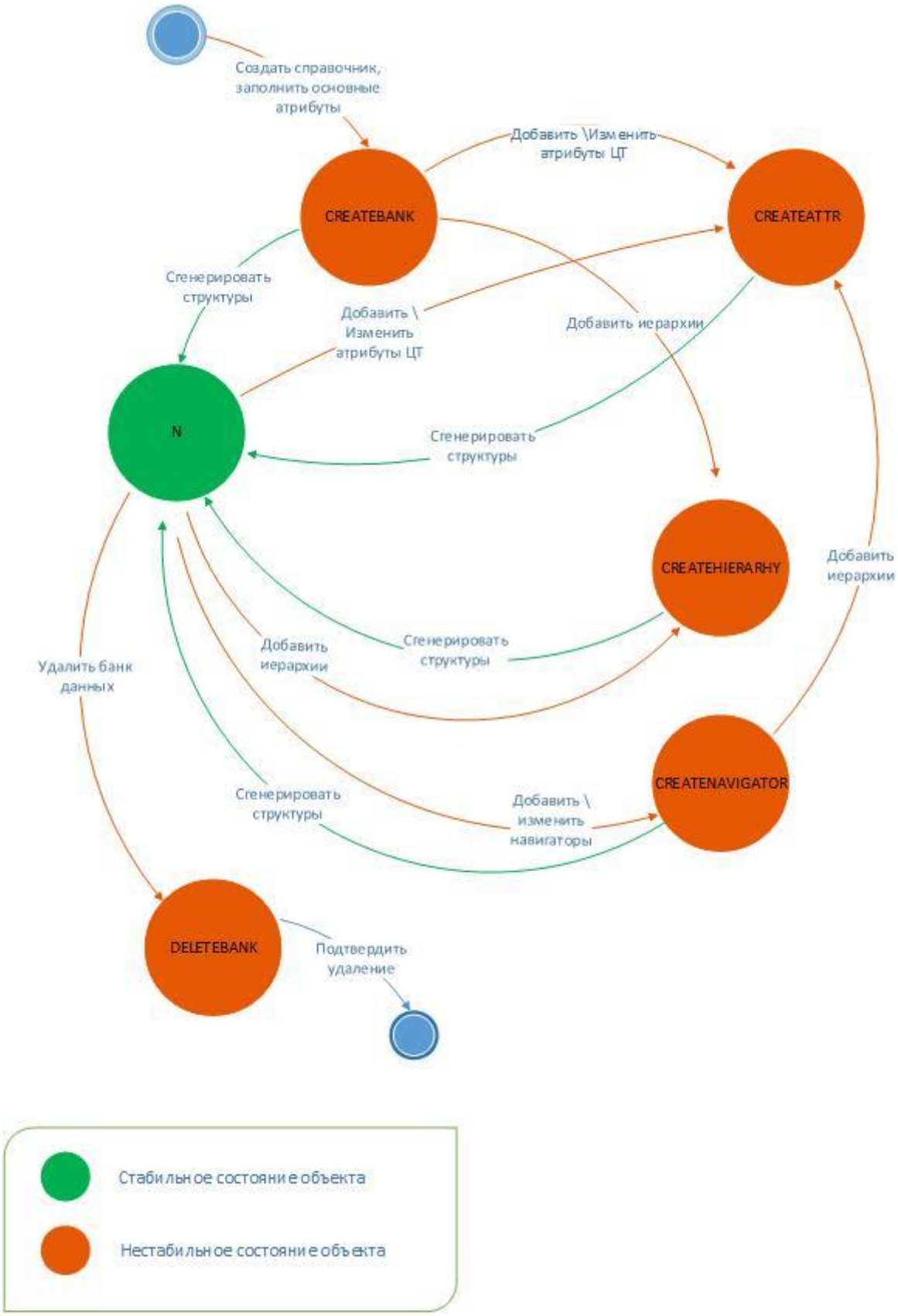

Рис. 4. Диаграмма состояний категории «Словарный банк данных»

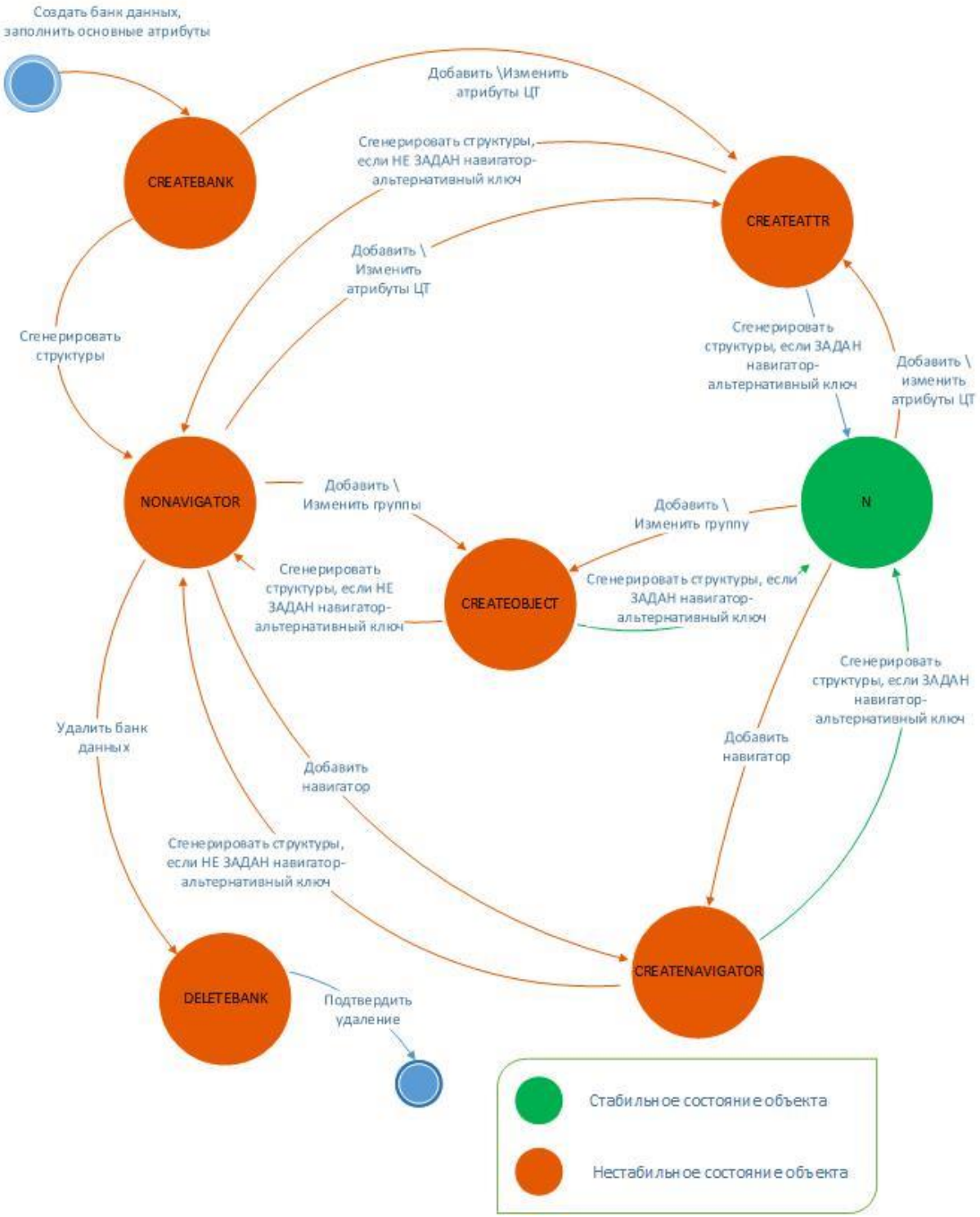

Рис. 5. Диаграмма состояний категорий «Интервальный банк данных» и «Транзакционный банк данных»

Приложение 3. Стандартный сценарий импорта с использованием произвольного ETL-инструмента

Описание стандартного сеанса загрузки с использованием произвольного ETL-инструмента представлено в таблице.

ETL- инструмент	**Подсистема импорта**	**Обязатель-ность выполнения**	**Примечание**
1. Запускает новый сеанс загрузки данных в ХД			
1.1. Вызывает процедуру SESSION_REGISTER, передает ей параметры сеанса.	1.2. Выполняет регистрацию сеанса в журнале сеансов импорта, 1.3. Проводит анализ параметров сеанса и обработку программных исключений, 1.4. Возвращает ID сеанса.	Да	Если хотя бы один из параметров сеанса не определен или задан некорректно, сеанс импорта прерывается. Пользователь получает сообщение о причинах прерывания сеанса.
2. Выполняет цикл регистрации метаобъектов.			
2.1. Для каждого метаобъекта вызывает процедуру LOADOBJ_REGISTER для заданного ID сеанса	2.2. Регистрирует сеанс импорта данных по метаобъекту в журнале импорта данных по метаобъектам. 2.3. Проводит контроль входных параметров процедуры, выполняет обработку программных исключений. 2.4. Проводит контроль наличия зарегистрированного сеанса в журнале сеансов импорта, выполняет обработку программных исключений. 2.5. Проводит контроль наличия банка данных в метаданных Системы и выполняет обработку программных исключений.	Да	Если один из параметров процедуры не определен или задан некорректно (метаобъект не зарегистрирован в системе, запрещена загрузка метаобъекта через Staging Area), процедура прерывает работу, идентификатору регистрации метаобъекта присваивается значение NULL. Сеанс загрузки при этом прерывается.

	2.6. Создает партиции stage-таблиц метаобъекта в рамках текущего сеанса 2.7. Возвращает значение идентификатора регистрации метаобъекта		
3. Выполняет проверку импортируемых метаобъектов			
3.1. Запускает процедуру LOADOBJ_CHECKS		Нет	
4. Выполняет цикл загрузки данных метаобъектов в Stage-таблицы. Для каждого метаобъекта последовательно выполняются шаги 4.1, 4.3 и 4.4.			
4.1. Вызывает процедуру LOADOBJ_STG_LOAD.	4.2. Регистрирует начало импорта данных в Staging Area для метаобъекта в журнале процесса импорта данных.	Да	Если не определен идентификатор регистрации метаобъекта, или метаобъект не зарегистрирован в журнале импорта по метаобъектам, процедура прерывает работу. Сеанс загрузки при этом не прерывается, продолжается загрузка данных остальных метаобъектов.
4.3. Заполняет stage-таблицы данными, полученными из внешних систем		Да	
4.4. Вызывает процедуру LOADOBJ_STG_FINALIZE	4.5. Выполняет контроль параметров процедуры и обработку программных исключений. 4.6. Подтверждает импорт данных по метаобъекту в журнале импорта метаобъектов (STG_LOG_LOADOBJS) 4.7. Подтверждает исполнение этапа загрузки в Stage для загружаемого метаобъекта в журнале процессов импорта данных (STG_LOG_PROCESSES).	Да	Формируется сообщение об успешной \ не успешной загрузке данных метаобъекта в Staging Area.
5. Осуществляет регистрацию параметров прикладных процедур обработки и контроля данных			

5.1. Запускает процедуру PARAMVALUE_REGISTER		Нет	
6. Завершает этап переноса данных в Staging Area.			
6.1. Вызывает процедуру SESSION_STG_FINALIZE.	6.2. Выполняет подтверждение загрузки данных в Staging Area для сеанса. 6.3. Обновляет запись в журнале сеансов импорта. 6.4. Выполняет обработку программных исключений.	Да	Определяется статус завершения этапа загрузки в Staging Area по сеансу («Успешно завершен», «Есть ошибки», «Не определен»).
7. Выполняет этап переноса данных из Staging Area в DWH.			
7.1. Вызывает процедуру PROCESS_SESSION, управляющую переносом данных из Staging Area в DWH	7.2. Выполняет контроль наличия запущенного сеанса переноса данных в DWH и обработку программных исключений. 7.3. Выполняет исполнение зарегистрированных в системе пользовательских процедур контроля данных по сеансу (вызывает процедуру RUN_REGISTERED_PROCEDURES с параметром P_PHASE='GLOBAL_CHECK'). 7.4. Выполняет исполнение системных процедур контроля данных по сеансу (вызывает процедуру RUN_REGISTERED_PROCEDURES с параметром P_PHASE='GLOBAL_SYSCHK') 7.5. Выполняет исполнение зарегистрированных в системе пользовательских предзагрузочных процедур контроля данных по сеансу (вызывает процедуру RUN_REGISTERED_PROCEDURES с параметром P_PHASE='GLOBAL_BEFORE') 7.6. Выполняет цикл переноса данных в DWH по банкам данных (для каждого банка вызывает процедуру LOAD_DATABANKS). Процедура LOAD_DATABANKS выполняет следующие действия:	Да	Если в текущих активных сессиях к СУБД обнаружена запись с текущим ИД сеанса, то сеанс импорта прерывается. В случае завершения какой-либо процедуры обработки и контроля данных с критическими ошибками, сеанс загрузки прерывается. При прерывании сеанса автоматическая очистка Stage-таблиц не выполняется. Внешний ETL-инструмент получает код ошибки.

	7.6.1. Выполняет зарегистрированные в системе пользовательские процедуры предзагрузочного контроля банка данных (вызывает процедуру RUN_REGISTERED_PROCEDURES с параметрами (P_PHASE = ‘DATABANK_BEFORE’, P_DATABANK = A_DATABANK[i])) 7.6.2. Выполняет системные процедуры контроля качества данных (вызывает процедуру SYSCHK_DATABANK с параметром P_DATABANK = A_DATABANK[i] 7.6.3. Выполняет перенос данных банка данных из Stage-таблиц и таблицы DWH (вызывает процедуру LOAD_DATABANK с параметром P_DATABANK = A_DATABANK[i]) 7.6.4. Выполняет зарегистрированные в системе пользовательские процедуры постзагрузочной проверки качества данных (вызывает процедуру RUN_REGISTERED_PROCEDURES с параметрами P_PHASE = ‘DATABANK_AFTER’, P_DATABANK = A_DATABANK[i]) 7.7. Выполняет зарегистрированные в системе пользовательские процедуры постзагрузочной проверки (вызывает процедуру RUN_REGISTERED_PROCEDURES с параметрами P_PHASE =’GLOBAL_AFTER’) 7.8. Выполняет очистку области Stage по сеансу (вызывает процедуру CLEAN_STAGE) 7.9. Возвращает статус исполнения переноса данных из Staging Area в DWH		
8. Завершает сеанс импорта данных			

Приложение 4. Пороговые значения уровней критичности системных процедур контроля и проверки качества данных

№	Этап импорта	SQL-имя процедуры	Назначение процедуры	Пороговые уровни критичности		
				Критический (R)	Ошибка (E)	Предупреждение (W)
1	DATABANK_SYSCHK'	CHECK_STAGE_EQUALS'	Системная проверка на недопустимые дубликатные значения	9	8	7
2	DATABANK_SYSCHK'	CHECK_STAGE_TYPES	Системная проверка на неоднозначность типов (только для типовых таблиц)	**9**	**8**	**7**
3	DATABANK_SYSCHK'	CHECK_STAGE_ACTUAL_DATE'	'Системная проверка атрибута actual_date на несоответствие 31.12.9999 (только для таблиц с историей)	**9**	**8**	**7**
4	DATABANK_SYSCHK'	CHECK_STAGE_RANGS	'Системная проверка рангов иерархий	**9**	**8**	**7**
5	DATABANK_LOAD	'CHECK_STAGE_INTEGRITY	Системная проверка целостности объекта (проверка присутствия записи в центральной таблицы банка данных для групповых, типовых и иерархических таблиц того же банка данных)	9	8	7
6	DATABANK_LOAD	ADD_OTHER_DATABANK_KEYS	Системная проверка ссылки объекта к транзакционному банку данных'	9	8	7

7	DATABANK_LOAD	ADD_OTHER_DATABANK_KEYS'	Системная проверка ссылки объекта к словарному банку данных	9	8	7
8	DATABANK_LOAD	ADD_OTHER_DATABANK_KEYS'	Системная проверка ссылки объекта к пользовательскому банку данных	9	2	1
9	DATABANK_LOAD	LOAD_HIERARHY_TABLE'	Системная проверка иерархической структуры'	9	8	7

intersoftlab
CORPORATE PERFORMANCE MANAGEMENT
E-mail: sales@iso.ru