Arenadata™ NSUD Data Mart

Версия - master

Руководство администратора

2020 Arenadata™

Оглавление

1 Общий обзор **3**

2 Требования к функциональным характеристикам **4**
- 2.1 Используемые библиотеки и фреймворки . . . 4
- 2.2 Требования к разделению классов по пакетам . . . 5

3 Логическая схема данных **6**
- 3.1 SQL+ . . . 6
- 3.2 Внешняя таблица . . . 6
- 3.3 Дельта . . . 7
- 3.4 Репликация . . . 7
- 3.5 Среда . . . 7
- 3.6 Логическая таблица . . . 7

4 Физическая схема данных **8**
- 4.1 Запись . . . 8
- 4.2 Обогащенный запрос . . . 8
- 4.3 Системные поля . . . 8
- 4.4 Физическая таблица . . . 9
- 4.5 Хранилище данных . . . 9

5 Запись данных **10**
- 5.1 Массивно-параллельная загрузка дельты . . . 10
- 5.2 Загрузка данных в ADB . . . 11
- 5.3 Загрузка данных в ADQM . . . 12
- 5.4 Загрузка данных в ADG . . . 13
- 5.5 Запись дельты (репликации) . . . 14

6 Чтение данных **15**
- 6.1 Массивно-параллельное извлечение данных . . . 15
- 6.2 Low-latency выборка данных . . . 15
- 6.3 Извлечение дельты (репликации) . . . 16
- 6.4 Извлечение снэпшота данных (репликация) . . . 17
- 6.5 Извлечение изменения результатов запросов (репликация) . . . 17
- 6.6 Выборка целостного набора данных в любой момент времени . . . 17
- 6.7 Выборка консистентного набора данных на определенный момент системного времени (for system_time) . . . 17
- 6.8 Тип данных . . . 17
- 6.9 JDBC-подключение к ADTM . . . 19

В Руководстве приведена информация по требованиям к функциональным характеристикам для работы **ADTM**, описана логическая и физическая схемы данных, а так же алгоритм записи и чтения данных.

Important: Контактная информация службы поддержки – e-mail: info@arenadata.io

Глава 1

Общий обзор

Дистрибутив **ADTM** – информационная система, сочетающая в себе свойства интеграционного решения и СУБД. Внешний программный интерфейс системы соответствует реляционной СУБД, подключение и взаимодействие с которой обеспечивает JDBC-драйвер. Управление функциями системы, в том числе чтением, массивно-параллельной загрузкой, массивно-параллельной выгрузкой, доступом к историческим данным, управление моделью данных производится с использованием расширенного диалекта SQL-команд.

С точки зрения архитектуры **ADTM** – интеграционное многокомпонентное кластерное решение, объединяющее хранилище данных, представленное кластерными гетерогенными СУБД, управляющее ядро системы и брокер сообщений, выполняющий роль внешнего и внутреннего сервиса интеграции.

Одной из важнейших функций управляющего ядра **ADTM** является поддержка темпоральности (версионности) данных, обеспечивающая эффективное хранение и доступ к целостным срезам данных на любой момент системного времени. Ядро системы также обеспечивает обработку внешних команд, обеспечивает координацию работы гетерогенных СУБД хранилища, контролируя процессы загрузки/выгрузки данных, гарантируя консистентное чтение и запись.

Глава 2

Требования к функциональным характеристикам

Для установки дистрибутива **ADTM** требуется:

- Язык программирования: C / Java / Lua.
- Тип реализующей ЭВМ: x86.
- Операционная система: Linux (CentOS 7.x).
- Объем дистрибутива программы для ЭВМ (в байтах): 658 907 байт.

2.1 Используемые библиотеки и фреймворки

ADTM использует библиотеки и фреймворки, представленные в таблице.

Таблица2.1.: Библиотеки и фреймворки

Наименование	Версия	Описание
Java	8.0.151	Язык программирования
Maven	3.6.3	Фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM
Spring Boot	2.2.4	RELEASE DI контейнер
Vert.x	3.8.5	Библиотека для разработки асинхронных приложений, основанная на событиях
liquibase	3.8.0	Библиотека с открытым исходным кодом для отслеживания, управления и применения изменений схемы базы данных
logback	1.2.3	Логгер
lombock	1.18.10	Плагин компилятора, который добавляет в Java новые «ключевые слова» и превращает аннотации в Java-код, уменьшая усилия на разработку и обеспечивая некоторую дополнительную функциональность
Jooq	3.13.1	Легковесная библиотека программного обеспечения для отображения базы данных в Java, которая реализует шаблон Active Record

2.2 Требования к разделению классов по пакетам

Arenadata DTM использует классы разработки по пакетам, представленным в таблице.

Таблица2.2.: Разделение классов по пакетам

Название пакета	Описание
outerlibrarypackage	Работа со сторонними библиотеками (например calcite)
configuration	Конфигурации приложения
controller	Работа с REST запросами
dao	Работа со внешними источниками данных
dto	Классы для передачи между сервисами
factory	Фабрики (генераторы сервисов)
model	Модели
service	Основные сервисы
utils	Вспомогательные сервисы
verticle	Классы для работы с вершинами Vert.x

Глава 3

Логическая схема данных

Important: Логическая схема данных являет собой внешнее представление о структуре загруженных в Систему объектов

3.1 SQL+

SQL+ – это SQL-запрос к логической схеме данных, предполагающий более широкий синтаксис, чем стандартный SQL. Запрос адресовывается драйверу **DM.JDBC**:

- **DDL** – это SQL+ выражение, модифицирующее логическую схему данных;
- **DELTA** – это SQL+ выражение управления дельтами изменений логической схемы данных;
- **DML** – это SQL+ выражение запроса данных из логической схемы данных;
- **EDDL** – это SQL+ выражение, управляющее структурой внешних таблиц логической схемы данных;
- **EDML** – это SQL+ выражение запроса данных логической схемы с участием внешних таблиц.

3.2 Внешняя таблица

Внешняя таблица представляет собой внешний интерфейс – временная регистрируемая функция перенаправления потока данных:

- На выгрузку – Внешняя таблица выгрузки, регистрируемая функция выгрузки данных из Системы. Определяет параметры выгрузки:
- подключение к приемнику данных;
- формат;
- ограничения (объемов/размера частей);
- На загрузку – Внешняя таблица загрузки, регистрируемая функция загрузки данных в Систему. Определяет параметры загрузки:
  - подключение к источнику данных;
  - формат;
  - схема данных (состав и тип полей).

3.3 Дельта

Дельта представляет собой логически целостную совокупность изменений информации об объектах. Каждой дельте поставляется в соответствие целое число из монотонно возрастающей последовательности целых чисел начиная с *0*, отражающее ее место в общей последовательности дельт и дата-время ее исполнения.

3.4 Репликация

Репликация представляет собой процесс отражения изменений информации об объектах Системы-источника (оригинал) в Системе-приемнике (реплика), Рис.3.1:

- Оригинал – Система-источник исходных данных в рамках процесса репликации;
- Реплика – Система-приемник, отражающая изменения объектов Системы-источника в рамках процесса репликации.

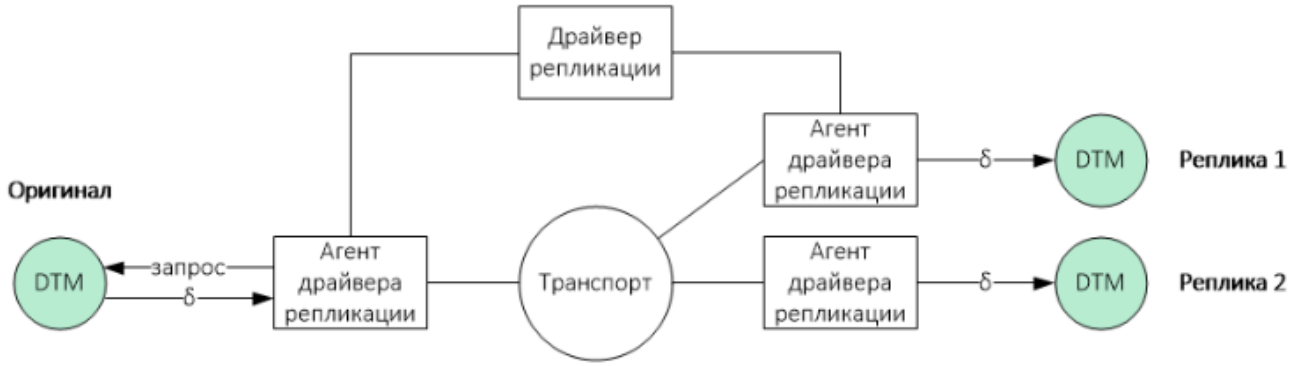


Рис.3.1.: Репликация

3.5 Среда

Среда представляет собой раздел логической схемы данных; совокупность datamart (информационных ресурсов), которые должны представлять собой совокупность исторических и актуальных данных об объектах и мета-данных объектов и состоять из набора логических таблиц.

3.6 Логическая таблица

Логическая таблица являет собой представление в виде таблицы БД в рамках datamart структурированной информации об объектах, на указанный момент системного времени. Где объект – это элемент данных (строчка таблицы БД) в ИС Поставщика; строка логической таблицы.

Important: Логическое представление являет собой сохраненный именованный запрос к логическим таблицам

Глава 4

Физическая схема данных

Important: Физическая схема данных представляет собой схему данных, используемую для организации размещения записей в хранилище данных

Для обеспечения возможности выборки актуального состояния объектов на дату и обеспечения высокой производительности в Системе используется подход к хранению данных, который предусматривает ведение истории изменения объектов с разделением на таблицу данных и таблицу истории.

4.1 Запись

Запись представляет собой строчку в физических таблицах, отражающую изменение объекта. Любые изменения объекта приводят к появлению новой записи:

- Актуальная [на определенный момент] запись – запись, для которой на определенный момент системного времени не существует новой версии. Отражает состояние объекта в определенный момент системного времени;
- Горячая запись – запись, относящаяся к загружаемой дельте. Отражает новое состояние объекта;
- Старая запись – запись, для которой существует новая версия. Отражает состояние объекта на дату.

4.2 Обогащенный запрос

Обогащенный запрос представляет собой SQL-выражение, адаптированное к целевой СУБД хранилища данных, позволяющее получить актуальные записи на произвольный момент системного времени (в том числе на текущий момент времени).

4.3 Системные поля

Системные поля бывают трех типов:

- `sys_from` (int) – целочисленное системное поле, хранящее значение номера дельты, в рамках которой запись появилась в БД;
- `sys_op` (int) – системное поле записи, отражающее операцию, которая была выполнена над объектом: *0* – INSERT (создание или модификация объекта); *1* – DELETE (удаление объекта);

- `sys_to` (int) – целочисленное системное поле, хранящее для старой записи максимальное значение номера дельты, в рамках которого запись оставалась актуальной. Для актуальных записей значение `sys_to` должно быть пусто (бесконечность).

4.4 Физическая таблица

Физическая таблица представляет собой таблицу, расположенную в хранилище данных, и может быть трех типов:

- Таблица staging – физическая таблица горячих записей;
- Таблица данных – физическая таблица актуальных записей. В момент загрузки дельты может дополнительно содержать старые записи и горячие записи. Включает в состав PK системное поле *SysFrom*;
- Таблица истории – физическая таблица старых записей. Включает в состав PK системное поле SysFrom. Включает поля: *SysTo* – последний номер дельты, в рамках которой запись была актуальна, *SysOp* – признак удаленного объекта.

4.5 Хранилище данных

Хранилище данных представляет собой совокупность кластеров СУБД, используемых для хранения информации об объектах:

- Кластер ADB;
- Кластер ADQM;
- Кластер ADG.

Глава 5

Запись данных

Запись данных в **ADTM** обеспечивает консистентное обновление (вставка, обновление, удаление) объектов в Системе.

К записи данных предъявляются следующие ограничения и допущения:

- Дельты загружаются строго последовательно;
- Данные из сообщений одной дельты загружаются параллельно;
- Дельты со стороны ETL ИС Поставщика выгружаются строго последовательно.

Запись данных имеет следующий процесс:

- Горячие записи загружаются в таблицу данных;
- Старые записи из таблицы данных перемещаются в таблицу истории.

5.1 Массивно-параллельная загрузка дельты

Массивно-параллельная загрузка дельты в **ADTM** обеспечивает массивно-параллельное консистентное обновление (вставка, обновление, удаление) объектов в Системе.

Массивно-параллельная загрузка дельты выполняет следующие назначения:

- Создание внешних UPLOAD таблиц загрузки;
- Определение начала и окончания загрузки дельты;
- Загрузка данных дельты через внешние UPLOAD таблицы загрузки в физические таблицы хранилища данных, включая перенос старых записей из таблицы данных в таблицу истории.

К массивно-параллельной загрузке дельты предъявляются следующие ограничения и допущения:

- Не предполагается нумерация входящих сообщений Kafka;
- Входящие сообщения Kafka для каждого удаленного объекта содержат признак `sys_op(int)=1`;
- Гарантируется, что данные из прочитанных (коммит чтения) сообщений Kafka загружены в Систему;
- Окончание потока сообщений Kafka определяется по заданному в конфигурации таймауту ожидания сообщений kafka (максимально допустимый интервал времени ожидания сообщений kafka в рамках потока).

Входные данные:

- JDBC подключение к Системе;

- Топик Kafka, содержащий загружаемые данные.

Выходные данные *SUCCESS*:

- Загружены данные в физические таблицы хранилища данных, включая перенос старых записей из таблицы данных в таблицу истории;
- Выполнен коммит чтения для входящих сообщений Kafka;
- Увеличен номер последней загруженной дельты в Системе.

Выходные данные *FAIL*:

- Частично загружены данные в физические таблицы хранилища данных, включая перенос старых записей из таблицы данных в таблицу истории;
- Выполнен коммит чтения для части входящих сообщения Kafka, которые были загружены в СУБД хранилища данных;
- Не изменен номер последней загруженной дельты в Системе;
- Сформировано сообщение об ошибке в ответ на входящий JDBC запрос;
- Зафиксирована ошибка загрузки дельты.

5.2 Загрузка данных в ADB

Загрузка данных в **ADB** в ПО **ADTM** обеспечивает массивно-параллельную загрузку данных из **Kafka** в физические таблицы **ADB**.

Загрузка данных в **ADB** выполняет следующие назначения:

- Загрузка горячих записей из топика Kafka в таблицу данных;
- Перенос старых записей из таблицы данных в таблицу истории в рамках транзакции.

Входные данные:

- Номер последней успешно загруженной дельты `DELTA_OK`;
- Номер загружаемой дельты `DELTA_HOT`;
- Kafka topicY, содержащий данные для загрузки;
- JDBC подключение к ADB;
- readable external table (tblExt), читающая данные из Kafka topicY;
- Таблица staging (*tbl_staging*);
- Таблица данных (*tbl_actual*);
- Таблица истории (*tbl_history*).

Выходные данные:

- Таблица данных содержит только актуальные данные (для момента времени `DELTA_HOT`);
- Таблица истории содержит старые записи (для момента времени `DELTA_HOT`);
- Таблица staging пуста.

К загрузке данных в **ADB** предъявляются следующие ограничения и допущения:

- В таблице staging могут оказаться дублированные записи;
- Данные в Kafka topicY содержат признак `SysOp=1` для каждого удаленного объекта;

- Перенос данных (удаление/вставка) выполняется в рамках каждого сегмента (не происходит перемещения данных между сегментами);
- Старая запись может иметь только два состояния:
- полностью перенесена (отсутствует в таблице данных, присутствует в таблице истории с требуемыми значениями системных полей);
- не перенесена (присутствует в таблице данных, отсутствует в таблице истории).

5.3 Загрузка данных в ADQM

Загрузка данных в **ADQM** в ПО **ADTM** обеспечивает массивно-параллельную загрузку данных из **Kafka** в физические таблицы **ADQM**.

Загрузка данных в **ADQM** выполняет следующие назначения:

- Загрузка горячих записей из заданного топика Kafka в таблицу данных;
- Обновление состояния старых записей в таблице данных;
- Обеспечение консистентной выборки в любой момент времени;
- Перенос старых записей в отдельное физическое хранилище (диск, том).

К загрузке данных в **ADQM** предъявляются следующие ограничения и допущения:

- Отсутствует таблица истории. Перенос старых записей в специальное хранилище осуществляется функционалом СУБД TTL;
- Момент определения записей подлежащих переносу можно настраивать;
- В момент загрузки дельты выборка может выполняться дольше (используется ключевое слово *FINAL*);
- Запросы, начатые до момента начала загрузки дельты, не затрагивают записи, созданные в процессе загрузки дельты, в том числе записи сторно;
- Повторные записи (полное совпадение полей) дедуплицируются при завершении загрузки дельты;
- В таблице данных отсутствует признак удаления `sys_op`;
- Предусмотрена работа в кластере.

Входные данные:

- Номер актуальной дельты – `@delta_ok`;
- Номер загружаемой дельты – `@delta_hot`;
- Имя топика – `topicY`;
- Константа `@ttl_wait_sec` – количество секунд после фактического устаревания записи (вставка записи) до готовности к переносу старых записей на другое хранилище;
- Формат данных (AVRO);
- ADQM:
  - таблица данных с указанием выражения TTL – tbl_actual (CollapsingMergeTree):
  - `[ columns ]`;
  - `sys_from` (Int64);
  - `sys_to` (Int64);

- `close_date` (DateTime64) – дата, с момента которой отсчитывается готовность к переносу на другое хранилище;
- `sign` (Int8) – признак отмены записи (сторно).

5.4 Загрузка данных в ADG

К загрузке данных в **ADG** предъявляются следующие ограничения и допущения:

- Движок таблиц memtx;
- Требуется точное совпадение состава, типа и порядка полей во входящих сообщениях и структуре таблиц staging;
- Допустим перенос данных в историческую таблицу и таблицу данных в процессе пополнения таблицы staging;
- Перенос данных из таблицы staging в таблицу данных и перенос старых записей в таблицу истории должен происходить в рамках одной транзакции.

Входные данные:

- `@delta_hot` – номер загружаемой дельты;
- Входящие сообщения в топике Kafka topicY;
- Кластер ADG:
  - таблица staging `tbl1_staging`:
  - (поля логической таблицы);
  - `sys_op` (int);
  - `bucket_id` (unsigned);
  - таблица данных `tbl1_actual`:
  - (поля логической таблицы);
  - `sys_from` (int) – последнее поле PK;
  - `sys_to` (int);
  - `sys_op` (int);
  - `bucket_id` (unsigned);
  - таблица истории `tbl1_history`:
  - (поля логической таблицы);
  - `sys_from` (int) – последнее поле PK;
  - `sys_to` (int);
  - `sys_op` (int);
  - `bucket_id` (unsigned).

Выходные данные:

- Таблица данных содержит только актуальные данные (для момента времени `@delta_hot`);
- Таблица истории содержит старые записи (для момента времени `@delta_hot`);
- Таблица staging пуста.

5.5 Запись дельты (репликации)

Запись данных (репликация) в **ADTM** обеспечивает репликацию в части массивно-параллельной загрузки данных в реплику.

Запись данных (репликация) выполняет следующие назначения:

- Принимает от Агента ПОДД (Подсистема обеспечения доступа к данным) входящий поток сообщений `delta.in`, содержащий дельту и системные данные дельты (δ – номер дельты, δ`-time` – дата-время дельты);
- Определяет состав и целостность полученной дельты;
- Выполняет массивно-параллельную загрузку данных;
- Устанавливает номер и дату-время загруженной дельты, равные номеру и дате-времени дельты оригинала;
- Формирует ответ `delta.in.rs` для Агента ПОДД.

К записи данных (репликации) предъявляются следующие ограничения и допущения:

- Загрузка дельт происходит строго последовательно – новая дельта после подтверждения загрузки предыдущей (`delta.in.rs`);
- Загрузка дельты происходит в потоковом режиме – загрузка дельты в реплику начинается после получения первого сообщения `delta.in` и заканчивается в момент, когда загружены данные из всех сообщений дельты;
- `delta.in` может содержать дубликаты сообщений;
- Дата-время одинакового номера дельты оригинала и реплики совпадают;
- Логическая схема данных реплики должна соответствовать загружаемым данным (мета-данные загружаемых данных определяются по логической схеме данных реплики);
- ПОДД поддерживает протокол обмена в формате `delta.rq`, `delta.rs`, `delta.in`;
- ПОДД является драйвером репликации.

Входные данные:

- DM.JDBC подключение;
- Поток сообщений `delta.in`, каждое из которых содержит:
- δ – номер дельты;
- δ`-time` – дата-время дельты;
- sql-запрос подписки;
- порядковый номер таблицы (номер потока);
- общее количество таблиц (количество потоков);
- порядковый номер сообщения в рамках таблицы (в рамках потока);
- признак последнего сообщения в рамках таблицы (в рамках потока);
- [подмножество данных таблицы].

Выходные данные:

- Загруженная в реплику дельта (δ, δ`-time`);
- Исходящее сообщение `delta.in.rs`.

Глава 6

Чтение данных

6.1 Массивно-параллельное извлечение данных

Массивно-параллельное извлечение данных в **ADTM** обеспечивает высокопроизводительную выгрузку данных из Системы.

Массивно-параллельное извлечение данных выполняет следующие назначения:

- Создание внешних таблиц выгрузки по SQL-запросу JDBC;
- Заполнение внешних таблиц выгрузки данными по SQL-запросу JDBC;
- Удаление внешних таблиц выгрузки по SQL-запросу JDBC.

Входные данные:

- Внешнее JDBC подключение к ADTM;
- kafka topicX, в который будут выгружены данные.

Выходные данные:

- Запрошенные данные, разделенные на потоки и части в виде сообщений указанного топика Kafka.

К массивно-параллельному извлечению данных предъявляются следующие ограничения и допущения:

- Нет автоматического создания/удаления топиков Kafka;
- Нет автоматического удаления внешних таблиц выгрузки;
- Внешние таблицы выгрузки не являются Writable External Table СУБД хранилища данных;
- Количество строк в одном сообщении регулируется базовыми настройками Системы;
- Запрос на выборку данных может выполняться произвольное количество раз после создания внешней таблицы выгрузки;
- Определение целевой СУБД выгрузки данных производится на основании конфигурации Системы.

6.2 Low-latency выборка данных

Low-latency выборка данных в **ADTM** обеспечивает выбор объекта Системы на заданный момент системного времени с минимальными задержками.

Low-latency выборка данных выполняет следующие назначения:

- Преобразование (обогащение) SQL+ DML выражения к запросу СУБД хранилища данных;

- Исполнение обогащенного запроса к СУБД хранилища данных;
- Возврат выбранных данных и мета-данных логической схемы данных (соответствующих SQL+ DML выражению).

Входные данные:

- SQL+ DML выражение обращение к DM.JDBC со стороны внешнего сервиса.

Выходные данные:

- Получение данных и мета-данных в ответ на входящий запрос к DM.JDBC.

6.3 Извлечение дельты (репликации)

Извлечение дельты (репликация) представляет собой извлечение подмножества измененных данных (колонки / фильтр по записям) в рамках одной дельты (диапазона дельт).

Извлечение дельты (репликация) данных в **ADTM** обеспечивает репликацию в части массивно-параллельного извлечения данных из оригинала.

Извлечение дельты (репликация) выполняет следующие назначения:

- Прием от Агента ПОДД запроса `delta.rq` на выгрузку дельты;
- Определение схемы реплики (подписка);
- Выполнение массивно-параллельного извлечения данных дельты в адрес реплики;
- Формирование ответа для Агента ПОДД `delta.rs`, содержащего дельту и системные данные дельты (δ – номер дельты, δ`-time` – дата-время дельты).

К извлечению дельты (репликации) предъявляются следующие ограничения и допущения:

- Логическая схема данных реплики соответствует структуре SQL-запросов подписки;
- Адаптер ПОДД хранит информацию в схемах реплик (подписка);
- Агент ПОДД поддерживает протокол обмена в формате `delta.rq`, `delta.rs`;
- Агент ПОДД является драйвером репликации;
- `delta.rs` может содержать дубликаты сообщений – повторный запуск после сбоя в процессе выгрузки приводит к формированию дублей сообщений в ответном потоке `delta.rs`.

Входные данные:

- Загруженная в оригинал дельта δ;
- Информация о подписках:
- каждая подписка – набор sql-запросов;
- DM.JDBC подключение;
- Входящее сообщение `delta.rq`:
- δ – номер дельты;
- идентификатор подписки (схемы реплики).

Выходные данные:

- Исходящий поток сообщений `delta.rs`, каждое из которых содержит:
- δ – номер дельты;
- δ`-time` – дата-время дельты;

- sql-запрос подписки;
- порядковый номер таблицы (номер потока);
- общее количество таблиц (количество потоков);
- порядковый номер сообщения в рамках таблицы (в рамках потока);
- признак последнего сообщения в рамках таблицы (в рамках потока);
- [подмножество данных таблицы].

6.4 Извлечение снэпшота данных (репликация)

Извлечение снэпшота данных (репликация) представляет собой выборку актуального среза данных (не дельт).

6.5 Извлечение изменения результатов запросов (репликация)

Извлечение изменения результатов запросов (репликация) представляет собой динамический расчет дельты изменений результатов запроса.

6.6 Выборка целостного набора данных в любой момент времени

Выборка целостного набора данных в любой момент времени обеспечивает обращение к целостному набору данных, вне зависимости от выполняемых процессов по загрузке обновлений.

6.7 Выборка консистентного набора данных на определенный момент системного времени (for system_time)

Выборка консистентного набора данных на определенный момент системного времени (for system_time) обеспечивает консистентную на заданный момент времени выборку данных из физической схемы данных на основании входящего SQL-запроса к логической схеме данных.

Выборка консистентного набора данных на определенный момент системного времени (for system_time) выполняет следующие назначения:

- Определение номера дельты `@SelectOn` соответствующего заданному моменту времени;
- Определение целевой СУБД;
- Используя входящий SQL-запрос к логической схеме данных, формирование запроса к физической схеме данных для определенного номера дельты `@SelectOn`;
- Выполнение общего сформированного запроса к целевой СУБД;
- Возврат результата выполнения запроса.

К выборке консистентного набора данных на определенный момент системного времени (for system_time) предъявляются следующие ограничения и допущения:

- Если момент времени не задан – используется номер последней успешно загруженной дельты.

6.8 Тип данных

К типам данных предъявляются следующие ограничения и допущения:

- Типы времени не содержат информации о time-zone;
- Поддерживается подмножество типов стандарта SQL;
- Поддерживаются типы вне стандарта SQL.

Логический тип дынных и должный соответствовать ему тип данных в кластере приведен в таблице.

Таблица6.1.: Типы данных

Логический тип	Описание	Тип данных ADB	Тип данных ADG	Тип данных ADQM
BOOLEAN	Логический тип	boolean	boolean	UInt8
VARCHAR (n)	Строка ограниченной длины (n символов)	varchar (n)	string	String
CHAR (n)	Строка ограниченной длины (n символов)	varchar (n)	string	String
UUID	Строка ограниченной длины (36 символов)	varchar (36)	string	String
BIGINT	Целые числа фиксированной длины со знаком в диапазоне от -9223372036854775808 до 9223372036854775807	int8	integer	Int64
INT	Целые числа фиксированной длины со знаком в диапазоне от -9223372036854775808 до 9223372036854775807	int8	integer	Int64
DOUBLE	Число с плавающей запятой с двойной точностью	float8	double	Float64
FLOAT	Число с плавающей запятой	real	double	Float32
DATE	Дата (без времени суток)	date	integer (знаковое число дней относительно 1970-01-01)	Int64 (знаковое число дней относительно 1970-01-01)
TIME (precision)	Время (без даты) с заданной точностью от 0 (секунды) до 6 (микросекунды)	time (precision)	integer (знаковое число микросекунд начиная с 00:00:00.000000)	Int64 (знаковое число микросекунд начиная с 00:00:00.000000)
TIMESTAMP (precision)	Дата и время с заданной точностью	timestamp (precision)	integer (знаковое число микросекунд начиная 1970-01-01 00:00:00)	DateTime64 (prescision) | Int64 (знаковое число микросекунд относительно 1970-01-01)

Загрузка данных *MPP-W* осуществляется в формате *AVRO*. Выгрузка данных *MPP-W* осуществляется в формате *AVRO* в соответствии со спецификой типов данных СУБД.

Таблица6.2.: Загрузка и выгрузка в формате AVRO

AVRO (загрузка)	Тип данных ADTM	ADB AVRO (выгрузка)	ADG AVRO (выгрузка)	ADQM AVRO (выгрузка)
boolean	BOOLEAN	boolean	boolean	int
string	VARCHAR (n)	string	string	string
long	BIGINT	long	long	long
double	DOUBLE	double	double	double
float	FLOAT	float	double	float
(int) date	DATE	(int) date	int	int
(long) time-micros	TIME (precision)	(long) time-micros	long	long
(long) timestamp-micros	TIMESTAMP (precision)	(long) timestamp-micros	long	(long) timestamp-micros

Выборка данных *LL-R* через *DM.JDBC* возвращает данные в виде типов *java.sql.Types* соответствующих типам логической схемы данных.

Плагины конвертируют полученные данные в *java.sql.Types* соответствующие логической схеме данных.

Выборка данных *LL-R* (через любые интерфейсы) позволяет выполнять запросы с указанием формата данных (секция *FORMAT* в *SELECT*).

Плагины конвертируют полученные данные в заданный *FORMAT* и/или запрашивают данные из СУБД в требуемом формате.

6.9 JDBC-подключение к ADTM

JDBC-подключение к **ADTM** обеспечивает JDBC-стандарт подключения к СУБД для **ADTM**.

JDBC-подключение к **ADTM** выполняет следующие назначения:

- Реализация функций стандарта JDBC;
- Подключение к REST на основании строки подключения;
- Предоставление результатов исполнения SQL-запросов потребителю.

Входные данные:

- Строка подключения к REST сервису обеспечивающему исполнение SQL-запроса;
- SQL-запрос.

Выходные данные:

- Результат выполнения SQL-запроса.