Базы Данных

Семинар 11

Описание семинара: ETL и ELT

Kimball vs. Inmon

Два пионера хранилищ данных: Билл Инмон и Ральф Кимбалл предлагают разные подходы к проектированию.

Подход **Ральфа Кимбалла** основывается на важности витрин данных, которые являются хранилищами данных, принадлежащих конкретным направлениям бизнеса. Хранилище данных — это просто **сочетание различных витрин данных**, которые облегчают отчетность и анализ. Проект хранилища данных по принципу Кимбалла использует подход «снизу вверх».

Подход **Билла Инмона** основывается на том, что хранилище данных является централизованным хранилищем всех корпоративных данных. При таком подходе организация сначала создает **нормализованную модель** хранилища данных. Затем создаются витрины размерных данных на основе модели хранилища. Это известно как нисходящий подход к хранилищу данных.

ETL – аббревиатура от Extract, Transform, Load.

Это системы корпоративного класса, которые применяются, чтобы привести к одним справочникам и загрузить в DWH и EPM данные из нескольких разных учетных систем. ETL – промежуточным слоем между OLTP системами и OLAP системой или корпоративным хранилищем. Хотя в принципе существуют ETL, который можно поставить между любыми системами, лучше интеграцию между учетными системами решать связкой MDM и ESB. Если же вам для интеграции двух зависимых учетных систем необходим функционал ETL, то это ошибка проектирования, которую надо исправлять доработкой этих систем.

Зачем нужна ETL система

Проблема, из-за которой в принципе родилась необходимость использовать решения ETL, заключается в потребностях бизнеса в получении достоверной отчетности из того бардака, который творится в данных любой ERP-системы. Этот бардак есть всегда, он бывает двух видов:

- Как случайные ошибки, возникшие на уровне ввода, переноса данных, или из-за багов;

- Как различия в справочниках и детализации данных между смежными ИТ-системами.

При этом если первый вид бардака побороть можно, то второй вид по большей части не является ошибкой – контролируемые различия в структуре данных, это нормальная оптимизация под цели конкретной системы.
Из-за этой особенности ETL-системы должны в идеале решать не одну, а две задачи:

- Привести все данные к единой системе значений и детализации, попутно обеспечив их качество и надежность;

- Обеспечить аудиторский след при преобразовании (Transform) данных, чтобы после преобразования можно было понять, из каких именно исходных данных и сумм собралась каждая строчка преобразованных данных.

Помнить об этих двух задачах бывает очень полезно, особенно если вы пишете ETL- процесс вручную, или делаете его с использованием фреймворков низкой готовности, в которых не задана готовая структура промежуточных таблиц. Легко упустить вторую задачу и иметь много проблем с поиском причин ошибок в трансформированных данных.

Как работает ETL система

Все основные функции ETL системы умещаются в следующий процесс:

В разрезе потока данных это несколько систем-источников (обычно OLTP) и система приемник (обычно OLAP), а так же пять стадий преобразования между ними:

Процесс загрузки – Его задача затянуть в ETL данные произвольного качества для дальнейшей обработки, на этом этапе важно сверить суммы пришедших строк, если в исходной системе больше строк, чем в RawData то значит — загрузка прошла с ошибкой;

Процесс валидации данных – на этом этапе данные последовательно проверяются на корректность и полноту, составляется отчет об ошибках для исправления;

Процесс мэппинга данных с целевой моделью – на этом этапе к валидированной таблице пристраивается еще n-столбцов по количеству справочников целевой модели данных, а потом по таблицам мэппингов в каждой пристроенной ячейке, в каждой строке проставляются значения целевых справочников. Значения могут проставляться как 1:1, так и *:1, так и 1:* и *:*, для настройки последних двух вариантов используют формулы и скрипты мэппинга, реализованные в ETL-инструменте;

Процесс агрегации данных – этот процесс нужен из-за разности детализации данных в OLTP и OLAP системах. OLAP-системы — это, по сути, полностью денормализованная таблица фактов и окружающие ее таблицы справочников (звездочка/снежинка), максимальная детализация сумм OLAP – это количество перестановок всех элементов всех справочников. А OLTP система может содержать несколько сумм для одного и того же набора элементов справочников. Можно было-бы убивать OLTP-детализацию еще на входе в ETL, но тогда мы потеряли бы «аудиторский след». Этот след нужен для построения Drill-down отчета, который показывает — из каких строк OLTP, сформировалась сумма в ячейке OLAP- системы. Поэтому сначала делается мэппинг на детализации OLTP, а потом в отдельной таблице данные «схлопывают» для загрузки в OLAP;

Выгрузка в целевую систему — это технический процесс использования коннектора и передачи данных в целевую систему.

Особенности архитектуры

Реализация процессов 4 и 5 с точки зрения архитектуры тривиальна, все сложности имеют технический характер, а вот реализация процессов 1, 2 и 3 требует дополнительного пояснения.

Процесс загрузки

При проектировании процесса загрузки данных необходимо помнить о том что:

- Надо учитывать требования бизнеса по длительности всего процесса. Например: Если данные должны быть загружены в течение недели с момента готовности в исходных системах, и происходит 40 итераций загрузки до получения нормального качества, то длительность загрузки пакета не может быть больше 1-го часа. (При этом если в среднем происходит не более 40 загрузок, то процесс загрузки не может быть больше 30 минут, потому что в половине случаев будет больше 40 итераций, ну или точнее надо считать вероятности:)) Главное если вы не укладываетесь в свой расчет, то не надейтесь на чудо — сносите и все, делать заново т.к. вы не впишитесь;

- Данные могут загружаться набегающей волной – с последовательным обновлением данных одного и того-же периода в будущем в течение нескольких последовательных периодов. (например: обновление прогноза окончания года каждый месяц). Поэтому кроме справочника «Период», должен быть предусмотрен технический справочник «Период загрузки», который позволит изолировать процессы загрузки данных в разных периодах и не потерять историю изменения цифр;

- Данные имеют обыкновение быть перегружаемыми много раз, и хорошо если будет технический справочник «Версия» как минимум с двумя элементами «Рабочая» и «Финальная», для отделения вычищенных данных. Кроме-того создание персональных версий, одной суммарной и одной финальной позволяет хорошо контролировать загрузку в несколько потоков;

- Данные всегда содержат ошибки: Перезагружать весь пакет в [50GB -> +8] это очень не экономно по ресурсам и вы, скорее всего, не впишитесь в

регламент, следовательно, надо грамотно делить загружаемый пакет файлов и так проектировать систему, чтобы она позволяла обновлять пакет по маленьким частям. По моему опыту лучший способ – техническая аналитика «файл-источник», и интерфейс, который позволяет снести все данные только из одного файла, и вставить вместо него обновленные. А сам пакет разумно делить на файлы по количеству исполнителей, ответственных за их заполнение (либо админы систем готовящие выгрузки, либо пользователи заполняющие вручную);

- При проектировании разделения пакета на части надо еще учитывать возможность так-называемого «обогащения» данных (например: Когда 12 января считают налоги прошлого года по правилам управленческого учета, а в марте-апреле перегружают суммы на посчитанные по бухгалтерскому), это решается с одной стороны правильным проектированием деления пакета данных на части так, чтобы для обогащения надо было перегрузить целое количество файлов (не 2,345 файла), а с другой стороны введением еще одного технического справочника с периодами обогащения, чтобы не потерять историю изменений по этим причинам).

Процесс валидации

Данный процесс отвечает за выявление ошибок и пробелов в данных, переданных в ETL. Само программирование или настройка формул проверки не вызывает вопросов, главный вопрос – как вычислить возможные виды ошибок в данных, и по каким признакам их идентифицировать? Возможные виды ошибок в данных зависят от того какого рода шкалы применимы для этих данных. (Ссылка на прекрасный пост, объясняющий, какие существуют виды шкал — http://habrahabr.ru/post/246983/).

Ближе к практике в каждом из передаваемых типов данных в 95% случаев возможны следующие ошибки:

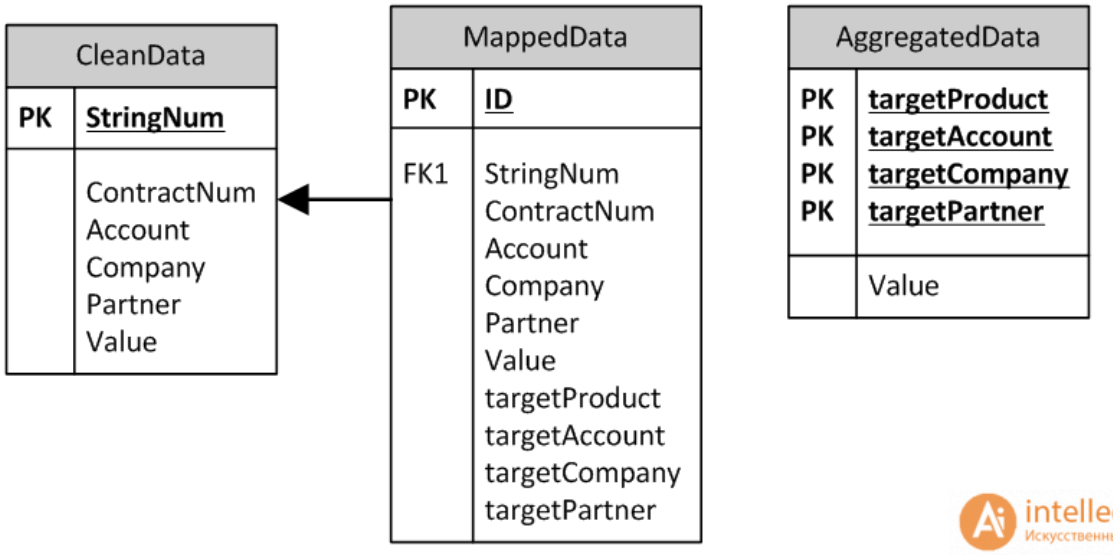
Типы данных	Внутри поля	По отношению к другим полям	Совместимость форматов при передаче между системами
Перечисление и текст	• Не из списка разрешенных значений • Отсутствие обязательных значений • Не соответствие формату (Все договора должны нумероваться «ДГВхххх..»)	• Не из списка разрешенных значений для связанного элемента • Отсутствие обязательных элементов для связанного элемента • Не соответствие формату для связанного элемента(например: для продукта «АИС» все договора должны нумероваться «АИСхххх..»)	• Символы допустимые в одном формате, недопустимы в другом • Кодировка • Обратная совместимость (Элемент справочника был изменен в целевой системе без добавления мэппинга) • Новые значения (нет мэппинга) • Устаревшие значения (не из списка разрешенных в целевой системе)
Числа и порядки	• Не число • Не в границах разрешенного интервала значений • Пропущено порядковое значение (например: данные не дошли)	• Не выполняется отношение y=ax+b (например: НДС и Выручка, или Встречные суммы равны) • Элементу «А» присвоен неправильный порядковый номер • Разницы за счет разных правил округления значений (например: в 1С и SAP никогда не сходится рассчитанный НДС)	• Переполнение • Потеря точности и знаков • Несовместимость форматов при конвертации в не число
Даты и периоды		• День недели не соответствует дате • Сумма единиц времени не соответствует из-за разницы рабочие/не рабочие/праздничные/ сокращенные дни	• Несовместимость формата даты при передаче текстом (например: ISO 8601 в UnixTime, или разные форматы в ISO 8601) • Ошибка точки отсчета и точности при передаче числом (например: TimeStamp в DateTime)

Соответственно проверки на ошибки реализуются либо формулами, либо скриптами в редакторе конкретного ETL-инструмента. А если вообще по большому счету, то большая часть ваших валидаций будет на соответствие справочников, а это [select * from a where a.field not in (select...)] При этом для сохранения аудиторского следа разумно сохранять в системе две отдельные таблицы – rawdata и cleandata с поддержкой связи 1:1 между строками.

Процесс мэппинга

Процесс мэппинга так же реализуется с помощью соответствующих формул и скриптов, есть три хороших правила при его проектировании:

- Таблица замэпленных данных должна включать одновременно два набора полей – старых и новых аналитик, чтобы можно был сделать select по исходным аналитикам и посмотреть, какие целевые аналитики им присвоены, и наоборот:



- Таблица замэпленных элементов должна иметь отдельное PK-поле, чтобы при связи 1:* и *:* можно было создать много строк в MappedData для одной строки в CleanData и сохранить аудиторский след

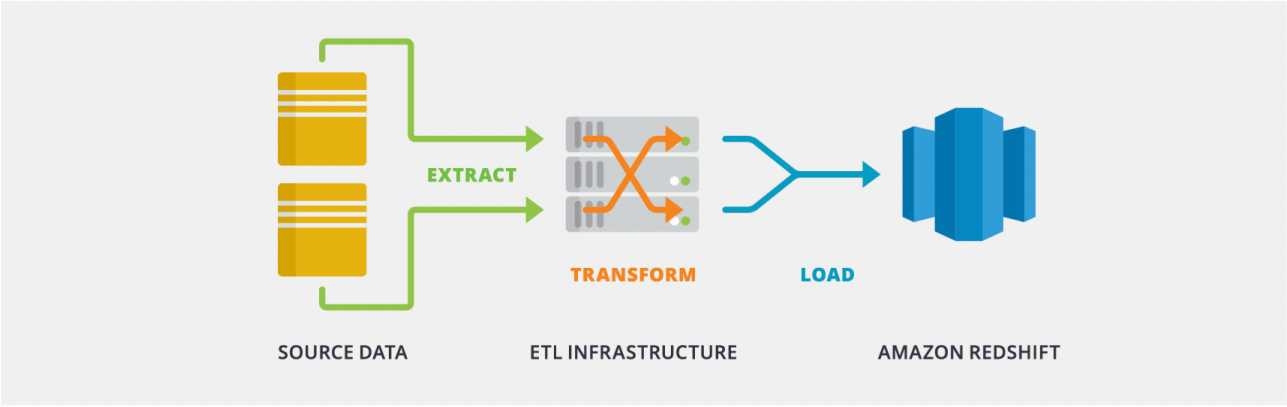
- Таблица MappedData должна быть отдельной от CleanData по тем же причинам что и пункт 2

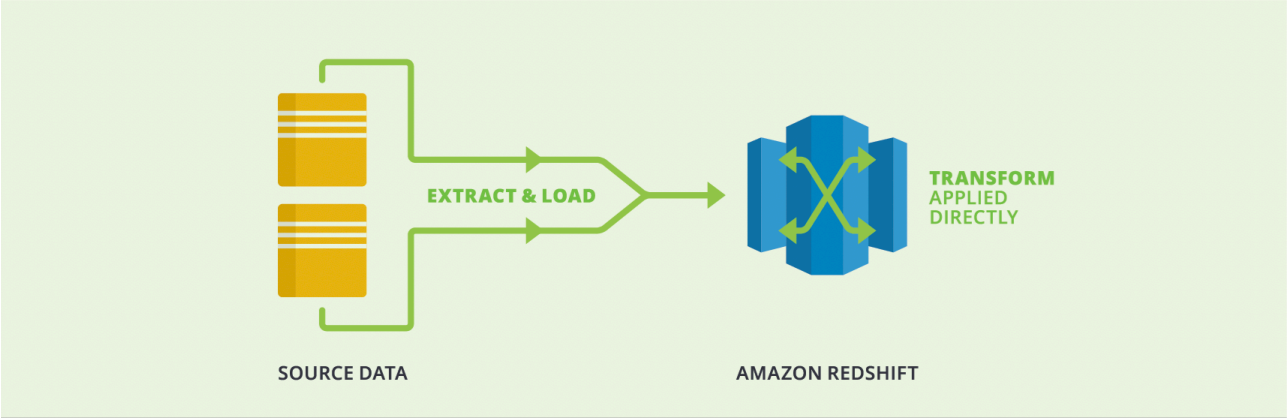
Заключение

В принципе это все архитектурные приемы, которые мне понравились в тех ETL инструментах, которыми я пользовался.
Кроме этого конечно в реальных системах есть еще сервисные процессы — авторизации, разграничения доступа к данным, автоматизированного согласования изменений, и все решения конечно являются компромиссом с требованиями производительности и предельным объемом данных.

Основное различие между ETL и ELT, где происходит обработка

Обработка ETL данных происходит в инструменте ETL (как правило, запись по времени и в памяти) Обработка ELT данных происходит в двигателе базы данных





Данные одинаковы, и конечные результаты данных могут быть достигнуты в обоих методах. Это очень зависит от вас и вашей среды Если у вас сильный движок базы данных и хорошее аппаратное обеспечение, и вы можете выполнять тяжелую обработку, ELT хорош для вас, Если у вас есть загруженный механизм datawarehouse, и вам нужно освободить его от обработки, перейдите на ETL.
Обратите внимание, что наличие инструмента ETL дает вам обе опции, такие как ETL (T), вы можете выполнить преобразование в инструменте ETL, и вы также можете сделать преобразование в движке базы данных, но ELT у вас есть только возможность преобразования в движке базы данных, но вы должны знать, что базы данных лучше работают на основе набора, чем инструменты ETL по времени.