# Допущения

1. Принимаем, что ключ в Системе 1 (Справочник товар – категория) это «**Номер товара**». \**Возможный вариант*\*: «**Номер товара**» + «**Название товара**», пример:

|  |  |  |  |
| --- | --- | --- | --- |
| Номер товара | Название товара | Номер категории | Название категории |
| 1001 | яблоки 1 кг | 1 | Фрукты |
| 1001 | яблоки 3 кг | 1 | Фрукты |

1. Принимаем тот факт, что в Системе 1 (Справочник товар – категория) атрибут «**Название категории**» не может быть частью естественного ключа, а является частью измерения. Так принято, в соответствии с описанием источника.
2. Принимаем, что в разрезе дат аналитика строится так: товар в заданную дату, соответствует той категории, которая у него была на заданную дату. То есть, если у него [2024.01.01-2024.06.30] была категория «Спорт», а после этого и по настоящее время является категория «Обувь», тогда для аналитика до 2024.06.30 будет рассматриваться этот товар как категория «Спорт», а далее строиться как «Обувь». То-есть для бизнеса ставим цель, что важно сохранить исторический контекст — не пересчитывать, а фиксировать изменения с определённой даты. \*Возможный вариант\*: рассматривать только актуальную категорию по товару для аналитики за весь 2024-ый год, возможно такая задача реальна, как ad-hoc (Возможно именно это и имелось в виду в разделе «Будущая визуализация» в задании), также если важна точность для отчетности и прогнозирования на новой категории, при изменении аналитики в отношении метрики, то следует учитывать актуальную дату и пересчитать всё.
3. Система 1 (Таблица фактических продаж) содержит информацию по всем продажам компании, однако в описании источника и ТЗ нигде не говорится о том, где хранится информация о возвратах и нужно ли ее учитывать при построении дашборда, что было бы логично. В соответствии с имеющейся информацией принимаем допущение, что товары невозможно вернуть обратно. \*Возможные вариант – 1) В таблице фактов будет строка с другой датой временем, но тем же номером заказа и номером товара, где сумма будет уже отрицательной, в таком случае, необходима будет дополнительная агрегация при реализации построения витрины. 2) Будет дополнительная таблица по возвратам, где будет содержаться информация по возвратам. В соответствии с этой таблицей тогда придется корректировать данные, которые будут поступать на дашборд.
4. Принимаем следующее допущение, что атрибут «**Дата**» в Системе 1 – таблица фактических продаж, является частью естественного ключа таблица, лишь из-за того, что это ключ партиции, в которой содержится запись по «**Номер заказа**» + «**Номер продукта**». \*Возможный вариант\* существует еще метрики / атрибуты в источнике по фактическим продажам, которые зависят от даты, поэтому необходимо сохранять полный естественный ключ.
5. Что данные в Системе 2 «Таблица с рекламными вложениями продавцов» добавляет на ежедневной основе, только те данные в источник, которые были обновлены (если стоимость рекламы не была изменена, тогда эта запись не будет добавлена, так как есть более ранняя запись, которая зафиксировала цену).
6. Также принимаем тот факт что везде данные не имеют дублей и все источники существуют с наложенными на них ограничениями в ddl схеме при создании таблиц.
7. Допускаем что в данной в ТЗ функции overflow есть 2 варианта отработки. replace – перезаписывает данные; append – добавляет данные (инкремент).
8. Если append невозможно применять в комбинации с truncate таблицы в Системе 3, тогда создаем реплики (RAW) изначальных источников. Иначе данные сразу уже можно делать в (STG) слое, где справочник по категориям будет полностью перезалит, а у остальных источников содержаться только инкремент данных.
9. Допустим, что если даты нету в таблице с рекламными вложениями продавцов то в эту дату он не тратился на заданный продукт.

# Вопросы

1. Где содержится информация, что Система 1 (справочник товар – категория) обновилась?
2. Можно ли зацепиться за ETL поток, который обновляет Система 1, чтобы потом по триггеру запускать следующий поток?
3. Естественный ключ Системы 1 (справочник товар – категория), это номер товара? Или же это номер товара + название?
4. Не совсем понятно, что значит высказывание в отношении возможностей окна: «категоризация должна быть в актуальном состоянии»? Требуются уточнения! А так как их нет, придется принимать ответственность за то, как считаем корректным проводить аналитику, после сбиться с аналитиками, как реально надо.
5. Для чего дата в источнике Система 1 – таблица фактических продаж относится к части естественного ключа таблицы? Какой смысл несет данное нагромождение ключа.
6. Может ли функция overflow (replace) делать, но при этом заливать только инкремент из источника? Если да, тогда везде выбираем вариант replace. Второй вариант, если невозможно, тогда нужно заливать продажи и рекламу через обновление (append) и используем табличку как источник, но уже внутри Системы 3 как реплику из источника Система 1 или Система 2.
7. Рекламные вложения если, дата была пропущена по заданному продукту, это значит что сумма не изменилась по рекламным вложениям и стоимость тарифа на рекламу остается той же или это значит что продавец не имел и не платил за рекламу в эту дату?

# Ход работы

## Описание таблиц в системах

Описание источников в системе 1:

* Источник «Справочник товар – категория», содержит только актуальную на сегодняшнюю дату i в себе. То-есть полностью обновляется и содержит только актуальную информацию. Поэтому, помимо того, чтобы завязаться на обновление этого источника, необходимо разработать схему SCD (именно SCD, а не RCD), так как данные являются информацией о категории товара, а не о транзакциях и продажах. Справочники имеют тип SCD, поэтому необходимо подобрать тип SCD и разработать витрину, которая может сохранять историчность данных и поможет строить аналитику в разрезе дат. Принимаем, что ключом является атрибут «Номер товара» по соответствующим допущениям.
* Источник «Таблица фактических продаж» содержит в себе факт об операциях, связанных с продажами. Обновляется инкрементально и данные добавляются в источник, а не полностью его перезаписывают. А что насчет возвратов? Примем допущения, что такое событие мы не учитываем или принимаем его невозможным, так как отсутствует точное описание, как такое событие обрабатывать. Также в таблице отсутствует состояние заказа / продажи, так что исходя из данного факта, принимаем, что все записи имеет завершённый статус заказа. То есть одна запись является реальной завершенной продажей продукта. Естественным ключом в таблице с рекламными вложениями продавцов будет связь «**Номер заказа**» + «**Номер товара**», дата же в данном случае выступает как ключ партиции в которой находится запись, для ускорения поиска по данным. Иначе не совсем понятно почему естественный ключ состоит из трех полей.

Описание источника из системы 2:

* Источник «Таблица с рекламными вложениями продавцов» содержит информацию по всем рекламным вложениям продавцов компании. Ежедневно таблица пополняется данными за вчерашний день. Естественным ключом является «**Дата**» + «**Номер товара**», то есть данные за 1 день не могут дублироваться по заданным полям. Источник уже имеет четкую версионность кортежей данных в зависимости от даты. SCD следует применить в таком, формате каждая строчка соответствует началу действия предложения на заданный продукт. Поэтому необходимо будет использовать диапазоны дат start\_dt + end\_dt для учета версии рекламного вложения продавцов.

Выбор колоночной СУБД - ClickHouse.

1. Высокая скорость работы:

* Использование партиционирования по интервалам дат ускоряет запросы.
* Колоночное хранение снижает объем используемой памяти.

1. Оптимизация загрузки данных:

* Полная перезапись STG с последующей инкрементальной обработкой в DDS.

1. Гибкость аналитики:

* Возможность динамического построения отчетности по историческим данным.

## План работы

SQL-запросы (CDM):

* Базовые запросы по продажам.
* Запросы по рекламным вложениям.
* Объединенные запросы для построения единой витрины данных.
* Учет по витрине, что необходима интерактивность в выборе интервалов дат, категорий и метрик аналитики.
* \*(Денормализованная партиционированная таблица).

Структура данных в системе 3 (DDS):

* Механизм версионности категорий (SCD 2).
* Разработка модели данных.
* Ключи в dds таблицах.

Разработка STG (Stage):

* Формирование временных таблиц для первичной загрузки данных из источников или RAW слоя.
* Очистка, приведение типов.
* Контроль качества данных (проверка на дубли и пропущенные значения).
* Логирование загрузок (Возможно создание и добавление информации в техническую табличку для учета контроля data flow на всех слоях данных и потоке ETL).

ETL-процесс:

* Опишем поток данных с учетом существующего механизма "overflow".
* Подберем структуру таблиц в системе 3. Data flow – (RAW)\* → STG → DDS → CDM.
* Определим тип SCD (SCD 2) и хранение историчности категорий.
* Триггерная зависимость от источников / Временная отсечка по загрузке данных.
* \*Создание таски триггера на завершение потока.

SQL-запросы для фронта:

* Фильтрация по дате, категории, метрикам.
* Объединение данных продаж и рекламы.

## Ход решения

Для начала уточним что выполняем 2 варианта параллельно, так как насчет работы overflow ничего неизвестно и пока нельзя протестировать как она работает. Единственное к чему это приведет, это использование разных источников для STG слоя в Системе 3.

Примем что используем вариант с тем, что в Системе 3 после работы overflow, теперь содержаться реплика таблиц из Системы 1 и Системы 2.

ETL-процесс по слоям в ClickHouse

1. STG (Stage)

* Перезапись данных по продажам и рекламе (полный снимок или инкремент).
* Полная перезапись справочника товаров и категорий.

1. DDS (Data Vault 2.0, SCD2)

* Историзация категорий с учетом версий (ReplacingMergeTree).
* Инкрементальная загрузка данных по фактам.
* Разбиение информации по модели данных Data Vault 2.0

1. CDM (Финальная витрина)

* Агрегированные данные по категориям, времени и метрикам.

STG:

При создании используем стандартный движок MergeTree с сортировкой по естественному ключу. Чтобы заливать данные используем комбинацию команд TRUNCATE + INSERT.

* Заливаем данные AS-IS.
* Category – replace (truncate + insert)
* Sales + Marketing – (truncate + insert) берем только инкремент данных.

DDS:

* Используем модель Data Vault 2.0, разделяя данные на Hub, Satellite, Link.
* Включаем поля date\_from и date\_to для категорий.
* Реализуем схемы поэтапно: сначала Hub, затем Satellite, и в конце Link.
* Подготовим скрипты DDL и ETL для каждого слоя.

Детальный слой

Необходимо построить hub. Поэтому разбиваем все таблицы на отдельные возможные атомарные сущности: «Продажа» – sale, «Категория» - category, «Продукт» - product, «Реклама» - marketing. В всех hub, кроме рекламы будет бизнес ключ (БК) id сущности, а в рекламе это будет связь «продукт» + «дата». Во всех hub будет дата загрузки и система источник, а также самое важное это внутренний суррогатный hash ключ.

Почему hash ключ:

* Хэш-ключи обеспечивают быстрый поиск и объединение данных в ClickHouse.
* Поиск по хэшированным значениям выполняется быстрее, чем по строковым бизнес-ключам (особенно если БК длинные, составные или не имеют четкого формата).
* Хэш-ключи имеют фиксированный размер (например, 64 бита для UInt64), что упрощает индексацию и ускоряет джойны.
* Хэш-ключ позволяет унифицировать представление, сводя разные форматы к единообразному виду.
* Хэш-ключ устраняет зависимость от изменения формата ключей в источниках данных.

Load\_dt нужна чтобы фиксировать дату загрузки нового БК и также для дедупликации данных и отслеживания истории. Для дедупликации используется в комбинации с Engine ClickHouse – ReplacingMergeTree.

Также задаем PARTITION BY (load\_dt) чтобы иметь быстрый доступ к срезам данных за конкретные даты. Можно оптимизировать тем, что уменьшить количество партиций, с гранулярностью до месяца, но это не целесообразно, в данной задаче, где будут различные разрезы в интервале дат и также имеется большой поток данных в разрезе дня.

|  |  |  |  |
| --- | --- | --- | --- |
| **Хаб** | **Бизнес-ключ (БК)** | **Состав хэш-ключа** | **Дата загрузки** |
| hub\_sales | sale\_id | cityHash64(sale\_id) | load\_dt |
| hub\_marketing | product\_id + marketing\_dt | cityHash64(product\_id, marketing \_dt) | load\_dt |
| hub\_products | product\_id | cityHash64(product\_id) | load\_dt |
| hub\_category | category\_id | cityHash64(category\_id) | load\_dt |

Использование ON DUPLICATE KEY UPDATE в сочетании с уникальным хэшем бизнес-ключа и движком ReplacingMergeTree:

* Обеспечивает идемпотентность загрузки — данные можно загружать многократно без создания дублей.
* Обновляет load\_dt для актуализации записей, если строки уже существуют.
* Гарантирует, что в таблице будет храниться только последняя версия бизнес-ключа.

Таким образом, если таблица в STG перезаливается каждый день, то в HUB попадет только уникальный набор бизнес-ключей, а если бизнес-ключ уже существует — он просто обновится

# Вывод

1. Гибкость: позволяет легко вносить изменения в источники данных.
2. Производительность: ClickHouse обеспечивает быстрые аналитические запросы благодаря колоночному хранению.
3. Историчность: Data Vault 2.0 позволяет сохранять полную версионность данных.