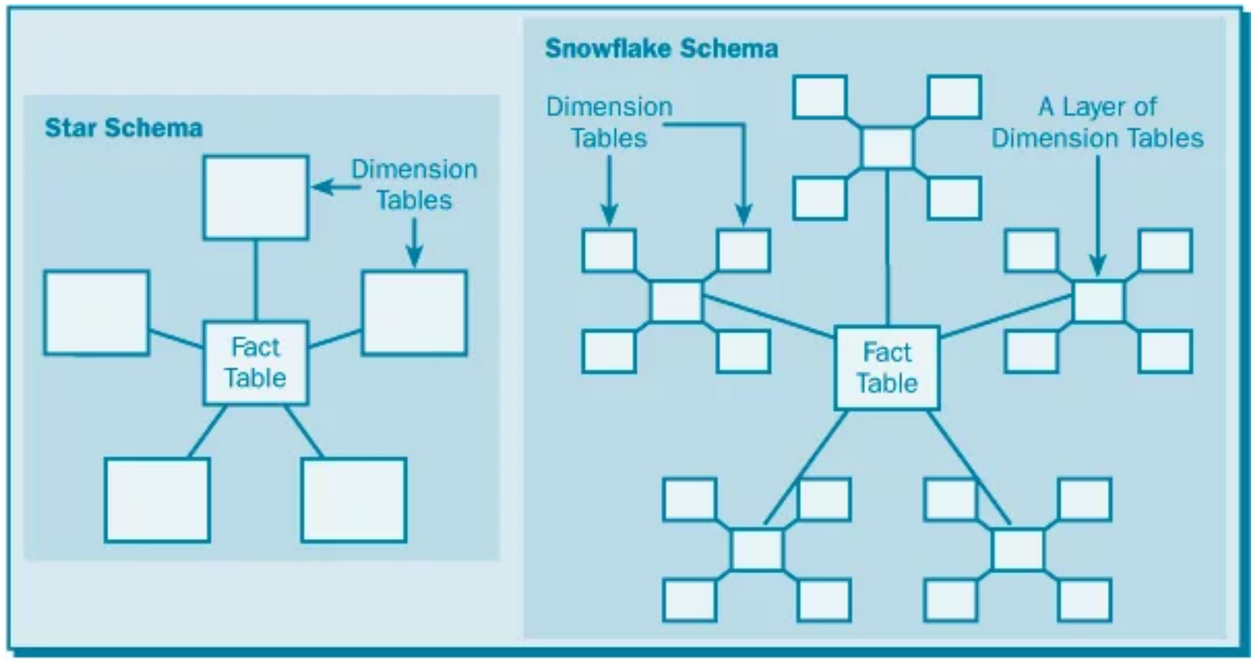
Гибкие методологии проектирования хранилищ данных

Модель DataWarehouse (DWH)

На текущий момент в качестве модели DWH может быть выбрана одна из следующих трех моделей хранения данных:

- Снежинка
- DataVault 2.0
- Якорная модель

Снежинка/Группа звездочек



Плюсы:

- Настройка проверок консистентности данных
- Структура соотносится с моделью данных

Минусы:

- Модель не подходит для крупных проектов или для проектов с быстро меняющейся структурой данных
- При внесении и добавлении данных в хранилище требуется множество согласований
- Снижается производительность запросов

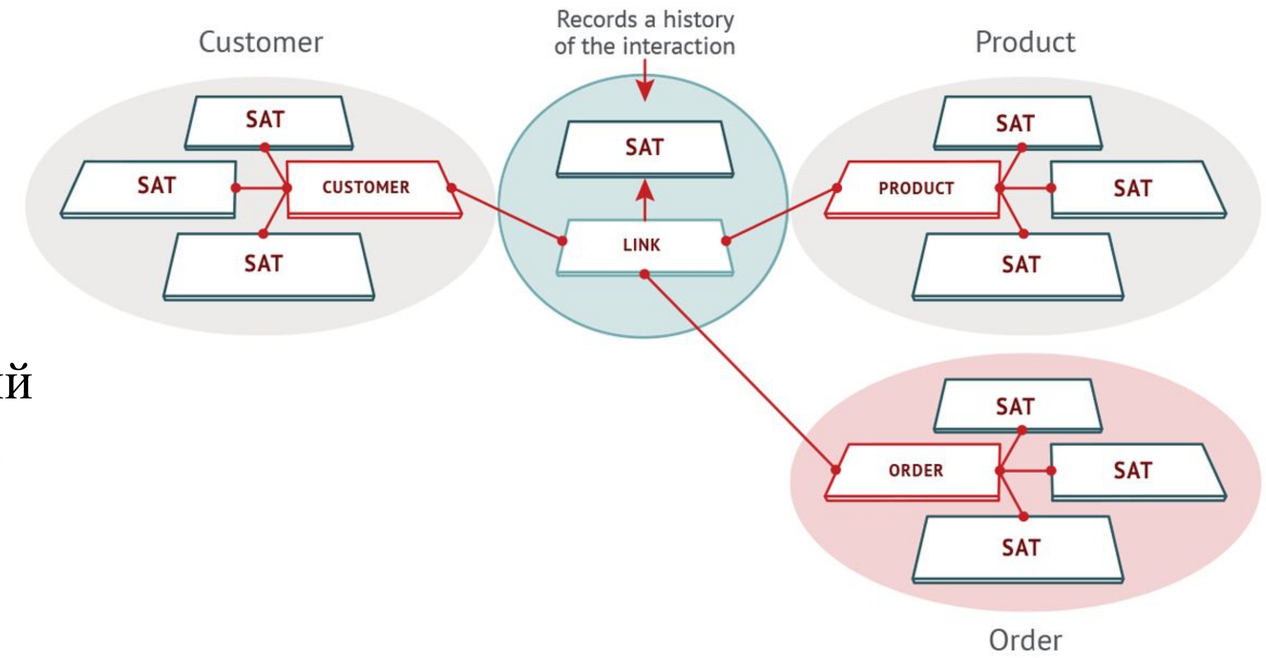
DataVault 2.0

Данные в Data Vault представлены в виде трех сущностей:

- **Хаб (hub)** – бизнес-сущность, например клиент, продукт, заказ. Хаб-таблица содержит несколько полей: натуральные ключи сущности, суррогатный ключ, ссылка на источник записи и время добавления записи.

- **Линк (связь, link**) – представление отношений между Хабами. Линки строятся в виде таблиц «многие ко многим» и содержат ссылки на суррогатные ключи связанных Хабов.

- **Сателлит (satellite)** - изменяемые атрибуты сущностей, контекстные данные для хаб-таблицы, связанные с Хабами и Связями по принципу «один ко многим».



DataVault 2.0

Плюсы Data Vault:

- Универсальная структура, которая позволяет сформировать витрины под любые потребности бизнеса
- Первые верхнеуровневые отчеты доступны уже после загрузки Хабов и Связей
- Можно независимо разрабатывать смежные источники и легко заменять их, так как данные почти полностью отвязаны друг от друга

Минусы Data Vault:

- Большое количество таблиц и join'ов, которые замедляют выполнение запросов относительно денормализованых хранилищ
- Сложность выстраивания проверки данных из-за большого количества сущностей и сложных связей между ними
- Методология требует от специалиста дополнительных навыков и знаний особенности работы специфических функций, например хеширования

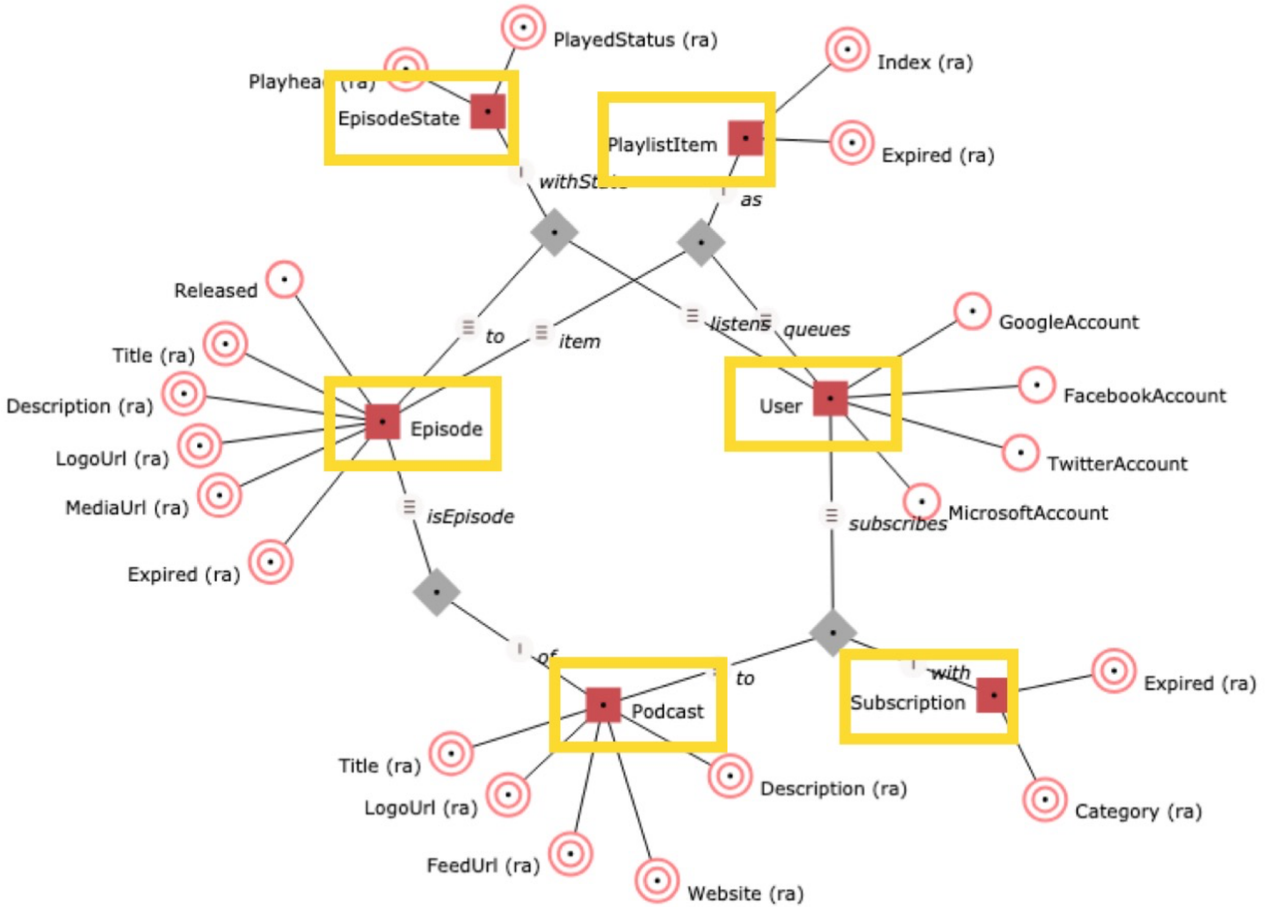
DataVault 1.0 vs DataVault 2.0

Аспект	Data Vault 1.0	Data Vault 2.0
Хэш-ключи	Хэш-ключи не были центральной концепцией, ограничивающей целостность и отслеживаемость данных.	Отдает приоритет хэш-ключам, обеспечивая целостность данных и улучшая отслеживаемость для повышения
Процедуры загрузки	Процедуры загрузки в Data Vault 1.0 могут быть сложными и часто включать порядковые номера, что влияет на эффективность.	Упрощает процедуры загрузки, повышает эффективность и устраняет необходимость в сложных порядковых номерах.
Зависимости	Имели значительные зависимости, потенциально замедляющие загрузку данных из-за последовательной обработки.	Уменьшает зависимости, обеспечивая более быструю обработку данных за счет распараллеливания.
Масштабируемость	Столкнулся с проблемами при работе с большими наборами данных из-за ограничений конструкции.	Эффективно обрабатывает большие данные, что делает его пригодным для сложных наборов данных.
Гибкость	Менее адаптируется к изменениям в источниках данных и бизнес-требованиях.	Гибкость и оперативность реагирования на изменения, идеально подходят для динамичных сред.

Якорная модель

Данные в якорной модели представлены в виде трех сущностей:

- **Якорь** – представляет собой сущность или событие, содержит суррогатные ключи, ссылку на источник и время добавления записи
- **Атрибут** – используется для моделирования свойств и характеристик якорей, содержит суррогатный ключ якоря, значение атрибута, ссылку на источник записи и время добавления записи
- **Связь** – моделирует отношения между якорями
- **Узел** – используется для моделирования общих свойств (состояния)



Якорная модель

Плюсы Anchor Modeling:

- Нормализованная модель, которая эффективно обрабатывает изменения и позволяет масштабировать хранилища данных без отмены предыдущих действий
- Можно независимо разрабатывать смежные источники, так как данные почти полностью отвязаны друг от друга
- Значительная экономия места в связи с отсутствием нулевых значений (null) и дублирования

Минусы Anchor Modeling:

- Создает высокую нагрузку на базу даже для основных видов запросов
- Сложно настроить проверку данных, так как модель состоит из большого количества сущностей со сложными связями
- Большое количество таблиц и join'ов, повышающих риск ошибок
- Требует постоянной поддержки и документирования, так как документация уникальна для каждого бизнеса. Специалистам необходимы дополнительные знания для понимания документации.

Как определить, какая модель подойдет лучше всего?