|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ |
| ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ |
| **«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»** |
| **(МОСКОВСКИЙ ПОЛИТЕХ)** |
| Факультет информационных технологий |

Кафедра «Прикладная информатика»

Форма обучения: очная

**ОТЧЕТ**

**о выполнении лабораторной работы по Data Vault**

**по дисциплине**

**«Хранилища данных»**

Студент (Е. А. Азарова)

(личная подпись)

**Москва 2021**

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 3](#_Toc87738778)

[Задание 4](#_Toc87738779)

[Создание реляционной базы данных обменного пункта 4](#_Toc87738780)

[Структура Data Vault 6](#_Toc87738781)

[Преимущества и недостатки Data Vault 9](#_Toc87738782)

[Создание БД в соответствии с моделью Data Vault. 11](#_Toc87738783)

[**ЗАКЛЮЧЕНИЕ** 14](#_Toc87738784)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 15](#_Toc87738785)

[**ПРИЛОЖЕНИЕ А** 16](#_Toc87738786)

# **ВВЕДЕНИЕ**

Большинство компаний сегодня накапливают различные данные, полученные в процессе работы. Часто данные приходят из различных источников — структурированные и не очень, иногда в режиме реального времени, а иногда они доступны в строго определенные периоды. Все это разнообразие нужно структурированно хранить, чтоб потом успешно анализировать, рисовать красивые отчеты и вовремя замечать аномалии. Для этих целей проектируется [хранилище данных](https://en.wikipedia.org/wiki/Data_warehouse) (Data Warehouse, DWH).

Существует несколько подходов к построению такого универсального хранилища, которые помогают архитектору избежать распространенных проблем, а самое главное обеспечить должный уровень гибкости и расширяемости DWH. Сегодня будет разобран один подход.

Data Vault — гибридный подход, объединивший достоинства знакомой многим схемы «звезды» и 3-ей нормальной формы. Впервые эта методология была анонсинована в 2000 году Дэном Линстедтом (Dan Linstedt). Подход был придуман в процессе разработки хранилища данных для Министерства Обороны США и хорошо себя зарекомендовал. Позже, в 2013 году, Дэн анонсировал версию 2.0, доработанную с учетом быстро набравших популярность технологий (NoSQL, Hadoop) и новых требований, выставляемых к DWH [1].

# **Задание**

В любой реляционной СУБД реализуйте структуру базы данных в соответствии с моделью 3NF. Каждый берет свой вариант. Тема моего варианта: Обменный пункт: сотрудники пункта, виды валют, курсы валют, операции обмена.

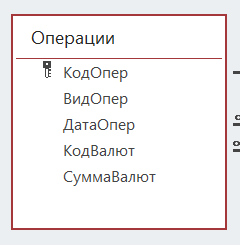
В СУБД PostgreSQL реализуйте структуру базы данных в соответствии с моделью Data Vault. Результат представьте в виде даталогической модели и задокументированного SQL кода создания базы данных.

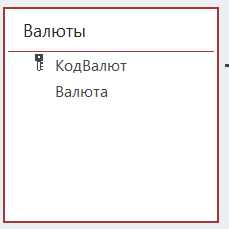
# **Создание реляционной базы данных обменного пункта**

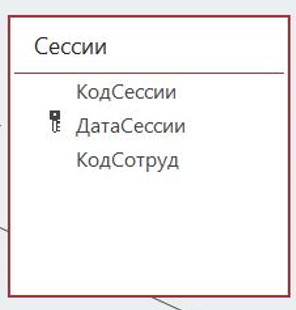
Рассмотрим обменный пункт валюты крупного банка, который обеспечивает обмен разного вида валют (доллары США, евро, фунты стерлингов, японские йены, рубли и т.д.). В базе данных должна храниться информация: кто и когда произвел обмен одной валюты на другую и в каком количестве. Анализ предметной области показывает, что должны быть следующие сущности:

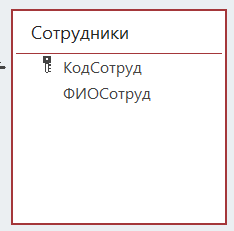
* Сотрудник;
* валюта;
* валютная операция
* курсы валют.
* сессия
* посещение

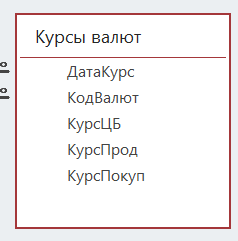
Для начала мы создаем таблицы проектируемой базы данных. На рисунках ниже представлены разработанные таблицы реляционной базы данных:

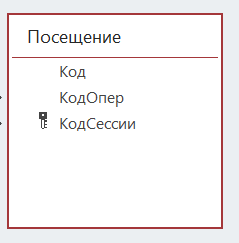
  
Рисунок 1 – операции

  
Рисунок 2 – валюты

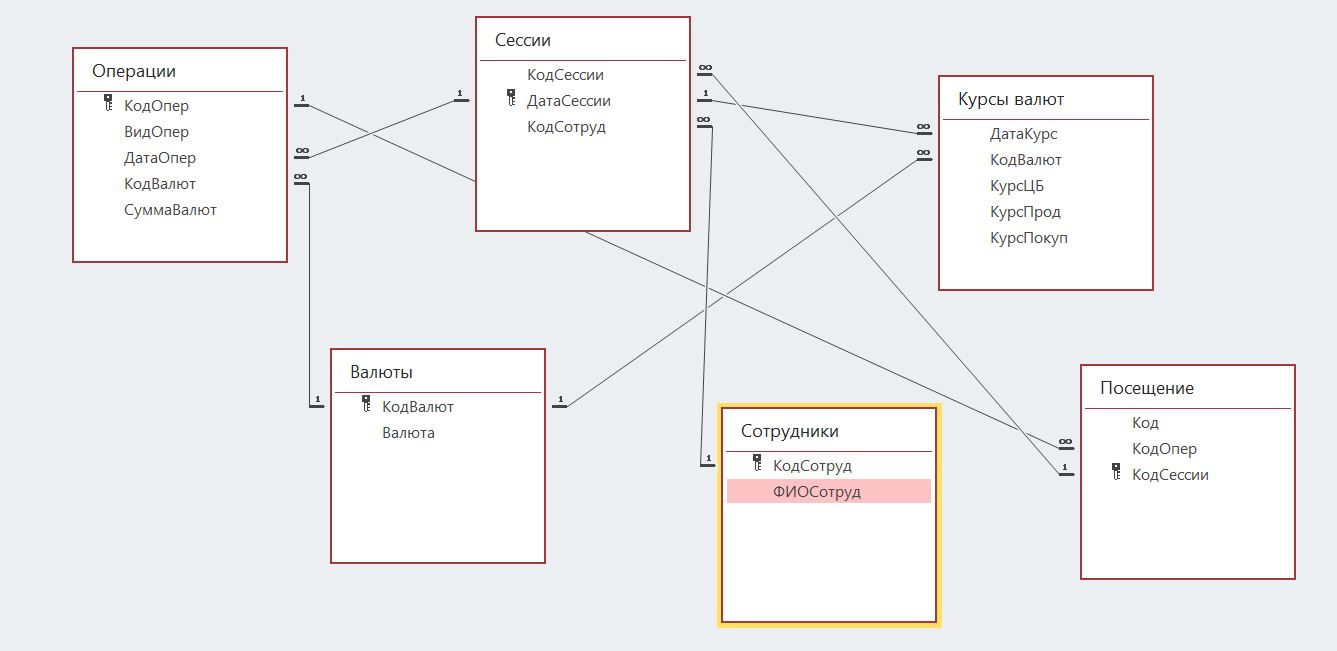
  
Рисунок 3 – сессии

  
Рисунок 4 – сотрудники

  
Рисунок 5 – курсы валют

  
Рисунок 6 - посещение

Затем формируем схему связи данных:

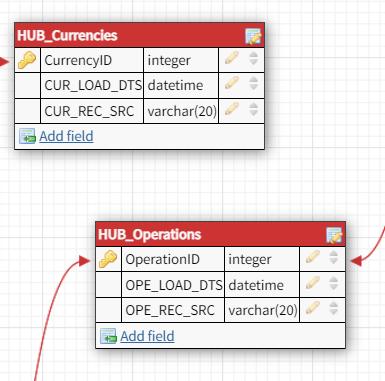
  
Рисунок 7 – реляционная БД обменного пункта

# **Структура Data Vault**

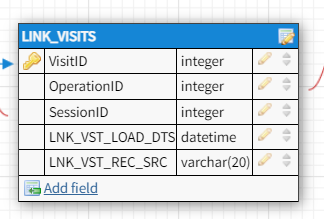
Data Vault состоит из трех основных компонентов — Хаб (Hub), Ссылка (Link) и Сателлит (Satellite).

Хаб — основное представление сущности (Клиент, Продукт, Заказ) с позиции бизнеса. Таблица-Хаб содержит одно или несколько полей, отражающих сущность в понятиях бизнеса. В совокупности эти поля называются «бизнес ключ». Идеальный кандидат на звание бизнес-ключа это ИНН организации или VIN номер автомобиля, а сгенерированный системой ID будет наихудшим вариантом. Бизнес ключ всегда должен быть уникальным и неизменным.

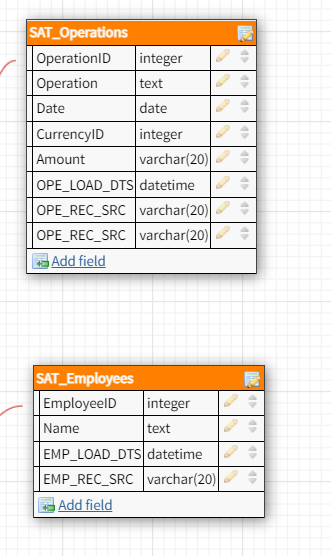
Хаб так же содержит мета-поля load timestamp и record source, в которых хранятся время первоначальной загрузки сущности в хранилище и ее источник (название системы, базы или файла, откуда данные были загружены). В качестве первичного ключа Хаба рекомендуется использовать MD5 или SHA-1 хеш от бизнес ключа.

  
Рисунок 8 – таблицы-хабы

Таблицы-Ссылки связывают несколько хабов связью многие-ко-многим. Она содержит те же метаданные, что и Хаб. Ссылка может быть связана с другой Ссылкой, но такой подход создает проблемы при загрузке, так что лучше выделить одну из Ссылок в отдельный Хаб.

  
Рисунок 9 – таблица-ссылка

Все описательные атрибуты Хаба или Ссылки (контекст) помещаются в таблицы-Сателлиты. Помимо контекста Сателлит содержит стандартный набор метаданных (load timestamp и record source) и один и только один ключ «родителя». В Сателлитах можно без проблем хранить историю изменения контекста, каждый раз добавляя новую запись при обновлении контекста в системе-источнике. Для упрощения процесса обновления большого сателлита в таблицу можно добавить поле hash diff: MD5 или SHA-1 хеш от всех его описательных атрибутов. Для Хаба или Ссылки может быть сколь угодно Сателлитов, обычно контекст разбивается по частоте обновления. Контекст из разных систем-источников принято класть в отдельные Сателлиты.

  
Рисунок 10 – таблица-сателлит

Сначала данные из операционных систем поступают в staging area. Staging area используется как промежуточное звено в процессе загрузки данных. Одна из основных функций Staging зоны это уменьшение нагрузки на операционные базы при выполнении запросов. Таблицы здесь полностью повторяют исходную структуру, но любые ограничения на вставку данных, вроде not null или проверки целостности внешних ключей, должны быть выключены с целью оставить возможность вставить даже поврежденные или неполные данные (особенно это актуально для excel-таблиц и прочих файлов). Дополнительно в stage таблицах содержатся хеши бизнес ключей и информация о времени загрузки и источнике данных.

После этого данные разбиваются на Хабы, Ссылки и Сателлиты и загружаются в Raw Data Vault. В процессе загрузки они никак не агрегируются и не пересчитываются.

Business Vault — опциональная вспомогательная надстройка над Raw Data Vault. Строится по тем же принципам, но содержит переработанные данные: агрегированные результаты, сконвертированные валюты и прочее. Разделение чисто логическое, физически Business Vault находится в одной базе с Raw Data Vault и предназначен в основном для упрощения формирования витрин.

# **Преимущества и недостатки Data Vault**

*[+] Гибкость и расширяемость.*

С Data Vault перестает быть проблемой как расширение структуры хранилища, так и добавление и сопоставление данных из новых источников. Максимально полное хранилище «сырых» данных и удобная структура их хранения позволяют нам сформировать витрину под любые требования бизнеса, а существующие решения на рынке СУБД хорошо справляются с огромными объемами информации и быстро выполняют даже очень сложные запросы, что дает возможность виртуализировать большинство витрин.

*[+] Agile-подход из коробки.*

Моделировать хранилище по методологии Data Vault довольно просто. Новые данные просто «подключаются» к существующей модели, не ломая и не модифицируя существующую структуру. При этом мы будем решать поставленную задачу максимально изолированно, загружая только необходимый минимум, и, вероятно, наша временнáя оценка для такой задачи станет точнее. Планирование спринтов будет проще, а результаты предсказуемы с первой же итерации.

*[–] Обилие JOIN'ов*

За счет большого количества операций join запросы могут быть медленнее, чем в традиционных хранилищах данных, где таблицы денормализованы.

*[–] Сложность.*

В описанной выше методологии есть множество важных деталей, разобраться в которых вряд ли получится за пару часов. К этому можно прибавить малое количество информации в интернете и почти полное отсутствие материалов на русском языке (надеюсь это исправить). Как следствие, при внедрении Data Vault возникают проблемы с обучением команды, появляется много вопросов относительно нюансов конкретного бизнеса. К счастью, существуют ресурсы, на которых можно задать эти вопросы. Большой недостаток сложности - это обязательное требование к наличию витрин данных, так как сам по себе Data Vault плохо подходит для прямых запросов.

*[–] Избыточность.*

Довольно спорный недостаток, но я часто вижу вопросы об избыточности, поэтому прокомментирую этот момент со своей точки зрения.

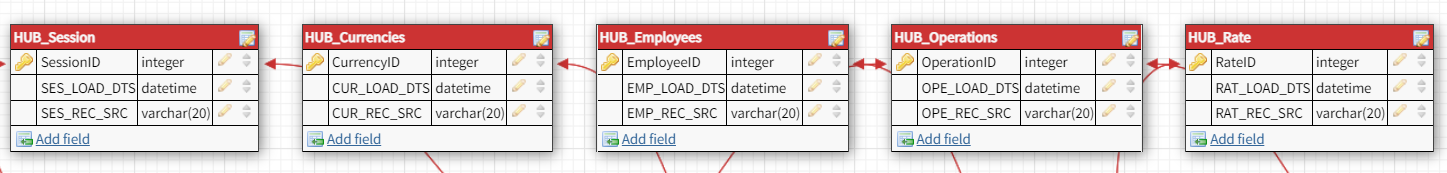
Многим не нравится идея создания прослойки перед витринами данных, особенно если учесть, что таблиц в этой прослойке примерно в 3 раза больше, чем могло бы быть в третьей нормальной форме, а значит в 3 раза больше ETL-процессов. Это так, но и сами ETL процессы будут значительно проще за счет своего однообразия, а все объекты в хранилище достаточно просты для понимания.

# **Создание БД в соответствии с моделью Data Vault.**

Сущности-концентраторы, или просто хабы (hubs), являются таблицей, которая содержит минимальный список бизнес-ключей (натуральных ключей).

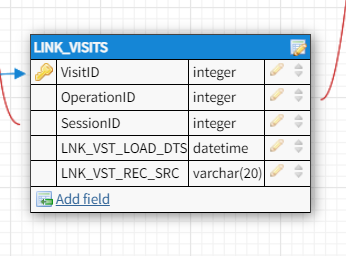
Могут иметь следующие атрибуты:

* суррогатный ключ – опциональный атрибут, который является обычно членом числовой последовательности.
* временная метка загрузки (Load Data/Time Stamp) – это дата и время, когда ключ впервые появился в БД.
* источник данных (Record Source) – записывается для трассировки данных.

  
Рисунок 11 – хабы БД обменного пункта

Связывающая сущность, или сущность-связь (Link Entitiy). Сущности-связи являются физическим представлением взаимосвязи "многие ко многим" в 3NF. Связь представляет собой взаимоотношение или операцию между двумя или более бизнес-компонентами или бизнес-ключами. Сущности-связи содержат следующие атрибуты:

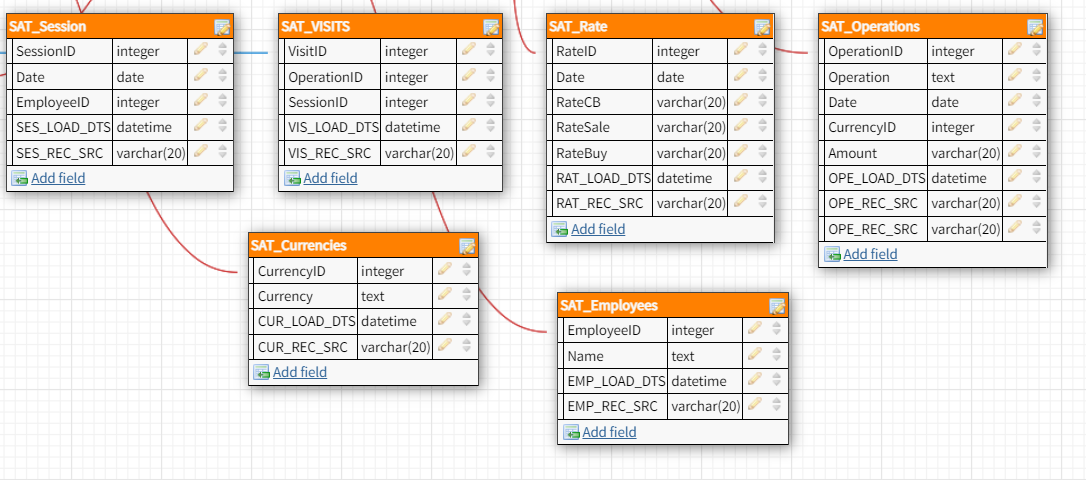
* суррогатный ключ – опциональный атрибут, который используется при связывании более двух концентраторов
* ключи концентраторов (Hub Key) – ключи концентраторов, которые мигрируют в сущность-связь для формирования составного ключа, связывающего эти концентраторы;
* временная метка загрузки (Load Data/Time Stamp) – дата и время записи связи в БД;
* источник данных (Record Source) – используется для трассировки данных.

Рисунок 12 – ссылки БД обменного пункта

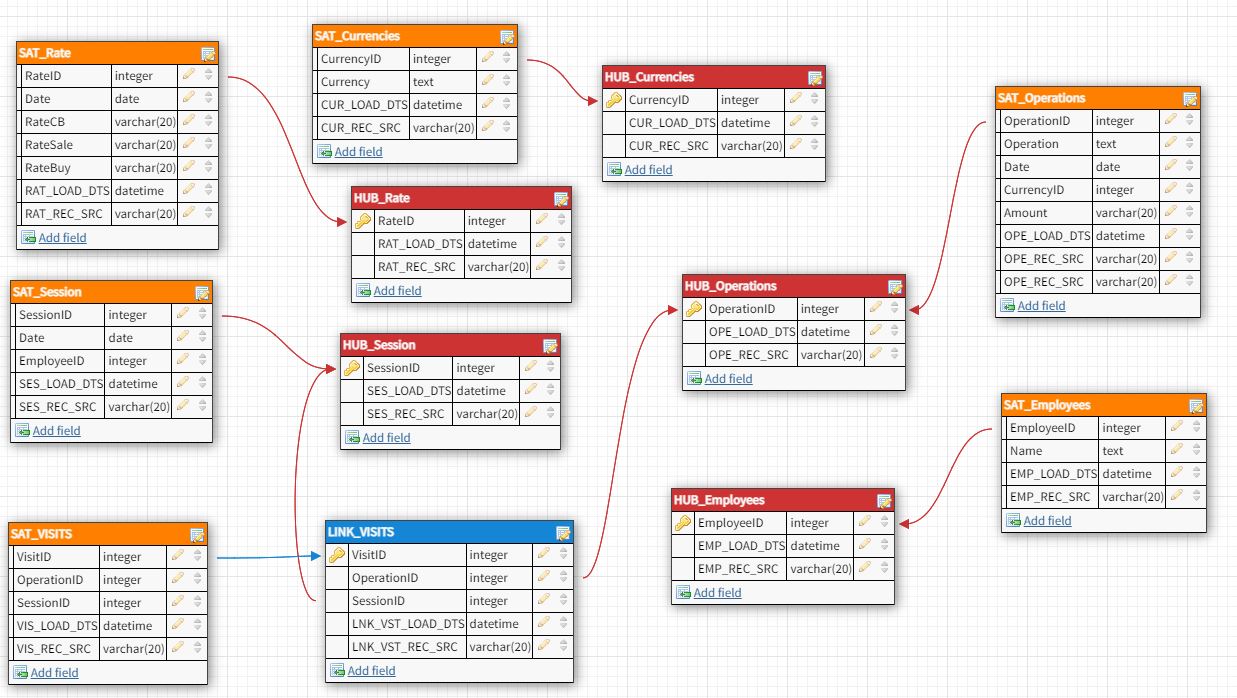
Сущности-сателлиты содержат описательную информацию о ключах концентраторов, а именно когда, почему, что, где и кто создает операции и бизнес-ключи. Например, в отличие от номера автомобиля, его цвет, марка и т.д. могут изменяться во времени, и, следовательно, структура данных должна отражать эти изменения на каждом уровне структурирования информации (гранулированности).

Сущности-сателлиты обычно содержат следующие атрибуты:

* первичный ключ — первичный ключ концентратора или связи, который мигрировал в сателлит;
* временная метка загрузки (Load Data/Time Stamp) – это дата и время записи описательной информации в БД;
* источник данных (Record Source) – записывается для трассировки данных;
* сущности-сателлиту может быть назначен суррогатный первичный ключ.

Рисунок 13 – сателлиты БД обменного пункта

В итоге БД по методологии Data Vault выглядит так:

Рисунок 14 - БД по методологии Data Vault

# **ЗАКЛЮЧЕНИЕ**

Data Vault™ – новый этап эволюции моделирования данных для хранилищ данных масштаба предприятия. Целевая аудитория этой статьи: проектировщики данных, желающие построить модель Data Vault, или специалисты в области хранилищ данных и BI, интересующиеся запросами к Data Vault.

Можно сделать вывод о растущей популярности этого метода построения базы данных.

В процессе выполнения работы мы изучили хранилище данных Data Vault и реализовали структуру базы данных в соответствии с данной моделью.

Все скриншоты выполнения работы предоставлены выше.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Введение в Data Vault [Электронный ресурс]: - режим доступа: <https://habr.com/ru/post/348188/>

# **ПРИЛОЖЕНИЕ А**