11.2 Традиционная архитектура хранилищ данных

Тексовая расшифровка видео:

LAMBDA И KAPPA

План:

- Баланс сложности и простоты;
- Общая схема;
- Lambda-архитектура;
- Kappa-архитектура;
- Почитать.

Баланс сложности и простоты

Мы уже знаем о понятиях «Batching» и «Streaming».

Поймем, что означает «Batching» на примере:

У нас может быть какое-то хранилище (Data Lake, Data Warehouse), куда мы оправляем данные. Это хранилище выступает как Single Source of Truth, то есть главным источником правды. По сути, мы можем делать достаточно сложные вещи, но медленно, так как данных много.

result = query(all data)

Девиз: «Делаем сложные вещи медленно»

Поймем, что означает «Streaming» на примере:

Данные приходят по записям или мини-батчами. Мы смотрим на конечный кусочек и быстро по нему принимаем решения. Здесь мы считаем достаточно приблизительно и просто, но при этом быстро.

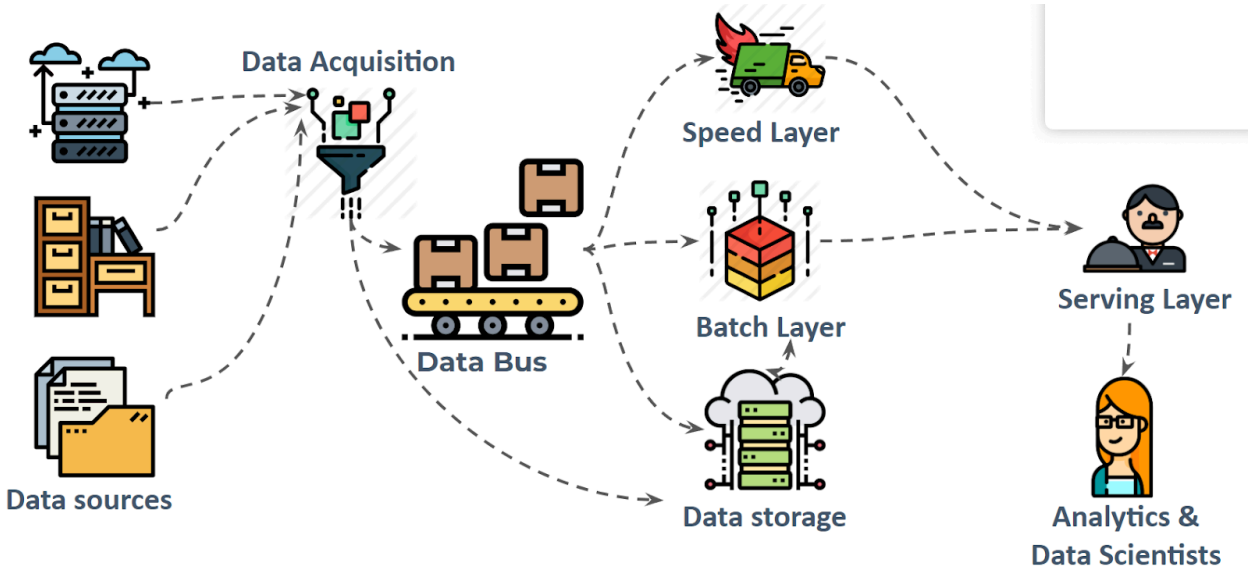
result = aggregation(small chunk of data)

Девиз: «Делаем простые вещи быстро»

В идеале хотелось бы объединить все положительные стороны подходов и построить архитектуру, в которой сможем процессить оба вида данных.

Общая схема

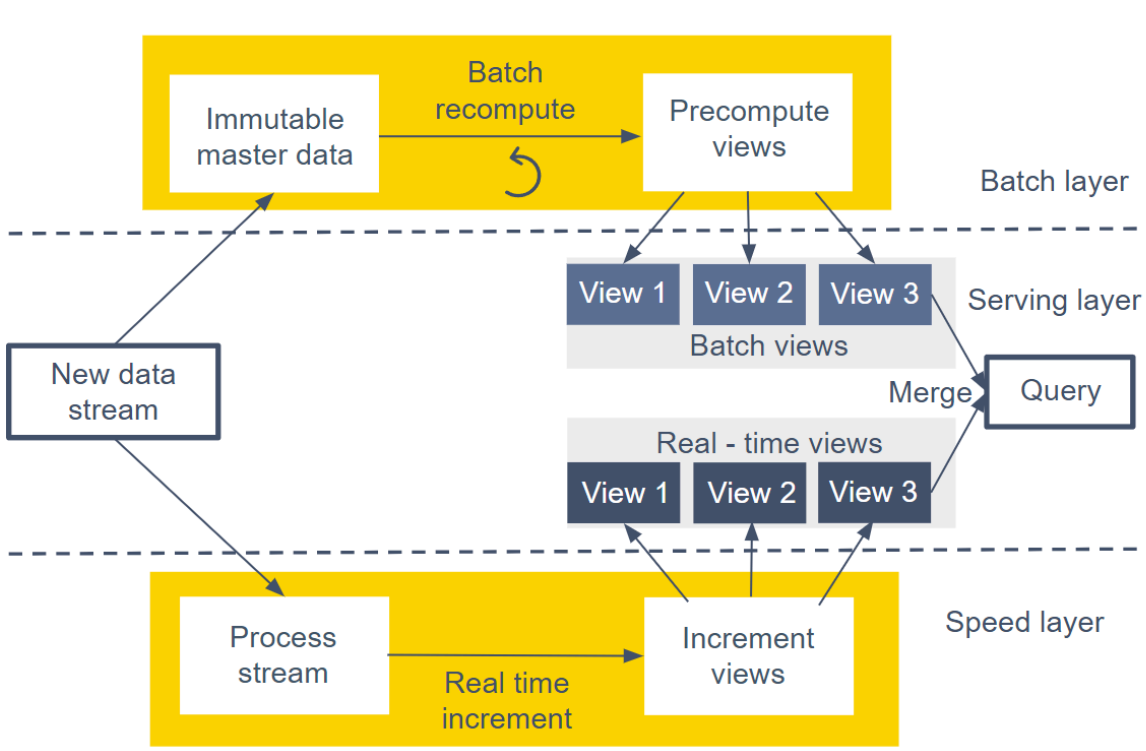
Если попытаться представить подобную схему, то выглядеть она будет примерно так:



- Слева находятся источники данных;
- Есть процесс (Data acquisition), то есть какой-то ETL, который мы используем, чтобы забирать данные из источников;
- Эти данные пишем напрямую в хранилище или в шину (например, в Kafka);
- Поверх хранилища/шины возникает два слоя – Speed Layer (скоростной слой или же слой Real time), Batch Layer (пакетный слой или же хранилище, где мы можем долго, но очень точно считать разные вещи). Два слоя существуют одновременно. Это два контура, по которым, в зависимости от типа данных, мы можем их отправить. Помимо этого, эти два слоя друг друга подстраховывают в случае падения.
- Результаты, посчитанные на каждом из этих слоев, хранятся в какой-то базе/представлении, чтобы мы могли делать поверх этого дашборд. Дашборд может быть Real time; также дашборд может периодически пересчитываться. На этом слое перед нами уже предстают витрины. Потребителями являются бизнес-люди, аналитики, дата-сайентисты.

Lambda-архитектура

Если схему выше перерисовать в «сухую» схему, то получится следующее:



Если погуглить о том, что такое «Lambda-архитектура», то вы наткнетесь на похожую схему, где есть:

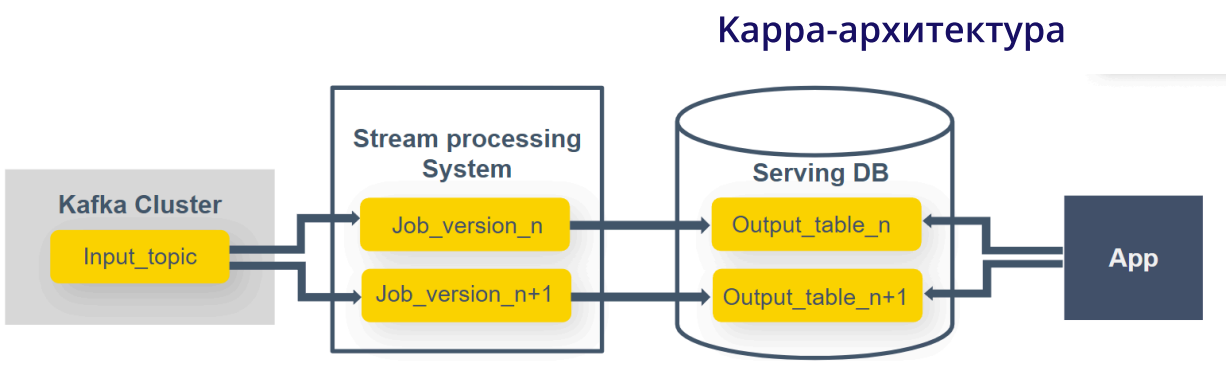
- Три слоя (по факту – 2, но мы считаем вместе с Serving Layer)

- Batch Layer, который периодически что-то пересчитывает и генерирует представление витрины, которую потребляют пользователи.
- Speed Layer, где мы в Real time смотрим на новые кусочки данных, быстро пересчитываем метрики. Все это с помощью Serving Layer или какой-либо базы агрегируем и отдаем потребителям. Здесь могут прийти дата-абстракции, например, prescott и т.д.

Если мы хотим страховать один слой другим, могут возникнуть неочевидные проблемы, например, дублироваться логика.

Помимо этого, возникает вопрос: «на каком языке писать?». Тем не менее, большое количество компаний и команд используют подобие этой Lambda-архитектуры.

Чтобы отрисовать проблемы, которые в ней присутствуют, придумали следующий шаг эволюции Lambda-архитектуры.

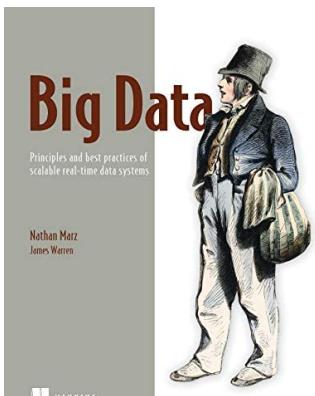


Основная идея: если мы работаем с данными (например, с clickstream), в которых нет смысла хранить эти данные исторически, и, которые мы можем обрабатывать налету и строить поверх них витрины, то нет нужны добавлять большой Batch Layer. В итоге у нас будет Kafka (или другая шина), в которой хранятся данные за какой-то период. Поверх Kafka запущены Streaming Job'ы, которые то-либо считают. Результат расчета – витрины, строящиеся в Serving DB.

Подобная система проще, чем Lambda-архитектура, но в ней мы не сможем ответить на вопрос: «что произошло 3 года назад?».

Почитать

Все эти концепции ввел Nathan Marz в данной книге:



Сама книга немного устарела, но если есть желание лучше понять разделение Batch Layer, Speed Layer, Serving Layer, то рекомендуем к чтению.

Данная книга позволит ознакомиться с вариантами построения Lambda-архитектуры с конкретными примерами технологий (книга скорее всего только на английском языке):