Data Model on Building Enterprise Decision Support System

Introduction:

I would like to provide a quick summary of experience gained over at different stages of my career as a data designer and modeller. Let me start with the recent project experience and compare it with other design strategies followed on various stages of the project.

Data Model Layers:

Every stage of a data project should be designed to satisfy the data requirement of current and next lower layer for which is dependent on. Eg: Whenever we build an end to end data warehouse model, the model to be associated with the data ingestion layer will be mostly different from the model associated with the serving/reporting layer. A data ingestion layer will be built for every business use case. For example: a data topic for every type of adjustments, a data topic for each industry type adjustment etc., (Topic on Kafka corresponds to Table on RDBMS). Basically, this will help us to determine what are the data items required for the business use case and reach out to appropriate data SME to leverage for the data platform. Since this serves as raw data layer, no manipulation or relationship constrains will be imposed. But most of the time spend will be with identifying relevant data models behind every source system and identify to reject if not required for not overloading the source layer. One more important aspect is : It will always be in append mode, time series will be maintained, schema enforcement will be applied.

Next layer involves with the design of data model to the middle layer(Single Source of Truth) which we call as “Enterprise Datawarehouse layer”. The current project has adopted the traditional Kimball based Dimensional modelling following both Star Schema and Snowflake Schema approach. Star schema is the most adopted one with the occasional support of wider dimension tables for snowflake. Various database principles will be enforced by having constraints like Foreign key references, Primary Key Constraint, Identity, Change Tracking, Table hints, reduce redundant data(keep normalised as possible) etc.,. In terms of storage, column store indexes will be preferred wherever applicable(fact tables) , tables will be partitioned based on time series, batch and real time loading using Lambda architecture, auditing columns will be added at every stage, will also be incorporating logging and traceability matrix at every loading phased. The design pattern will be data driven rather than end-user query driven. However, one of the major hurdle will be on providing an integrated data model which could be scalable and dynamic enough to accept integration of any future data sources. This made me to compare my experience with data vault modelling proposed by Dan Linstedt using Hub/Satellite/Link tables. Hub for holding business keys and satellites for description about business keys and link table representing relationships between two or more business keys. Though it sounds pretty good on handling schema evolution, there are quite few things to be considered at storage, read performance, many convoluted joins ,derivation of logical model from the business model and quite hard for end-user querying or analysing etc.,

Though the above two layers are pillars or foundational layer for a data platform, the business data processing layer which is the third layer is the ultimate interface for the Business, Data Scientists and Business SME’s who are interested. These could result in a layer of data model which can be different from the above two data models. One of the primary focus is keep the models query driven, be able to support high read intensive operations and aggressive queries. Most of these

cube models involve aggregated results, time series and mostly de normalised. Since this layer involves writing business measures like aggregation over time periods, ratio related measures, triggering metrics, Discrete value measures etc., it is mostly recommended to get the domain specific knowledge to substantiate for various data reporting and business self serving needs. Here, most of the infrastructure needs, will be focused on compute and memory intensive machines rather than the storage intensive. And also consider using high processing and distributed framework like Spark for faster and heavy volume queries.

One of the other aspect to be taken into consideration while handling Unstructured or Semi Structured data. Cassandra or other No-SQL database can be considered for this option but again its data model needs to be tailored according to the architecture of its processing engine and its storage engine.

With that, I would like to depict a very simplified version of end to end data pipeline which would be adopted both for batch processing and real time processing.

Simplified version of data flow diagram: