***One or More DBs?***

**Issue** The system needs to handle data from multiple sources and different types of usage, such as data-mining and statistics. Using specialized databases may result in better performance and scalability.

**Decision** The difference in usage and certain data being unrelated to each other makes using more than one database the best choice.

**Status** Decided

**Group** Data Storage

**Assumptions** Data mining and statistics will receive a lot of writes whereas the website will be balanced between read and write.

**Constraints** None

**Positions** A single database:  
+ Only 1 database to manage.  
- Unable to use specialized databases because it needs to be able to handle multiple situations.  
- If you need to scale for one part of the DB usage, you need to scale the whole database.  
- All data is grouped together no matter how diverse the data may be.

**Argument** + You can use specialized databases; get the most performance for the given type of usage.  
+ You only need to scale the database that requires it.  
+ Data not related to each other is kept separate.  
- You need to maintain more than one database.

**Implications** Raises the questions; How many databases (depends on what we want to separate) and what type of database?

**Related** **decisions**

**Related** **requirements** Performance, Scalability

**Related** **artifacts**

**Related** **principles**

**Notes**

***How many databases?***

**Issue** How many databases gives us a good maintainable system and allows us to get the most performance for all the different database usages?

**Decision**

**Status** Pending

**Group** Data Storage

**Assumptions** At the very least we have website data, data-mining data and statistical data.

**Constraints** None

**Positions** Option 1; Data-mining and Website databases:  
+ You can have both a specialized database for data-mining and website data, increasing performance of either task.  
+ You only have to scale the database that really needs scaling.  
~ Only 2 databases to maintain.  
Option 2; Data-mining, website and statistical databases:  
- Data gets to fragmented and hard to use/access.  
- Managing databases becomes unclear.  
~ Linking data has to be handled in the application layer

**Argument**

**Implications** What types of databases do we use?

**Related** **decisions** One or More DBs?

**Related** **requirements** Maintainability, Performance

**Related** **artifacts**

**Related** **principles**

**Notes**

***Type of Database for data-mining***

**Issue**

**Decision**

**Status**

**Group**

**Assumptions**

**Constraints**

**Positions**

**Argument**

**Implications**

**Related** **decisions**

**Related** **requirements**

**Related** **artifacts**

**Related** **principles**

**Notes**

***Type of Database for Statistics***

**Issue**

**Decision**

**Status**

**Group**

**Assumptions**

**Constraints**

**Positions**

**Argument**

**Implications**

**Related** **decisions**

**Related** **requirements**

**Related** **artifacts**

**Related** **principles**

**Notes**

***Type of Database for general storage***

**Issue**

**Decision**

**Status**

**Group**

**Assumptions**

**Constraints**

**Positions**

**Argument**

**Implications**

**Related** **decisions**

**Related** **requirements**

**Related** **artifacts**

**Related** **principles**

**Notes**

***How do we handle large data-sets?***

**Issue** Data collected from external sources requires a great amount of storage over time. It also needs to be analyzed and have statistics generated/stored.

**Decision** Event type of dispatching. By collecting the data and sending it to the analyzers, we can generate the rating and score before storing it all. This way we don’t need to read and write twice from intermediate storage which improves performance.

**Status** Pending

**Group** Data storage

**Assumptions** x tweets, x facebook posts, x reviews

**Constraints** None

**Positions** Pipeline; mine and store in a database, then read this database out and analyze the data:  
- Lots of read and writes on the database used during mining.  
- System has to actively check for new data in the database.  
- Harder to scale properly due to the heavy reliance on this database of several modules.

**Argument** + When stored, it is in its final format.  
+ Passive analyzing; only performs work when it needs to.  
+ Easy to queue data analysis to spread load and thus increasing scalability.

**Implications** Data in queue is lost with a crash.

**Related** **decisions**

**Related** **requirements** Performance, Scalability

**Related** **artifacts**

**Related** **principles**

**Notes**

***Recalculate or combine rating?***

**Issue** When receiving new information from twitter, facebook, tripadvisor or our own website, the rating for the airline company needs to be updated.  
This can be done by recalculating every review/tweet/etc or by using the amount of reviews/tweets/etc and the current rating.

**Decision** Utilize the amount of reviews/tweets/etc that make up the current rating and use this number to balance the weight of the value to be added to the rating.

**Status** Pending

**Group** Data Analysis

**Assumptions** We keep track of the number of reviews/tweets/etc and don’t need to count rows (otherwise there isn’t much of a performance gain).

**Constraints** None

**Positions** Recalculate:  
- Traversing the whole database will take more and more time over time.  
+ Allows you to update the weight of previous entries at every update.

**Argument** + No need to traverse everything in the database to calculate a rating.  
+ A recalculation can always be done if it is really necessary.

**Implications** If the weight of a review/tweet/etc changes, a recalculation has to be initiated by the db/system manager.

**Related** **decisions**

**Related** **requirements** Performance

**Related** **artifacts**

**Related** **principles**

**Notes**

***Incremental or Time-interval updates?***

**Issue** Updating the stats and rating with data from external sources can be done when receiving new data or on a time interval.

**Decision** Updating when receiving data, utilizing a queue to spread load.

**Status** Pending

**Group** Data Analysis

**Assumptions**

**Constraints** None

**Positions** Time-Interval:  
- Can create high peaks of load and lots of idling; variable resource usage. This means you need a server that can handle the peak when it occurs which means you’re wasting resources.

**Argument** + Load is spread out; you need less resources as no peaks will occur.

**Implications**

**Related** **decisions**

**Related** **requirements** Performance, Scalability

**Related** **artifacts**

**Related** **principles**

**Notes**

***Flight-data usage***

**Issue** The stakeholders had the interest of using flight data to increase or decrease the weight of reviews. For instance if a flight is delayed due to bad weather, reviews should weigh less because it is out of the hands of the airline company. However, what is the responsibility of the airline companies is the service of handling such a situation. Because of this the review should not weigh less.

**Decision** Don’t use flight data for weighing reviews. At most, only use it when displaying statistics so you can see if delays have influenced ratings yes or no.

**Status** To discuss with stakeholders

**Group** Data Integrity, Data Analysis

**Assumptions**

**Constraints** None

**Positions** Adapt weight with flight data:  
- Review integrity cannot be guaranteed; you’re lowering weight where it probably should not.  
- You have to pull in a lot of data when updating the ratings thus increasing the load on the system.

**Argument** + Review rating fairness is guaranteed.  
+ Updating the rating is faster.

**Implications** Ratings are more trustworthy for the user.

**Related** **decisions**

**Related** **requirements** Integrity, Performance

**Related** **artifacts**

**Related** **principles**

**Notes**

***Title***

**Issue**

**Decision**

**Status**

**Group**

**Assumptions**

**Constraints**

**Positions**

**Argument**

**Implications**

**Related** **decisions**

**Related** **requirements**

**Related** **artifacts**

**Related** **principles**

**Notes**