Prepared for:

by:

Smarter Balanced Reporting
Data Warehouse and Reporting
Technical Architecture

Approvals

Representing	Date	Author	Status
Consortium		Joe Willhoft	
Consortium	2014.06.09	Brandt Redd	Submitted for Review
PMP		Kevin King	
Workgroup		Henry King	

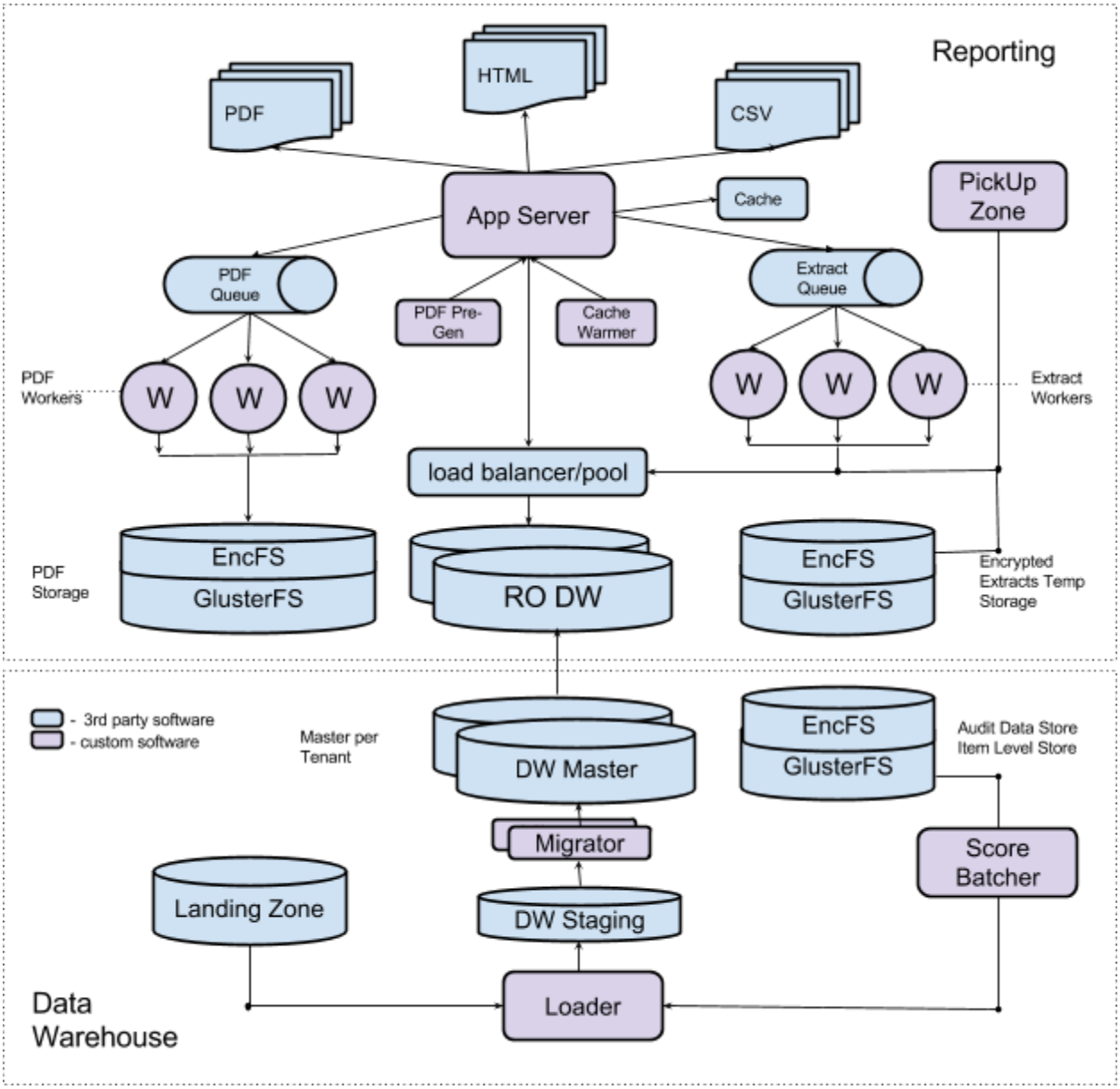
Revision History

Revision Description	Author/Modifier	Date
Initial Release (DRAFT)	Anna Grebneva (Amplify)	2014.06.03
Updated for ScoreBatcher and PickUp Zone	Anna Grebneva (Amplify)	2014.08.19

Table of Contents

Smarter Balanced Data Warehouse and Reporting offer a secure and scalable multi-tenant system that houses student assessment and student registration data and provides tools for data access in HTML, CSV, and PDF formats.

System Architecture



Reporting Layer - a web based application and supporting system that end users interact with. User roles define access level to the application and user context defines a scope of student PII the user is exposed to.

Data Warehouse - a student assessment and registration multi-tenant data store and a loader process that is responsible for processing new data.

Reporting Layer

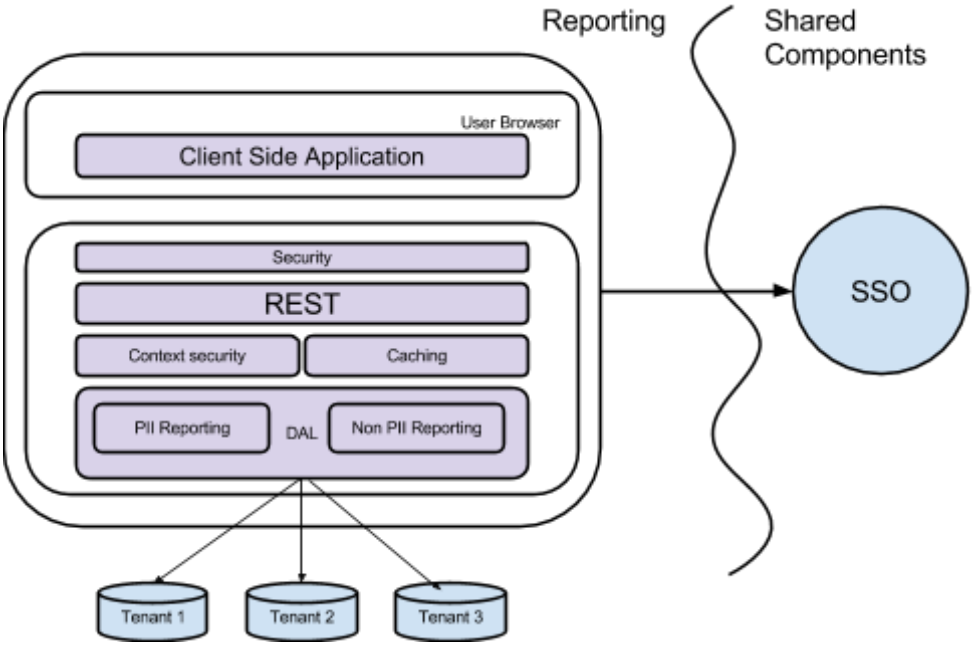
Reporting layer consists of the following components:

Application Server

Application server is a stateless web application responsible for end-user interactions and security. It consists of the service side application, which provides a secure RESTful interface to query the data warehouse, and a client side application, which uses the interface to provide visualizations. The client side application is html/javascript and runs in a user's web browser.

The server delegates tasks, like PDF and CSV extract generation, to the appropriate downstream processes. It also relies on a caching layer for non PII data.

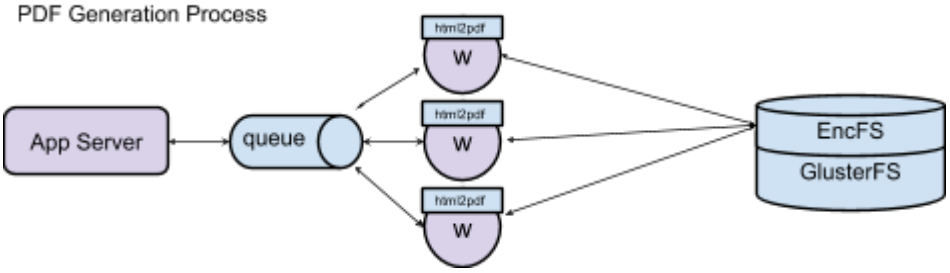
The application server includes security modules for user access policies and data access tenant routing.



The stateless nature of the application allows it to scale horizontally by adding more servers to the cluster.

PDF Generator

PDF generator is a service that produces PDF versions of reports, such as the Individual Student Report. An open source (LGPL) tool, wkhtmltopdf, is used to convert HTML reports to PDF formats. PDF generation is a resource-intensive process. In order to accommodate a large number of PDFs and concurrent users, the service is distributed and scales horizontally by adding new worker nodes. The generated PDFs are cached on an encrypted file system for subsequent requests to facilitate re-use.
The PDF process is distributed across multiple servers. Within each server, multiple workers can pick up a PDF request and return an existing document, if available, or use wkhtmltopdf tool to generate a new document.



PDF Pre-Generator

PDF pre-generator is a standalone application that triggers PDF generation tasks when new data is loaded into the data warehouse. Pre-generation is useful to reduce the response time on user PDF requests. The Pre-Generator can be run on a schedule to avoid sudden usage spikes. Each install should contain only one instance of PDF Pre-Generator.

CSV Extract Generator

Extract Generator is a task-based application similar to the PDF Generator. It is used to create bulk raw data files in csv format. However, unlike PDFs, csv extract requests are unique per user, therefore, none of the extracted files are pre-generated to be cached.

CSV Extract Generator is designed to scale horizontally by adding new servers that will pick up extract tasks from the queue. The system supports two retrieval types of extracts: synchronous and asynchronous. Synchronous extracts are returned to the user's browser immediately, whereas asynchronous extracts, which often take a longer period of time to generate, are placed in a secure pick-up zone for later retrieval.

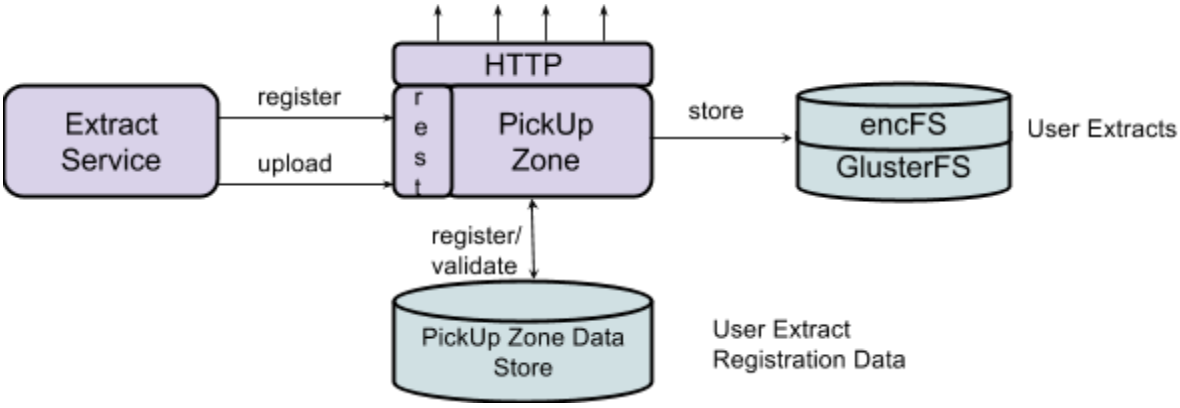
Cache Warmer

The Cache Warmer re-populates the cache used by the Application Server for results of popular data requests. Cache Warmer is triggered on a schedule, and it is mainly designed to re-

populate data for the Comparing Districts and Comparing Schools Reports, which are resource-intensive aggregations of data. The data requests are configurable for the types of demographics filters to apply to the data requests. Each install should contain only one instance of Cache Warmer.

PickUp Zone

PickUp Zone is a secure user based temporary file storage area that provides an HTTP interface. The system provides an internal RESTful API for the system to register user extracts to be exposed to the user with a URL. PickUp Zone integrates with SSO and only gives access to URLs registered to a specific user.



Read-Only Data Stores

Data stores are physically separated by tenant (state). Each tenant cluster contains a number of read-only data stores scaled to serve a projected number of users. These data stores are load-balanced and connections are pooled using third-party database middleware.

Data Warehouse

The Data Warehouse consists of the following components:

Landing Zone

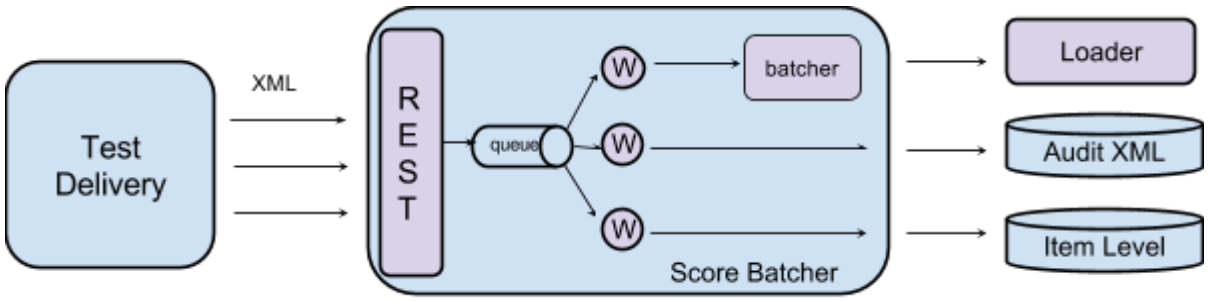
Landing Zone is a tenant-based drop-off zone for incoming batched data files. The landing zone is not designed for plain-text PII data and requires data files to be encrypted and compressed. The landing zone format is described in a separate document.

Score Batcher

Batching service is designed to receive individual assessment results that are produced near-real time by the Test Delivery System, and gather them for batch processing. The batching

service provides a RESTful interface to collect individual responses. After receiving data, Score Batcher queues it to be picked up by an available worker for parsing and conversion. The Score Batcher is responsible for Item Level Data, Audit XML, and assessment data in the Landing Zone format. The assessment data is collected in a batch data file, which is then forwarded to the Loader for ETL.

The score batcher is designed to scale horizontally by adding http servers and workers to handle TDS traffic.

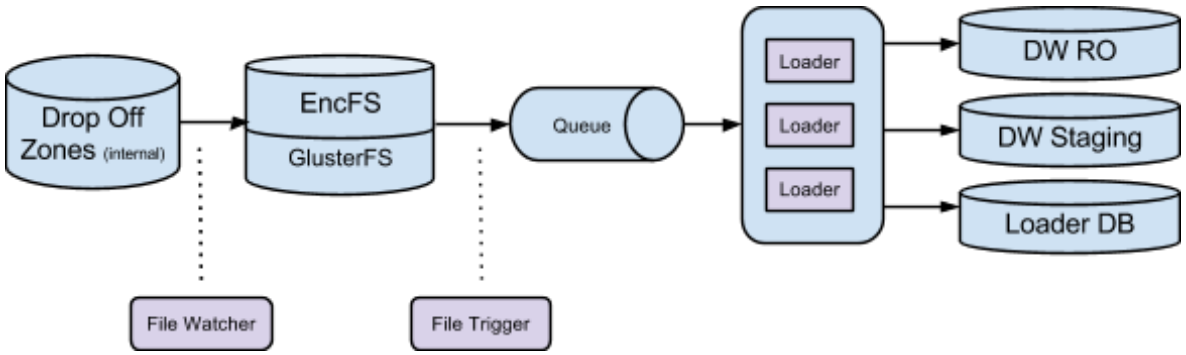


Score Batcher response codes:
202 Accepted - The payload was received and stored locally. No proper validation has been done and it is not an indication of a successful record ingestion.
401 Unauthorized - The requesting service has not been authenticated.
403 Forbidden - The requesting services has been authenticated but does not have proper rights.
412 Precondition failed - The payload failed basic validation, such as xsd validation.

Loader

The data loader is a distributed system that processes incoming data files and loads data to a staging database. A File Watcher watches for new files being dropped into the Landing Zone. Once the watcher detects a new, complete file in the Landing Zone, it transfers the file to an encrypted storage, where the Loader will decrypt the file for processing. Once decrypted, the Loader will then process the file and insert the data into the Data Warehouse staging database, where it will wait to be migrated into production by the Migrator.

The Loader is designed to scale horizontally by adding additional servers that are able to pick up tasks from the queue (see diagram). By scaling horizontally, the system will be able to process more batched data files concurrently.



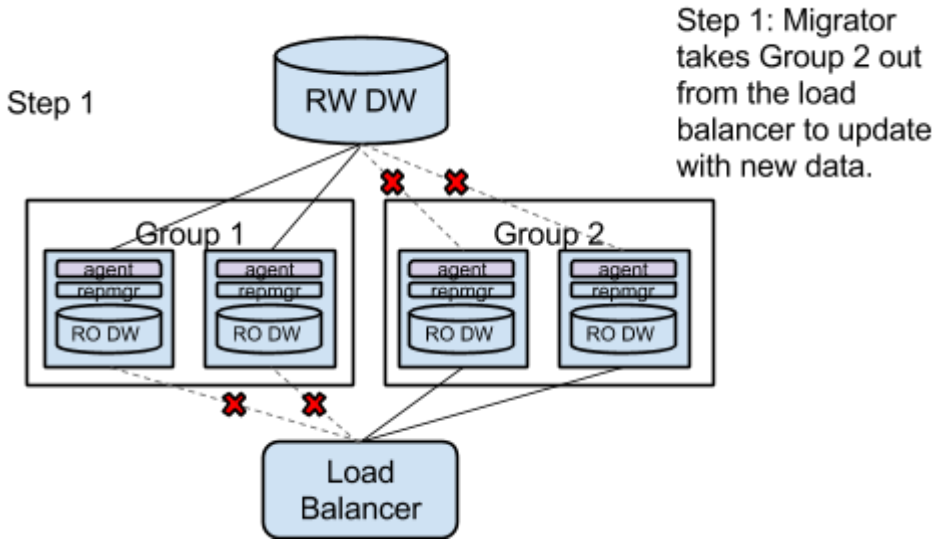
DW Staging

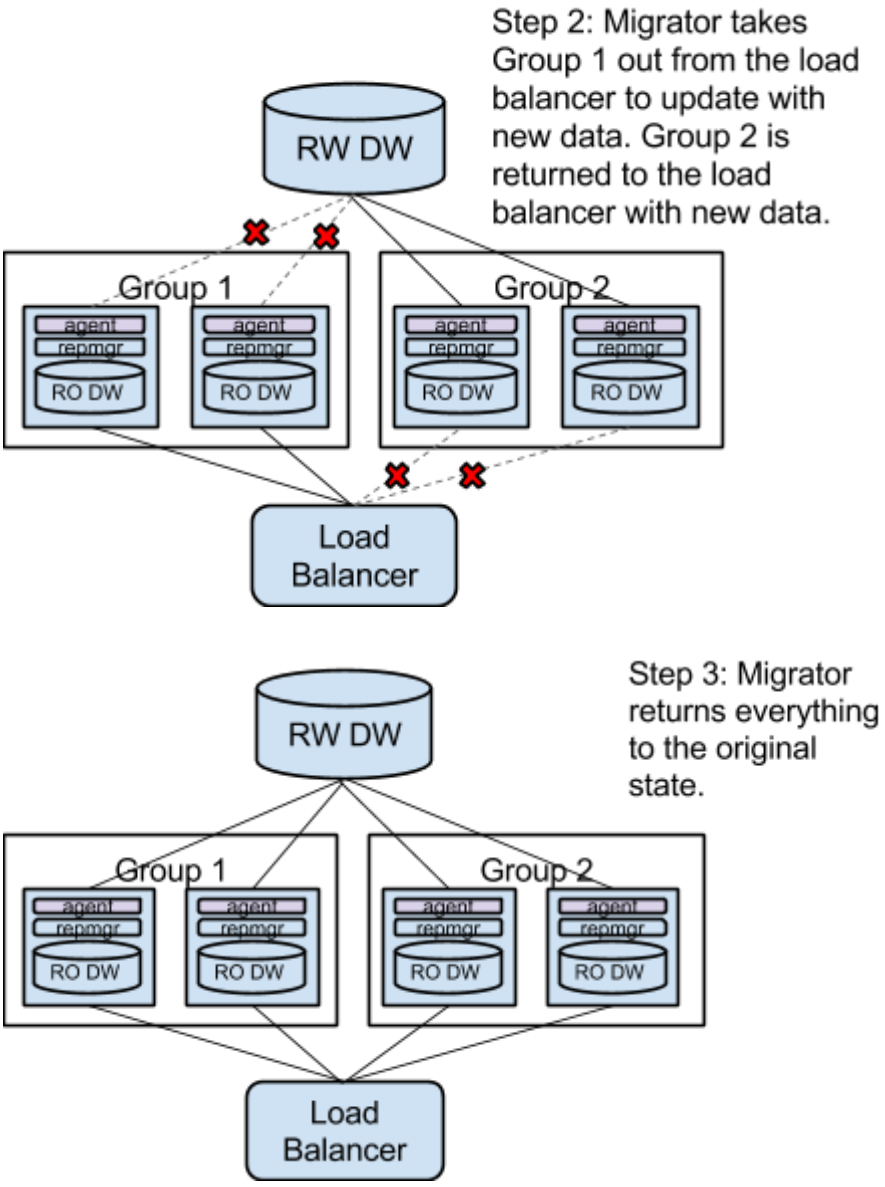
The staging zone contains newly-imported data that is ready to be migrated to production. The staging data store contains data deltas only (i.e only the data that has changed since the last migration). The DW Staging database is configured per tenant.

Migrator

Migrator is an application responsible for moving data from staging to production, while ensuring minimal impact for the reporting users. The migration process will orchestrate replication and load balancing of production database slaves so only half of them are down during the migration process and end-users will be able to continue accessing reports.

The steps of the migration process are outlined below:





Read/Write Data Stores

Data stores are physically separated by tenant (state). The RW data stores are not used by the reporting application directly. The reporting application reads from the Read-Only data stores.

Item Level Data Store

Item level data store contains a subset of individual student item level responses.

Audit XML Data Store

Audit data store contains original XML posted by Test Delivery Service.

Capacity Planning

The number of instances required for each machine type is correlated to three independent criterias: the number of tenants/states in the system, the number of users accessing the system, and the number of students in the tenant/state.

Table 1 - Sample configuration for data warehouse and reporting for 1 tenant containing 2M students and 100K users.

Machine	Avg Usage (3% user concurrency) instances	Peak Usage (10% user concurrency) instances	Cores (vCPU per instance)	Memory (GB per instance)	non-OS portion Storage (GB)
web servers[2]	9	30	4	8	
load balancers[2]	2	2	2	2	
database masters[1]	1	1	8	64	1000
database replicas for web front end[2]	2	7	8	64	1000
database replicas for extraction[2]	1	3	8	64	1000
db load balancer[1]	2	1	8	64	
db pool[2]	2	3	4	8	
cache[3]	2	2	8	64	
pdf	2	2	4	8	

messenger[2]					
pdf workers[3]	2	2	8	64	
pdf generator[1]	1	1	2	8	
extract messaging[2]	2	2	4	8	
extract worker[2]	2	7	8	32	
cache warmer[1]	1	1	2	8	
landing zone	2	2	4	8	50
pickup zone	3	9	4	8	
pickup zone database	1	1	4	8	10
loader[3]	2	2	4	8	
loader messenger	2	2	4	8	
loader database	1	1	8	32	1000
migrator[1]	1	1	4	32	
score batcher web[3]	2	5	4	8	
score batcher messenger	2	2	4	8	
score batcher worker [3]	2	7	8	32	
score batcher trigger	1	1	4	32	
database staging[1]	1	1	8	32	1000
gluster					26480

(storage)[3]					

[1] The number of instances for this machine type is proportional to every one Tenant/State.
[2] The number of instances for this machine type is proportional to every 100,000 Users.
[3] The number of instances for this machine type is proportional to every 2,000,000 Students.
[4] The number of storage for this is proportional to 10 years of item level data storage plus 1 years of pdf.
[5] The number of storage for this is proportional to proportional to every 100,000 Users.