# DESIGN CONCEPTION OF INFORMATION DASHBOARDS FOR MASSIVE DATA VISUALIZATION

**ALABI, I. O.[1]; ETUK, S. O.[2]; AMINU, F. E.[3]; & EKUNDAYO, A.[4]**
[1,2] Department of Information Technology, Federal University of Technology, Minna, Nigeria.
[3,4] Department of Computer Science, Federal University of Technology, Minna, Nigeria.
**E-mail:** isiaq.alabi@futminna.edu.ng **Phone No:** +234-816-947-4875

## Abstract

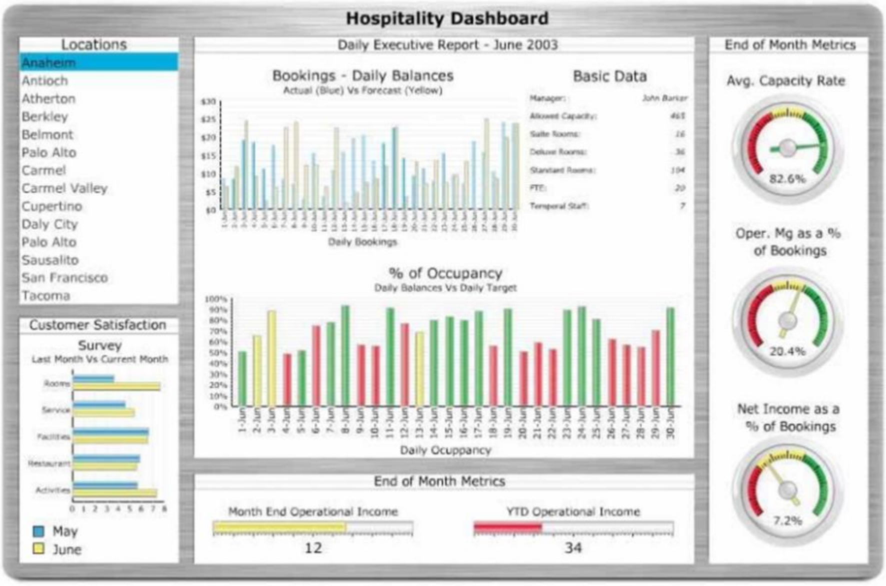
*Dashboards as a data visualization tool, are business reporting tools that aggregate all data in a single screenshot. Dashboards are links to data files, application program interfaces (APIs) and metadata to provide users major metrics and key performance indicators (KPI) about systems' processes or a business entity. Hence, Dashboards extract, aggregate highlight and communicate high-level information to infer anomalies, prospects, issues and trends. This study proposed Wander join technique to provide data aggregation for fast interactive queries for data visualization with minimal latency. The importance sampling approach of the Wander join algorithm was used to support joins for data convergence and augment latency in common visualization queries. This enabled a uniform convergence rate for all displayed data aggregation categories. 30,000 samples out of 120 million records of flights arrival and departure at an airport were drawn with a convergence of 0.05 relative error with near-zero latency as the aggregation occurs. With adjustable weights, the Wander Join algorithm allows groups of interest to be sampled often, while groups of less interest can be weighted to be sampled less often if the weight is set to 0.*

**Keywords:** Information dashboard, data visualization, massive data visualization, online aggregation. data convergence with minimal delay for visualization

## Introduction

Taming and distilling information from a deluge of a data source is a primary goal of the information technology (IT) industry. Visualization of information from massive data is feasible especially through many media and interfaces such as dashboards. A dashboard is an information media whereby the most important information required to make critical business decisions or to accomplish a specific task is presented at a glance and within a screen display. A dashboard is usually effective when it is properly designed in a manner that communicates clearly and instantly. An effective visual display requires an understanding of visual perception about design principles and practices that are aligned with the way people see and think (Battle, Chang & Stonebraker, 2016).

Dashboards are a unique interface for an organization's need for instant visual information to guide in decision making and to depict overall organizational management. A business information or business intelligence (BI) dashboard provides an at-a-glance view of major business indicators, suggesting problems or areas requiring urgent attention. Business Intelligence (BI) dashboards focus on exploring, reporting and analysis of structured data to discern business-specific trends, see Figure 1. Importantly, dashboards provide an overview of information about processes in a system or a business environment.



| | |
|---|---|
| Manager: | John Barker |
| Allowed Capacity: | 465 |
| Suite Rooms: | 16 |
| Deluxe Rooms: | 36 |
| Standard Rooms: | 104 |
| FTE: | 20 |
| Temporal Staff: | 7 |

**Fig. 1: A dashboard of a hotel chain measures of performance.**
**Source:** Few, 2006

The dashboard in Figure 1 provides managers of a hotel chain the instant view of multiple performance measures in the hotel, which can be viewed one at a time for each of the hotels in the conglomerate. The dashboard provides a List-box in the top-left corner to enable users to select a particular hotel by location.

A major challenge of dashboard design is the need to squeeze a great deal of information into a small amount of screen space. In the process of doing this, latency often occurs due to query selection, aggregation and display on dashboards. The objective of this study is to provide a data aggregation technique for fast interactive queries for data visualization with minimal latency. Due to latency in the display of most visualization techniques, this study reviewed some data visualization techniques from recent works of literature. Techniques such as data cube, random sampling, online aggregation, progressive visualization, importance sampling to reduce the number of records that need to be read from across large databases were considered.

The ultimate goal of visual information is communication to the main users of the dashboard while tracking and rapidly processing (or analyzing) the selected key data points. The main data pixels are well projected and enhanced in a condensed catchy form (such as summaries and exceptions) that fits on a single computer screen without loss of meaning. Dashboards are front-end interfaces that connect to multiple large files, often complex databases, metadata, APIs at the back-end (Wu & Nandi, 2015).

Computer-generated images and human vision mediated by the principles of perceptual psychology are the means used in scientific visualization to achieve visualization communication (Ertug *et al.*, 2018). Information dashboards impart on vision as well as the thought process of humans since both are correlated. The understanding of visual perception is therefore applied in dashboards design. For instance, visual and thought processes are applied to bind data together, separate data, or make some data distinct, so that human eyes can focus quickly and the brain can easily comprehend it (Kosara, 2016). Quite often, “dashboard” and “data visualization” are used interchangeably.

Harbig and Gehlenborg (2019) posit that dashboards are useful for all kinds of tasks, examples include information displays for monitoring projects' performance or daily bank transactions; process, storage and disposal of finished products at an industrial complex; products sales and distribution monitoring across regions; monitoring shares capital market movements by financial analysts; demographic and climatic change information; onshore/offshore revenue generation monitoring, and so on. Essentially, the quest for monitoring, control and communication creates the need for information dashboards (Godfrey, Gryz, & Lasek, 2016).

In the 1990s, Data Warehousing (DW), Online Analytical Processing (OLAP), Executive Information System (EIS) or Decision Support System (DSS) and eventually Business Intelligence (BI) were used to mine and project massive data across multiple platforms. The emphasis then was on collecting, correcting, integrating, storing, and accessing information in ways that sought to guarantee its accuracy, timeliness, and usefulness. The essence of data mining could not be realized in the time past due to inadequate data handling technologies required to bridge many disparate information sources. Now, such technological challenges have been overcome (Qin, Luo, Tang, & Li, 2020). Presently, massive data projection and reporting efforts have largely focused on the technologies, needed to navigate through large, often complex databases and metadata (Wu et al., 2017).

Table 1 shows some scenarios or environments where a dashboard might be required, though not exhaustive. The list of such environments is increasing.

**Table 1: Examples of industrial dashboards information display**

| Industry Category | Dashboard Events |
|---|---|
| Airport | Number of flights arrivals<br>Number of flights departure<br>Number of passengers' arrival<br>Flights arrival/departure time<br>Weather information<br>Expected flights arrivals<br>Flights boarding information |
| Sales | Prices/Billings<br>Sales volume<br>Number of orders<br>Number of sales per region<br>Quarterly Sales |
| Technical Support | Number of support issues<br>Number of resolved issues<br>Escalated issues<br>Issues duration<br>Issues that led to downtime |
| Information Technology Infrastructure | Connectivity downtime<br>Duration of downtime<br>System usage / idle capacity<br>Resources allocation |
| Finance | Expenditure<br>Income<br>Profit<br>Sectional expenses/income/profit<br>Growth/decline ratios |

Dashboards are categorized in several ways to express the types of process or business activities it supports such as based on the type of data it projects (quantitative or non-quantitative); data span (enterprise-wide, departmental, regional or individual); role (operational, strategic or analytical); data domain (sales, human resources, weather, manufacturing, etc.); and interactivity (static or interactive display), as well as other categories deemed necessary. Regardless of a dashboard category, the objective is the same — to track, aggregate and display salient performance metrics of a process(es) in the form of tables, graphs, line charts and bar charts for visual information.

## Related Works

Many researchers have proposed various techniques for data aggregation suitable for big data visualization to augment the effect of data latency and blocking, especially for interactive dashboards. Zgraggen *et al.* (2016) conducted a study to understand the effect of latency and blocking on user exploration of big data. The authors find that instantaneous result(s) delivered progressively led to the discovery of more insights. Similarly, Moritz *et al.* (2017) and Hellerstein, Haas and Wang (1997) reported that using the random sampling online aggregation technique yields more instantaneous outcomes for improving the interactive database systems that process analytical queries. Further, Rosenbaum and Schumann (2009) opined that visual data users have to prioritize data regions of interest for optimum results, while Mühlbacher *et al.* (2004) proposed progressive visual analytics (PVA) technique for big data visualization.

Another popular approach in big data visualization is the use of importance sampling to reduce the number of records that need to be read from across large databases. This approach is promising since the potential latency reduction is substantial. However, typical sampling systems still rely on offline preprocessing where these systems assume a set of predetermined queries that the analyst will use (i. e. known query workload) to build a representative sample of the original data. Park, Cafarella, and Mozafari, (2016), Ding *et al.,* (2016), and Fakete and Primet (2016) submitted that the importance sampling method is highly promising as the latency reduction is considerable with an overload of offline pre-processing. Once the samples are computed, the analyst’s queries are executed using the smaller, sampled data to speed up the computation.

Responsive interactive data visualization was supported by a data cube. Pahins *et al.* (2017) and Wang *et al.* (2017) proposed different types of partial results or data structures were pre-computed to speed up query execution. They both concluded that data cubes efficiently store the results of a carefully crafted aggregation query that can be used to answer similar subsequent aggregation queries. The downside to the data cube technique is that its storage requirement increases exponentially as the number of data dimensions increases. Also, the data cube limits query to referencing the predetermined grouping attributes, as some time would be expended to re-build the appropriate data cubes for each new query. Meanwhile, (Procopio *et al.,* 2019) stated that big data visualizations using data cube exploratory queries are no longer desirable in so far as the structure of the tables and attributes to be analyzed were unknown beforehand.

Researchers have also explored the use of other big data visualization techniques such as ripple join, predictive prefetching (Cetintemel *et al.,* 2018), specialized databases methods like column stores (Stonebraker *et al.*, 2005) and in-memory databases (Kemper & Neumann, 2011). Many other techniques can be used to support big data visual exploration and analysis (Godfrey, Gryz, & Lasek, 2016).

In dashboard design, online aggregation is intertwined with visualization interfaces and underlying data processing systems that analyze the data rendered on the dashboard. To achieve this, Hellerstein, Haas and Wang (1997) proposed a random sampling online aggregation technique for improving the interactive database systems that process analytical queries.

Progressive computation where data is processed incrementally in small chunks was also considered. Progressive systems are an extension of sampling-based approaches that incrementally compute results over increasingly larger samples to provide more accurate results to the user over time. This concept is well known in the graphics domain and is widely used on websites to improve the user experience when loading high-resolution images. It offers an interesting tradeoff between result accuracy and computation speed (Harrison, Dey, & Hudson, 2010).

This study is leveraged on Wander Join (Procopio, Scheidegger, Wu, & Chang, 2019) technique which supported fast, iterative queries that do not require pre-computation and large storage spaces adequate for big data visualization. A variant of Wander join is the selective Wander join technique proposed by Li, Wu, Yi, and Zhao (2016) which was a progressive visualization system that enables the user to prioritize data analysis of interest across multiple tables in data joins. The selected data are weighted appropriately, thereby enabling online aggregation values to converge faster. Finally, as part of efforts towards faster data aggregation and convergence for visualization, (Badam *et al.,* 2017) demonstrated interface controls and visualization augments for progressive visualization steering.

With the recent increase in data and complex algorithms to bound latency, recent researchers have proposed an approach called Progressive Visual Analytics (PVA) (Luo *et al.,* 2020). Instead of performing long computation, PVA quickly adjusts the parameters, generates estimates of the results and updates them continuously for data visualization. This approach aligns with the Wander join technique adopted in this study.

## Method

Dashboards visualization requires that the underlying data must be readily available, sometimes interactively as explained earlier. Meanwhile, data to be visualized must be stored on an auxiliary storage disk or in remote databases. As explained earlier, the queries needed to populate a visual dashboard could lag due to latency resulting in long waiting times thereby diminishing the user's ability to quickly explore the data. Also, recall that as the data increases in size and complexity, the latency in response worsens, especially if the analysis query involves data requests across multiple database tables as in the following query which finds the average airline arrival per airport in a region:

```
SELECT FLIGHT.airliner_name, AIRPORT.terminal_location, AVG
(TERMINAL.arrival_no)
FROM    FLIGHT, AIRPORT, ARRIVAL
WHERE FLIGHT.airliner_id = AIRPORT.airliner_id AND AIRPORTT.airliner_id =
          ARRIVAL.location_id
GROUP BY ARRIVAL.location;
```

In this query, three tables, FLIGHT, AIRPORT and ARRIVALS are joined with a common attribute (as usual), which were set equal to each other in the WHERE clause as depicted in Figure 2.

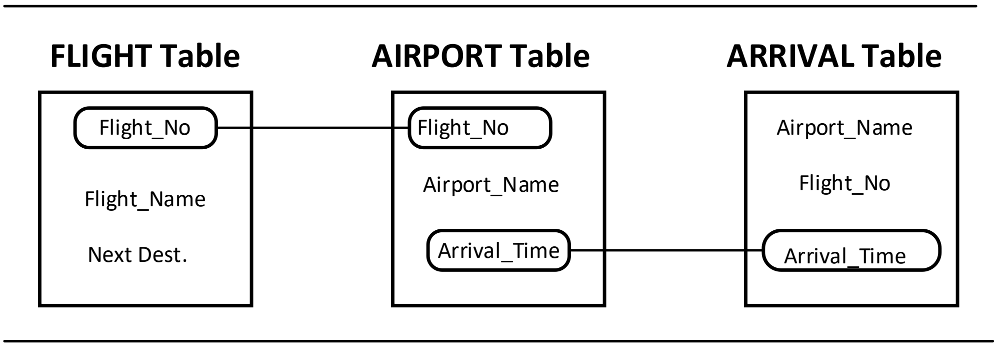


**Fig. 2: A 3-way relational join.**

This query fetches data from three different tables, combines them with the WHERE clause, partition the result into locations and finds the average per location. Such a JOIN operation is computationally expensive since the JOIN operation has to fully scan the three input tables entirely. Instead, Wander Join (WJ) visualization technique was used to mitigate data processing latency issues by prioritizing data analysis of interest while performing exploratory analysis, across multiple tables during data joins.

The selected data for the different components of the dashboard were weighted appropriately so that these records were sampled more often in the online aggregation query so that the query values can converge faster. To achieve this, the researchers applied importance sampling of Wander Join online aggregation method, by ensuring even sampling of different components when there is an uneven distribution of data across the grouping attributes.

Wander Join was implemented by extending the Python Wander Join implementation to use importance sampling. The source code for the implementation of Wander Join, Selective Wander Join and the evaluations are publicly available on http://github.com/promanrand/selectivewanderjoin as well as the "flight" data used in the evaluations are available on http://stat-computing.org/expo/2009/2008.csv.bz2. The data contain flight arrival details for all commercial flights in the USA. A large dataset of 120 million records in 1.6 Gigabytes compressed space (12 Gigabytes when uncompressed).

For example, if the first table in a Group By query is assigned a fractional weight, while the remaining tables are assigned binary weights (1 or 0). Also, if one allows 32 bits to store the fractional weight (of the first table), the total space needed to store the row index and weight for the first table in a Group By query would be less than 8 GB in a table with 1 billion rows. The remaining tables in the Group By query require the same space as a table in a filter query if the Group By query also includes filters. Otherwise, no weights need to be stored and rows are sampled uniformly. Table 2 shows the amount of memory needed to store the weights and row indexes for different sized tables in filtering or grouping queries.

**Table 2: The amount of memory needed to store the weights**

| Table Size (Rows) | Filter Query | Group By Query (First Table) |
|---|---|---|
| 10k | 40 kB | 80 kB |
| 100k | 400 kB | 800 kB |
| 1M | 4 MB | 8 MB |
| 10M | 40 MB | 80 MB |
| 100 M | 400 MB | 800 MB |
| 1 B | 4 GB | 8 GB |

Since Wander Join uniformly samples each record from the underlying table, each group's convergence rate depends on the proportion of records that belong to the group. Now, for a large sample record, it takes some time before the filter selects the candidate records, thereby delaying the query from making comparisons between groups during an online aggregation process. For example, if a table is JOINed by the attributes (location, month) and store the aggregate COUNT(*) for each combination of (location, month) values, then subsequent COUNT of any grouping or filtering operation over location, month, or both can use this table to quickly compute. Indicating that the JOIN operation speeds up subsequent similar query executions.

Wander Join leverages importance sampling and random sampling to enable user-driven interactivity with database relations in the sampling and dashboard display processes. In this study, the Wander Join interactive techniques were applied that allow the user to monitor and control the sampling rate of each group. This gives users more transparency during the online aggregation process and prioritizes their data of interest. Though importance sampling adds additional overhead its impact does not significantly affect performance as it performs no worse than other data sampling methods considered.

Recall that the main idea of the importance sampling technique was to lower the variance since certain input attributes have more impact on the sampling records than another. Hence, the researchers (Gonçalves et al., 2012) employed the random sampling biasing technique in equation (1).

$$\boldsymbol{g^{*} = min_{\ g}\ var_{g}\left(X\frac{f(X)}{g(X)}\right)}, \qquad (1)$$

where X is the data samples with $\frac{f(X)}{g(X)}$ as the likelihood ratio and $f$ is the probability density function of the desired distribution and $g$ is the probability density function of the biased distribution hence we, chose the sample distribution $g$ that minimizes the variance.

$$\boldsymbol{g^{*}(X) = \frac{|X|f(X)}{\int |x|f(x)dx}} \qquad (2)$$

Numerically, it can be shown that equations (1) and (2) are equivalent, also when $X \geq 0$, the variance becomes 0. By this, the probability mass shits into the desired event region and minimizes the dimensionality effect and does not diminish the efficiency. The essence is to choose a distribution which "propels" the important values, then the simulated outputs were weighted (using likelihood ratio) to correct for the use of the biased distribution with weighting function as follows:

$$\boldsymbol{W(x) = a\left(\frac{f(x)}{f(x/a}\right)}, \qquad (3)$$

where **aX** is a weighted random variable, for **a>1.** Here, the samples' frequencies were sorted directly according to their weights such that "important" or the desired data values of interest are selected for visualization.

Thus, as indicated above, this study used importance sampling approach to optimize Group By queries to uniformly sample from each group to ensure uniform convergence rate. The desired outcome was achieved by weighting the records in each group relative to the number of records in the group and the number of total groups. Specifically, the weight of each record was set in the table referenced in the Group By clause as:

$$\omega_i = \frac{1}{\alpha\beta} \qquad (4)$$

where ɑ is the number of records in the same group as record *i*, which was computed when the index was updated; *β* is the number of distinct groups. Numerically, equations (3) and (4) are similar and the idea was to sample from each group and records evenly, hence, the term $\frac{1}{\alpha}$ ensures uniform sampling of records in the same group, while $\frac{1}{\beta}$ ensures uniform sampling of records from across each group.

With this weighting, Wander Join randomly selects from the Group By table first, with even sampling from each group, see Figure 3.

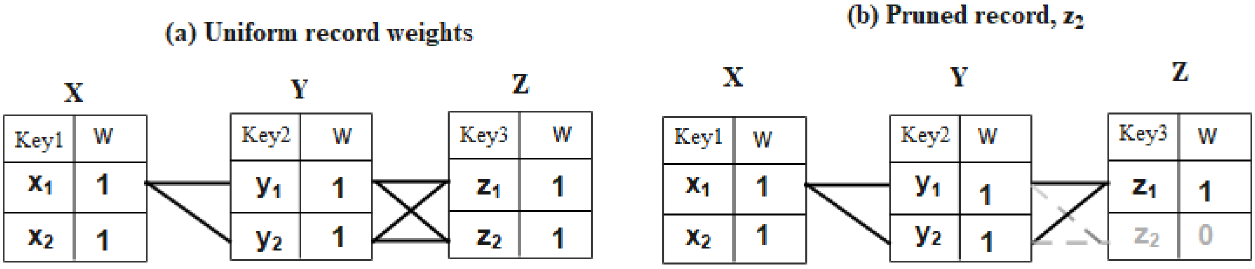


**Fig. 3: (a) A three-table join with uniform record weights**
**(b) The join outcome with pruned record**

In Figure 3, Suppose table X has the attribute one is grouping on and records $x_1,..x_n$ are weighted using Equation (3). Records in Table Y and Table Z will be weighted evenly. The sampling rate need not be varied, since uniform sampling by group is assured in Table X.

In Figure 3a, notice that the weight of each record was set evenly to 1. When a query is issued and a record fails the filter, then the failed record's weight is set to 0 to ensure that the record is not sampled again in subsequent filters. Such records are pruned from all paths that lead to them, see Figure 3b. Also, there are two possible paths to sample: $x_1 \rightarrow y_1 \rightarrow z_2$ and $x_1 \rightarrow y_2 \rightarrow z_2$. If the Wander Join algorithm selects $x_1$, then $y_1$, then $z_2$ and record $z_2$ fails the filter, $z_2$'s weight is set to 0 and is pruned. Hence, $x_1 \rightarrow y_1 \rightarrow z_2$ and $x_1 \rightarrow y_2 \rightarrow z_2$ are eliminated as possible paths. By pruning out the failed records, such records are excluded from sampling again. Thus, this technique prevents records sampling multiple times, especially for big data. Consequently, dashboards can connect via XML or HTML for such pruned data sources. A robust dashboard intelligently gathers, processes and displays grouped data without disrupting the workflow.

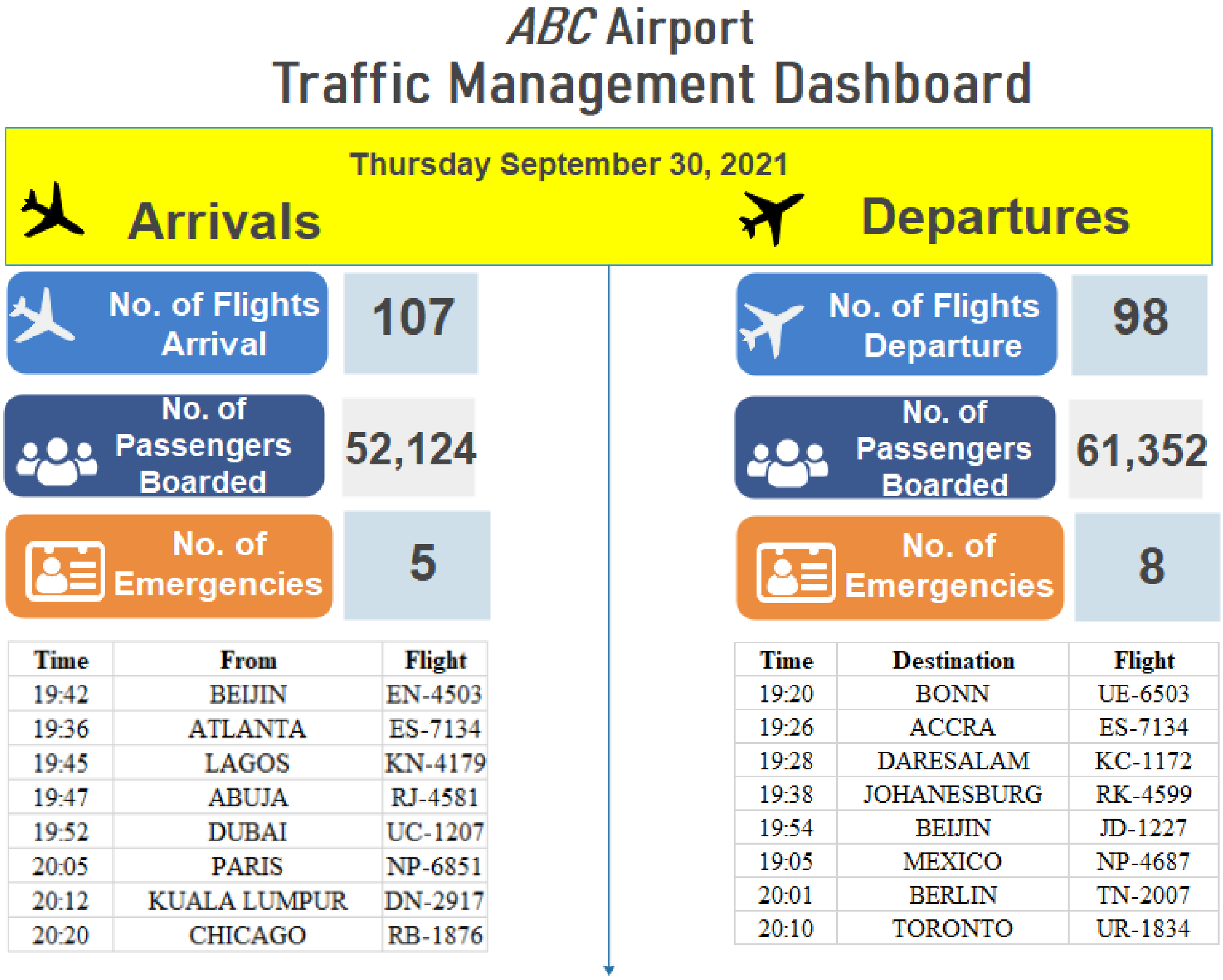
## Results

In the simulated case study experiments conducted for this study. A study of the number of flights arrival and departure from a busy United States of American airport was conducted using an online archived data on *http://stat-computing.org/expo/2009/2008.csv.bz2*, 30,000 samples out of 120 million records were sampled and it was sufficient for all engine types to reach 0.05 relative error. It was observed that the Wander Join algorithm sample required 977,432 samples for all groups to reach 0.05 relative error (see Table 3).

**Table 3: The number of samples required by Wander Join for all groups to achieve the same relative error rates for the flights' Group By group**

| Relative Error | Wander Join Algorithm | Sample Ratio (%) |
|---|---|---|
| 0.05 | 977,432 | 1.02% |
| 0.06 | 579,110 | 0.68% |
| 0.07 | 439,061 | 0.69% |
| 0.08 | 321,732 | 0.57% |
| 0.09 | 238,492 | 0.72% |
| 0.1 | 221,116 | 0.61% |

Specifically, the selective Wander Join algorithm allows users to readjust the weights so that groups of interest are sampled more often. This allows faster convergence on the dashboard and uniform sampling of the remaining groups. Additionally, groups of even less interest can be weighted to be sampled less often, or not at all if the weight is set to 0.



ABC Airport
Traffic Management Dashboard

Thursday September 30, 2021

Arrivals

Departures

No. of Flights Arrival: 107

No. of Passengers Boarded: 52,124

No. of Emergencies: 5

No. of Flights Departure: 98

No. of Passengers Boarded: 61,352

No. of Emergencies: 8

| Time | From | Flight |
|---|---|---|
| 19:42 | BEIJIN | EN-4503 |
| 19:36 | ATLANTA | ES-7134 |
| 19:45 | LAGOS | KN-4179 |
| 19:47 | ABUJA | RJ-4581 |
| 19:52 | DUBAI | UC-1207 |
| 20:05 | PARIS | NP-6851 |
| 20:12 | KUALA LUMPUR | DN-2917 |
| 20:20 | CHICAGO | RB-1876 |

| Time | Destination | Flight |
|---|---|---|
| 19:20 | BONN | UE-6503 |
| 19:26 | ACCRA | ES-7134 |
| 19:28 | DARESALAM | KC-1172 |
| 19:38 | JOHANESBURG | RK-4599 |
| 19:54 | BEIJIN | JD-1227 |
| 19:05 | MEXICO | NP-4687 |
| 20:01 | BERLIN | TN-2007 |
| 20:10 | TORONTO | UR-1834 |

**Fig. 4: An airport traffic management Dashboard**

Figure 4 is a typical dashboard that fetches data from several backend tables. For instance, the dashboard shows the daily total of the number of flight arrivals and the daily totals of all departed flights and the cumulative number of passengers aboard at ABC Airport. As an operational data visual, the selected data for display must converge timelessly with near-zero latency when the aggregation occurs.

## Discussion

The Wander Join technique enables the dashboard to display the daily total of the number of flight arrivals and the daily totals of all departed flights and the cumulative number of passengers at ABC Airport with minimal latency using the interactive queries. Similarly, the arrival time, origin and destination of flights converge faster with near-zero latency when the aggregation of the total number of flights arrival, the total number of passengers on board as well as the total number of emergencies recorded for the day occurs without the need for pre-computation and extra storage.

The Wander Join optimization is apposite for use in data analytics as it was put to use in this regard. For instance, the weights readjustment to ensure that groups of interest are sampled more often as specified in the WHERE and GROUP BY clauses and excluding the records that failed the selection criteria from being sampled subsequently.

Hence, samples that are more likely to satisfy the filters were prioritized by integrating the concept of importance sampling into Join queries by weighting them based on their importance to the query. In this case, the sampling rate of each group alters and therefore increases or decreases the convergence rate for the targeted group. In addition, a visual interface was designed to allow the users to see the sampling ratio of all groups relative to one another, especially when adjusting the weight applied to each group. This allows Wander Join to uniformly sample based on group and all groups to converge at the same rate. It is assured that users are not waiting for a group with low membership to converge.

## Conclusion

A wide variety of strategies, including precomputation, prefetching, sampling, and progressive computation, have been proposed by the researcher community to overcome the latency issue of dashboard display or generally, visualization. However, each of these approaches comes with its own set of advantages and challenges.

Through this work, the use of the importance sampling technique to support joins for common visualization queries, was amplified for data convergence with minimal delay for visualization. Thus, providing a uniform convergence rate for all GROUP BY categories, regardless of data membership in each group. The Wander Join variant enables interaction techniques to view and adjust sampling rates as in progressive visualizations.

## References

Badam, S. K., Elmqvist, N., & Fekete, J. D. (2017). Steering the craft: UI elements and visualizations for supporting progressive visual analytics. In *Computer Graphics Forum*; Wiley Online Library: Hoboken, NJ, USA, 36, 491–502.

Battle, L., Chang, R., & Stonebraker, M. (2016). Dynamic prefetching of data tiles for interactive visualization. In *Proceedings of the 2016 International Conference on Management of Data*, San Francisco, CA, USA, pp. 1363–1375.

Cetintemel, U., Cherniack, M., DeBrabant, J., Diao, Y., Dimitriadou, K., Kalinin, A., Papaemmanouil, O., & Zdonik, S.B. (2018). Query Steering for Interactive Data Exploration. In *proceedings of the conference on innovative data systems research (CIDR),* Asilomar, CA.

Ding, B., Huang, S., Chaudhuri, S., Chakrabarti, K., & Wang, C. (2016). Sample + Seek: Approximating aggregates with distribution precision guarantee. In *Proceedings of the 2016 International Conference on Management of Data,* San Francisco, CA, USA, 679–694.

Few, S. (2006). *Information dashboard design: The effective visual communication of data*. O'Reilly Media Inc., Italy.

Godfrey, P., Gryz, J., & Lasek, P. (2016). Interactive visualization of large data sets. *IEEE Transactions on Knowledge and Data Engineering*, *28*(8), 2142-2157.

Gonçalves, I., Silva, S., Melo, J. B., & Carreiras, J. M. (2012). Random sampling technique for overfitting control in genetic programming. *In European Conference on Genetic Programming*, 218-229.

Harrison, C., Dey, A. K., &. Hudson, S. E. (2010). Evaluation of progressive image loading schemes. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* ACM, 1549–1552.

Hellerstein, J. M., & Haas, P. J. Wang., HJ (1997) Online aggregation. In *Proc. ACM–SIGMOD Conference on Management of Data*, 171-182.

Kemper, A., & Neumann, T. (2011). HyPer: A hybrid OLTP&OLAP main memory database system based on virtual memory snapshots. In *2011 IEEE 27th International Conference on Data Engineering, Hanover, Germany,* 195-206.

Kosara, R. (2016). Presentation-oriented visualization techniques. *IEEE Computer Graphics and Applications*, 36(1), 80-85.

Li, F., Wu, B., Yi, K., & Zhao, Z. (2016). Wander Join: Online Aggregation via Random Walks. In *Proceedings of the 2016 International Conference on Management of Data*, San Francisco, CA, USA, 615–629.

Luo, Y., Chai, C., Qin, X., Tang, N., & Li, G. (2020). Visclean: Interactive cleaning for progressive visualization. *Proceedings of the VLDB Endowment,* 13(12), 2821-2824.

Nusrat, S., Harbig, T., & Gehlenborg, N. (2019, June). Tasks, techniques, and tools for genomic data visualization. *In Computer Graphics Forum*, 38(3), 781-805.

Moritz, D., Fisher, D., Ding, B., & Wang, C. (2017). Trust, but verify: Optimistic visualizations of approximate queries for exploring big data. In *Proceedings of the 2017 CHI conference on human factors in computing systems, Denver, USA,* 2904-2915.

Mühlbacher, T., Piringer, H., Gratzl, S., Sedlmair, M., & Streit, M. (2014). Opening the black box: Strategies for increased user involvement in existing algorithm implementations. *IEEE Transactions on Visualization and Computer Graphics*, *20*(12), 1643-1652.

Pahins, C.A., Stephens, S.A., Scheidegger, C., & Comba, J.L. (2017). Hashed cubes: Simple, low memory, real-time visual exploration of big data. *IEEE Transactions on Visualization and Computer Graphics*, 23(1), 671–680.

Park, Y., Cafarella, M., & Mozafari, B. (2016). Visualization-aware sampling for very large databases. *In Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE),* Helsinki, Finland, 755–766.

Procopio, M., Scheidegger, C., Wu, E., & Chang, R. (2019). Selective Wander Join: Fast Progressive Visualizations for Data Joins. *In Proceedings of Informatics Multidisciplinary Digital Publishing Institute.*, 6(1), 1-14.

Qin, X., Luo, Y., Tang, N., & Li, G. (2020). Making data visualization more efficient and effective: a survey. Proceedings of t*he VLDB Journal,* 29(1), 93-117.

Rosenbaum, R., & Schumann, & H. (2009). Progressive refinement: More than a means to overcome limited bandwidth. *In Proceedings of the International Society for Optics and Photonics Electronic Imaging, (IS&T/SPIE),* San Jose, CA, USA.

Stonebraker, M., Abadi, D. J., Batkin, A., Chen, X., Cherniack, M., Ferreira, M., Lau, E., Lin, A., Madden, S., & O'Neil, E. (2005). C-store: A column-oriented DBMS. In Proceedings of the 31st International Conference on Very Large Data Bases, Trondheim, Norway, 553–564.

Wang, Z., Ferreira, N., Wei, Y., Bhaskar, A. S., & Scheidegger, C. (2017). Gaussian cubes: Real-time modeling for visual exploration of large multidimensional datasets. *IEEE Transactions on Visualization and Computer Graphics*, *23*(1), 681-690.

Wu, E., & Nandi, A. (2015). Towards perception-aware interactive data visualization systems. *In Proceedings of the DSIA Workshop*, Chicago, IL.

Wu, E., Psallidas, F., Miao, Z., Zhang, H., Rettig, L., Wu, Y., & Sellam, T. (2017). Combining Design and Performance in a Data Visualization Management System. *In Proceedings of the Conference on Innovative Data Systems Research,* Chaminade, CA, USA.

Zgraggen, E., Galakatos, A., Crotty, A., Fekete, J. D., & Kraska, T. (2016). How progressive visualizations affect exploratory analysis. *IEEE Transactions on Visualization and Computer Graphics,* 23(8), 1977-1987.