# Build your own Octopus(OctopusDB). Blinktopus

Pavlo Shevchenko, Ali Hashaam, Guzel Mussilova, Ali Memon
Otto-von-Guericke-University, Magdeburg
firstname.lastname@st.ovgu.de

Gabriel Campero Durand
Otto-von-Guericke-University, Magdeburg
gabrielcampero@acm.org

***Abstract*—What is this paper about?**

## I. Introduction

### A. Background and Motivation

Over the last decades we are witnessing that modern enterprises need to pick only specialized DBMSs(e.g. OLAP, OLTP, streaming systems and etc.) each tailored to their specific use-case. Consequently, it leads to additional costs in terms of licensing, maintenance, integration and DBA. Nevertheless, it is affordable for some companies, to adapt these integrated solutions to constantly changing workload and requirements. However, it may still be a challenging and non-trivial task to achieve. Thus, in order to cope with these problems an implementation of new all-purpose system will be a perfect solution.

Nowadays there exists a great variety of systems that claim to solve the aforementioned problems. One of the most radically departing from existing architectures solutions was proposed by Jens Dittrich and Alekh Jindal - a new type of database system, coined OctopusDB[1]. By mimicking several types of systems, OctopusDB shows a considerably better performance. Meanwhile, it became evident that a system performance can be further enhanced by applying techniques from completely different perspectives. As long as it is plausible for a system to retrieve results approximately rather than exactly, Approximate Query Processing(AQP) comes into play. Several successful examples(e.g. BlinkDB, SnappyData on Facebook's Presto) have already proved an efficiency of an implementation of approximation features into existing DBMSs[5]???. Therefore, we believe that combining OctopusDB and AQP techniques will improve the performance of our system by several orders of magnitude. Thus, our goal is to provide a user a tool called Blinktopus, that will allow to build his own prototype of OctopusDB with embedded AQP techniques.

In Section 2 we start with the brief discussion of what is the idea behind OctopusDB and AQP. Moreover, in Section 2 we introduce the concepts of AQP main data synopses and explain why exactly one type of synopsis was chosen for our system. In Section 3 we discuss an experimental part of our project where we provide the results of the tests performed on the Storage Views(SVs) with the different physical layouts, namely LogSV, ColSV and RowSV. In Section 4 we present the related work. We conclude this paper in Section 5.

## II. Fundamentals

### A. OctopusDB

Main Idea. Architecture briefly.

### B. Approximate Query Processing

Main Idea. Synopses for Massive Data: Histograms. Sketches. More on histograms. Probably something on HLL, if it's eventually included in Blinktopus.

## III. Experimental Part

Evaluation of the test results:

1. Comparison of runtimes of different types of SVs. 2. Comparison of the diff SVs by means of percentage representation.

## IV. Related Work

Some related work has to be mentioned here.

## V. Conclusions

Sum up what happened

## VI. Future Work

## Acknowledgement

Thanks to Gabriel, DBSE

## References

[1] Dittrich, Jens, and Alekh Jindal. "Towards a One Size Fits All Database Architecture." CIDR. 2011.
[2] Jindal, Alekh. "OctopusDB: flexible and scalable storage management for arbitrary database engines." (2012).
[3] Jindal, Alekh. "The mimicking octopus: Towards a one-size-fits-all database architecture." VLDB PhD Workshop. 2010.
[4] Mozafari, Barzan. "Approximate query engines: Commercial challenges and research opportunities." SIGMOD, 2017.
[5] Mozafari, Barzan, and Ning Niu. "A Handbook for Building an Approximate Query Engine." IEEE Data Eng. Bull. 38, no. 3 (2015): 3-29.