**Universidad de las Américas Puebla**

Proyecto final – Laboratorio de manejadores de bases de datos

Carlos Andrés Reyes Evangelista

157068

**Big Data Management and Processing**

**Introducción**

La intención del presente reporte es presentar una sinopsis de los conocimientos adquiridos durante la realización de las prácticas del Laboratorio de Manejadores de Bases de Datos. Estos conocimientos refieren y están relacionados con diversos paradigmas que atañen la administración y almacenamiento de datos, mas no es ésa la única característica que comparten: todos están relacionados con el renombrado concepto de Big Data.

La noción de Big Data no abarca únicamente, como especifican Adiba, Castrejón, Vargas, Zechinelli y Espinosa (2016), “un diluvio de datos para procesar y administrar grandes volúmenes de bytes. Sino que además de esta visión superficial existe un consenso sobre las tres V que caracterizan a Big Data: Volumen, Variedad y Velocidad”, lo que implica que la percepción sobre Big Data es más profunda que lo que un terrenal pensamiento normalmente englobaría. El doctor Zechinelli (2019) mencionó en clase que Big Data “es una colección de datos tan masiva que es difícil o imposible (impráctica) de manejar con una base de datos tradicional”, pero que además es caracterizada por las tres V referenciadas previamente, éstas son definidas como sigue: Volumen se refiere a la cantidad de datos, pero “no sólo a los datos estáticos, sino a los datos en flujo” (Zechinelli, 2019); Variedad trata sobre los “diferentes tipos de representaciones que los datos pueden adoptar” (Adiba, Castrejón, Vargas, Zechinelli & Espinosa, 2016); y finalmente, Velocidad hace mención a la rapidez con que los datos son producidos y consumidos.

En principio, las tres tecnologías que fueron estudiadas y puestas en práctica están relacionadas con Big Data en alguna forma, sin embargo, cada una de ellas aborda un enfoque distinto e independiente entre ellas. A continuación, es presentada una breve descripción de cada una de ellas y posteriormente se detallará el desarrollo de cada una de las prácticas realizadas para cada una de estas herramientas respectivamente.

**MongoDB** es un sistema de gestión de bases de datos NoSQL orientado a documentos. De acuerdo a Rick Cattell (2010), NoSQL significa «No sólo SQL» o «No Relacional», además un sistema NoSQL se caracteriza por ciertos puntos clave entre los cuales se incluye: la habilidad de escalar horizontalmente, capacidades de replicación y distribución, modelos de concurrencia más flexibles que ACID y la habilidad de dinámicamente añadir nuevos atributos a los registros de datos. MongoDB destaca en los primeros dos de ellos: integra un concepto conocido como Balanceo de Carga mediante el cual efectúa escalamiento horizontal al aplicar una técnica conocida como fragmentación (*sharding*). “MongoDB soporta fragmentación automática, distribuyendo documentos a través de los servidores” afirma Rick Cattell (2010), a su vez Vargas-Solar realza la importancia del *sharding* al aseverar que “sharding permite a los sistemas de bases de datos paralelos aprovecha la banda ancha de entrada/salida de múltiples discos al leer y escribir simultáneamente sobre ellos”, de esta manera, MongoDB balancea la carga y replica los datos para mantenerse resiliente.

**Apache Pig** es una herramienta de programación originalmente utilizada para implementar programas de tipo Map-Reduce a ser ejecutados por Hadoop para solucionar problemas en paralelo sobre grandes colecciones de datos que puedan resolverse mediante la aplicación de las funciones provenientes del paradigma funcional Map y Reduce, pero adaptadas para funcionar en ambientes paralelos o distribuidos. Vargas-Solar (2017) habla de los notables beneficios que Map-Reduce ofrece: “Ambientes en tiempo de ejecución basados en Map-Reduce proveen un buen rendimiento en arquitecturas sobre todo en pruebas analíticas que operan sobre vastas colecciones de datos”. Además de la ya mencionada, entre algunas de las características que hacen destacar a Pig se encuentra su lenguaje integrado conocido como *Pig Latin* que permite al usuario concentrarse en la semántica en lugar de la eficiencia debido a que, según la página oficial de Pig (“Welcome to Apache Pig”, 2018): “es trivial conseguir simples pruebas de análisis de datos en paralelo. Las pruebas complejas, compuestas de múltiples transformaciones de datos interrelacionadas, son explícitamente codificadas como secuencias de flujo de datos, haciéndolas fácil de escribir, entender y mantener”. *Pig Latin* también ofrece al usuario la capacidad de crear sus propias funciones para procesamiento específico.

**GraphX** es una herramienta de alto nivel empleada por la plataforma Apache Spark, especializada en computación en clúster sobre el que se pueden realizar tanto algoritmos iterativos como análisis de datos interactivos o exploratorios. GraphX es lo que se conoce como un método de acceso de tipo Grafo. Hellerstein, Stonebraker y Hamilton (2007) definen un método de acceso como “una colección de algoritmos y estructuras para organizar y acceder a los datos”, mientras que un método de acceso de tipo grafo, aunque no es propiamente definido, es descrito por Athanassoulis e Idreos (2016) como un tipo de método de acceso que se mimetiza con el sistema propio debido a dos razones: “la primera, la representación en forma de grafo juega un rol importante en el rendimiento y cada representación puede requerir una muy diferente manera de acceder a los datos; en segundo lugar, ha habido un mucho menor esfuerzo en la estandarización comparado contra los sistemas relacionales, por lo tanto se permiten soluciones más nativas que no se adhieren a un estándar global”. GraphX hace uso también de los llamados RDD o Conjunto de datos Resiliente Distribuido (*Resilient Distributed Dataset*) que forman parte de la arquitectura de Apache Spark; este tipo especial de conjuntos “representan una colección de objetos de sólo-lectura particionados a través de un conjunto de máquinas y que pueden ser reconstruidos si una partición es perdida” (Zaharia, Chowdhury, Franklin, Shenker & Stoica, 2010), de ahí que su nombre incluya el adjetivo *resiliente*, que hace referencia a la capacidad humana de recuperarse ante situaciones adversas.

**Descripción de la práctica 1**

**Objetivo**

**Pasos principales**

**Análisis y presentación de los resultados**

**Conclusiones**

**Descripción de la práctica 2**

**Descripción de la práctica 3**

**Discusión general sobre lo aprendido en las prácticas**

La civilización evoluciona progresiva y súbitamente y con ella lo hace su necesidad de información, así que los sistemas que ofrecen servicios de esta índole se ven obligados a evolucionar a la par, como resultado los arquetipos que los sistemas de gestión de datos pueden seguir han visto incrementado su espectro en amplia variedad en los últimos años; este incremento en los paradigmas ha generado nuevos tipos de sistemas con características diversas que cubren y proponen soluciones a problemáticas diversas. A lo largo de la impartición del curso hemos sido testigos y hemos formado parte del hecho de que entre más conoces del mundo, más amplio se vuelve, en concreto: lo que otrora se limitaría a Sistemas de Gestión de Bases de Datos Relacionales ahora ha sido extendido a Sistemas NoSQL, Distribuidos, Paralelizados, Depósitos de Datos, sistemas con distintos Métodos de acceso y la lista continúa.

Específicamente, como alumno, he aprendido que los Sistemas de Gestión de Bases de Datos Relacionales, aunque parten de una base muy sólida y bien estructurada, no son panacea y cómo el no cerrarse a las posibilidades o extensiones posibles puede generar paradigmas distintos que, aunque ofrezcan ciertas limitaciones y debilidades frente a los SGBDR, pueden ofrecer alternativas que solucionen mejor cierta clase de problemas, en referencia, por supuesto, a los Sistemas NoSQL de los que ahora entiendo que, pese a no seguir una serie de lineamientos tan estricta como Relacional en el sentido de que no cumplen con “formas normales” o propiedades ACID, sus ventajas se hacen claras al cambiar el lente de perspectiva, pues al sacrificar un poco de coherencia y modelo de concurrencia, es posible obtener beneficios formidables como la replicación, distribución, escalamiento, flexibilidad y eficiente uso de índices.

Además, he comprendido el rol que juega la arquitectura Map-Reduce en pruebas de procesamiento intensivo sobre extensas colecciones de datos para paralelizar la ejecución. Tenía ya conocimiento previo sobre las funciones *map* y *reduce* del paradigma funcional, mas ahora soy capaz de comprender las adaptaciones que se realizaron sobre estas funciones para mantener su esencia, pero adecuarlas a un ambiente de trabajo distribuido y paralelo.

También adquirí conocimientos sobre los diversos métodos de acceso a datos existentes

**Bibliografía**

[1] Adiba, M., Castrejón, J., Vargas, G., Zechinelli, J., & Espinosa, J. (2016). *Big Data Management: Challenges, Approaches, Tools and their limitations* [Ebook].

[2] Catell, R. (2010). *Scalable SQL and NoSQL Data Stores* [Ebook].

[3] Vargas-Solar, G. (2017). *Efficient data management for putting forward data centric sciences* [Ebook].

[4] Kerstern, M., Idreos, S., Manegold, S., & Liarou, E. (2011). *The Researcher’s Guide to the Data Deluge: Querying a Scientific Database in Just a Few Seconds* [Ebook].

[5] Athanassoulis, M., & Idreos, S. (2016). *Design Tradeoffs of Data Access Methods* [Ebook].

[6] Idreos, S., Athanassoulis, M., Guo, D., Kester, M., Maas, L., & Zoumpatianos, K. (2016). *Past and Future Steps for Adaptive Storage Data Systems: From Shallow to Deep Adaptivity* [Ebook].

[7] J. M. Hellerstein, M. Stonebraker, and J. R. Hamilton. Architecture of a Database System. Found. Trends Databases, 1(2):141–259, 2007.

[8] Zaharia, M., Chowdhury, M., Franklin, M., Shenker, S., & Stoica, I. (2010). *Spark: Cluster Computing with Working Sets* [Ebook]. Retrieved from https://amplab.cs.berkeley.edu/wp-content/uploads/2011/06/Spark-Cluster-Computing-with-Working-Sets.pdf

Welcome to Apache Pig!. (2018). Retrieved from https://pig.apache.org