Revista Digital del Departamento de Ingeniería e Investigaciones Tecnológicas de la Universidad Nacional de la Matanza

ISSN: 2525-1333
Vol.: 5 - Nro. 2 (DICIEMBRE 2020)

Artículo original

TELEMETRÍA DE ALTAS PRESTACIONES SOBRE BASE DE DATOS DE SERIE DE TIEMPOS

HIGH PERFORMANCE TELEMETRY OVER TIME SERIES DATABASES

Pablo SOLIGO [(1)], Jorge Salvador IERACHE[(2)] y Germán MERKEL[(3)]

[(1)]Universidad Nacional de La Matanza - Departamento de Ingeniería e Investigaciones Tecnológicas
psoligo@unlam.edu.ar

[(2)] Universidad Nacional de La Matanza - Departamento de Ingeniería e Investigaciones Tecnológicas
jiearche@unlam.edu.ar

[(3)]Universidad Nacional de La Matanza - Departamento de Ingeniería e Investigaciones Tecnológicas
gmerkel@unlam.edu.ar

Revista Digital del Departamento de
Ingeniería e Investigaciones
Tecnológicas de la Universidad
Nacional de la Matanza

ISSN: 2525-1333
Vol.: 5 - Nro. 2 (DICIEMBRE 2020)

Resumen:

Los mapeadores objeto relacional (ORM) presentan grandes ventajas en términos de productividad en el desarrollo de software. Los motores de base de datos relacionales (RDBMS) representan la herramienta dominante en términos de almacenamiento y recuperación. El uso de ORMs y RDBMS sin embargo presentan límites cuando se trabaja con grandes volúmenes de datos. En este trabajo analizamos la factibilidad de implementación de una base de datos de serie de tiempos sobre un prototipo de segmento terreno costo efectivo [1]. El objetivo es determinar si estas tecnologías híbridas, montadas sobre un motor relacional, permiten mantener los tiempos de almacenamiento y recuperación constantes para grandes volúmenes de datos, ofrecer capacidades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad, del inglés Consistency, Isolation and Durability) y, al mismo tiempo, compatibilidad con frameworks de desarrollo rápido y ORMS.

Abstract:

The object relational mappers (ORM) have great advantages in terms of productivity in software development. Relational database engines (RDBMS) represents the dominant tool in terms of storage and data access. The use of ORMs and RDBMS present however limits when working with large data volumes. In this paper we analyze the feasibility of implement a time series database over a prototype of cost-effective ground segment [1]. The goal is to determine if these hybrid technologies, mounted on a relational engine, allow maintaining constant storage and recovery times for large volumes of data, offering ACID (Atomicity, Consistency, Isolation and Durability) capabilities and, at the same time, compatibility with rapid development frameworks and ORMS.

Palabras Clave: Satélites, Segmento Terreno, Diseño de Software

Key Words: Satellites, Ground segment, Software Design.

ISSN: 2525-1333. Vol.: 5 - Nro. 2 (DICIEMBRE-2020)

I. CONTEXTO

La UNLaM (Universidad Nacional de La Matanza) posee un prototipo de segmento terreno denominado UGS (UNLaM Ground Segment) ([1], [2], [3] y [4]). En el corto plazo, bajo el proyecto PROINCE C230 el grupo de investigación GIDSA [5] se ha establecido como objetivo la publicación del prototipo [1] utilizando telemetría de la red satnogs [6] y de otras fuentes alternativas obligando a revisar la eficiencia en el acceso a los datos. El UGS ha sido desarrollado como una plataforma de investigación, experimentación y docencia, y tiene como premisa lograr que las soluciones para el área aeroespacial sean costo-efectivas desestimando alternativas que impliquen desarrollos ad-hoc, herramientas de escasa penetración en la industria de software propósito general o de costosa implementación. Desde su primera versión, el prototipo ha tenido como objetivo ser multimisión, pudiendo trabajar con satélites de organizaciones y fabricantes independientes, lo que ha condicionado decisiones de diseño. Utilizando un RDBMS, accede a la base de datos mediante un ORM produciendo un diseño normalizado, condición que limita la eficiencia en el acceso a los datos [7] pero contribuye a mantener la solución costo-efectiva [8].

II. INTRODUCCIÓN

Un sistema moderno de operación multimisión debe proveer de herramientas de explotación de datos que incluyan, entre otras, las siguientes características [9] y [10]:

- Habilidad de mantener de forma eficiente, segura y transparente los datos de la misión completa.
- Integración para cambiar entre datos históricos y de tiempo real.
- Una performance que permita obtener datos de años en unos pocos segundos.
- Manejar grandes volúmenes de datos y niveles de variabilidad y granularidad.
- Garantía de fiabilidad.
- Punto de acceso único disponible desde interfaces de programación de aplicaciones (API del inglés Application Programming Interface)

Estas características se vuelven especialmente importantes ante los avances disponibles en el área de aprendizaje automático, un área que exige alta disponibilidad y grandes volúmenes de datos históricos [11] para su explotación.

Los desafíos que presenta la administración de grandes volúmenes de datos no son nuevos, aunque en las últimas décadas el surgimiento de las redes sociales, aplicación de uso masivo en internet y la popularidad adquirida por dispositivos interconectados IoT(Internet de las cosas, del inglés Internet of Things) han propiciado la aparición de herramientas especialmente adaptadas. Las soluciones genéricamente denominadas NoSql (No solo Sql, del inglés Not Only Sql) intentan ofrecer una solución escalable al problema que presenta las, cada vez mayores, necesidades de almacenamiento y posterior recuperación.

ISSN: 2525-1333. Vol.: 5 - Nro. 2 (DICIEMBRE-2020)

Por otro lado los sistemas RDBMS clásicos han dominado casi desde de su aparición y lo siguen haciendo concentrando más del 80% de las soluciones de almacenamiento y recuperación [12], manteniendo su popularidad como muestra la figura 1 Popularidad de modelos de base de datos. El éxito de este tipo de almacenamiento tiene entre sus razones, que los datos empresariales ajustan al modelo relacional [12]. Un segmento terreno aeroespacial concentra ambas problemáticas, grandes volúmenes de datos con característica que se adaptan perfectamente al modelo relacional.

Las características de seguridad ACID [13] ofrecidas por un RDBMS, ausentes o parcialmente disponibles en un NoSql, también juegan un papel en la determinación del modelo de base de datos a implementar.

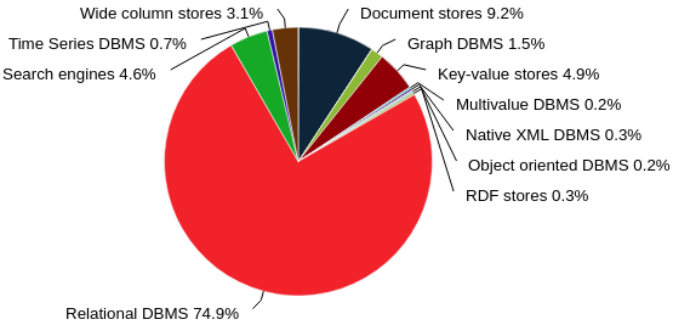

Fig 1 Popularidad de modelos de base de datos

Como muestra la Fig 2 Tabla de variables de ingeniería y relacionadas, el UGS posee una única tabla donde almacenar las variables de ingeniería, este diseño de tipo Narrow-table [14] y [15], es simple y a la vez lo suficientemente versátil para adaptarse a escenarios con diferente cantidad y tipos de sensores aunque afecta negativamente el rendimiento ya que cada registro almacena un único valor de telemetría. Bajo este diseño y utilizando RDBMS algunas entidades pueden sufrir problemas de escalabilidad si la cantidad de datos crece demasiado, problemas relacionados con las limitaciones en bases de datos estrictamente normalizadas.

Las soluciones a los límites impuestos por la tecnología pueden ir desde desnormalizaciones que incluyan vistas materializadas con datos conciliados, bases de datos NoSQL o soluciones híbridas que implementen capas lógicas adicionales a probados RDBMS [16].

Las desnormalizaciones sobre RDBMS, han demostrado ser exitosas en varios ámbitos de la industria y los segmentos terrenos no son la excepción. Grandes compañías lo han usado como estrategia [9] teniendo como una de sus grandes ventajas el uso de una única, probada y madura herramienta de persistencia y recuperación, evitando el uso de herramientas especiales para datos según su naturaleza. Como contrapartida, un diseño desnormalizado no puede por sí mismo garantizar la integridad, y conlleva un costo asociado a la implementación de lógica de control y actualización adicional.

Las soluciones NoSQL, o el uso de motores de persistencia que no posean características ACID debería circunscribirse un grupo de entidades particulares, aquellas que presentan características especiales ya sea de acceso, actualización o volumen de datos. En cualquier caso, esta estrategia obliga a utilizar una herramienta adicional, generando costos adicionales de mantenimiento y desarrollo.

El surgimiento en los últimos años de alternativas híbridas, adaptadas a problemáticas de administración de datos específicas, que ofrezcan mejores rendimientos sin perder la seguridad y estandarización presentes en las

ISSN: 2525-1333. Vol.: 5 - Nro. 2 (DICIEMBRE-2020)

RDBMS ha ganado atención incluso en ámbitos de máxima exigencia [17].

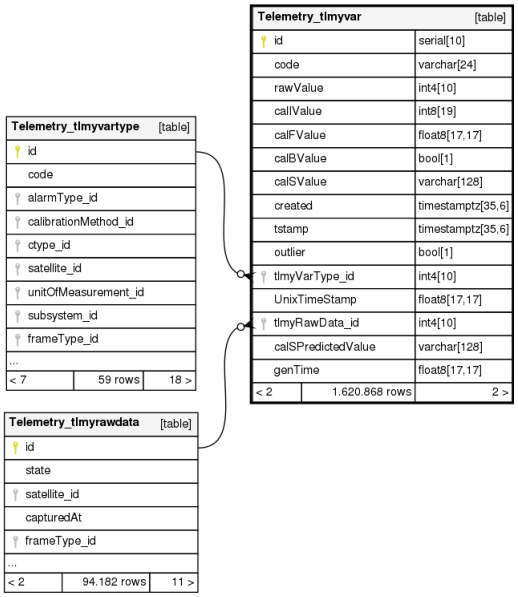

Fig 2 Tabla de variables de ingeniería y relacionadas

En este trabajo exponemos los resultados obtenidos de las simulaciones realizadas para evaluar las capacidades de estos sistemas híbridos sobre las necesidades de un segmento terreno de bajo costo. El objetivo es evaluar la viabilidad de utilizar esta tecnología que permitiría una migración transparente, costos de desarrollo y mantenimiento acotados y poder ofrecer rendimientos constantes de manera independiente a la cantidad de misiones y la cantidad de datos de cada una.

III. MÉTODOS

Con el objetivo de verificar las posibles diferencias de rendimiento y la escalabilidad de un sistema hibrido se ha utilizado la extensión de PostgreSQL, TimescaleDB [18] aplicada a la base de datos actual del UGS. No se han aplicado modificaciones al diseño buscando que continúe siendo accedida tanto como sea posible mediante un ORM.

Para las pruebas y simulación se ha tomado como referencia 3 satélites simulados con 7500 variables de telemetría con una frecuencia de reporte de 8 segundos. Utilizando la API de timescaleDB *create_hypertable* se generó una hipertabla *Telemetry_htlmyvar* replica de la tabla relacional *Telemetry_tlmyvar* y se han configurado periodos de tiempo en 2 horas buscando hacer coincidir el tamaño de los fragmentos o *chunks* con el tamaño recomendado por el equipo de desarrollo de *timescaledb* [15] y un periodo de tiempo discreto. A la tabla regular se le ha agregado un índice ordenado, no cluster, por fecha hora. Con esto último se intenta dar a la tabla regular herramientas similares que las que la hipertabla implementa.

La Tabla 1 - Hardware utilizado en las simulaciones, muestra una lista del equipamiento de hardware utilizado.

Tabla 1 - Hardware utilizado en las simulaciones

Material	Descripción
SO	UBUNTU
DBMS	Timescale sobre Docker
Equipo	
Procesador	AMD A10-7860K Radeon R7, 12 Compute Cores 4C+8G
Memoria	6916MiB
Disco	Western Digital 6TB SATA 3.5 Hard Drive.

La siguiente lista describe las simulaciones realizadas.

ISSN: 2525-1333. Vol.: 5 - Nro. 2 (DICIEMBRE-2020)

- Inserciones: Inserciones de telemetría semiordenada
- Consultas de tiempo real: 250 consultas desde entre 10 y 50 minutos antes de la fecha/hora del último registro hasta la fecha hora del último registro. Las consultas son realizadas por 5 procesos.
- Consultas históricas: 250 consultas desde entre 10 y 50 minutos en todo el rango de datos. Las consultas son realizadas por 5 procesos.
- Consultas agregadas: 250 consultas desde entre 10 y 50 minutos en todo el rango de datos agrupando cada 12 minutos.

Para las inserciones se genera de manera simulada telemetría para 4 paquetes para cada satélite, posteriormente se mezclan para evitar la inserción en orden y lograr un escenario más realista. Finalmente se insertan mediante el ORM en bloques de 1000.

IV. RESULTADOS

Las figuras Fig 3 Inserciones entre los 450 y 550 millones de registros y Fig 4 Inserciones entre los 1010 y 1060 millones de registros muestran un comportamiento similar e independiente a la cantidad de registros. Se observan mejores rendimientos para la hipertabla. La tabla regular muestra valores altos periódicamente para una inserción mientras que la hipertabla muestra un comportamiento totalmente estable.

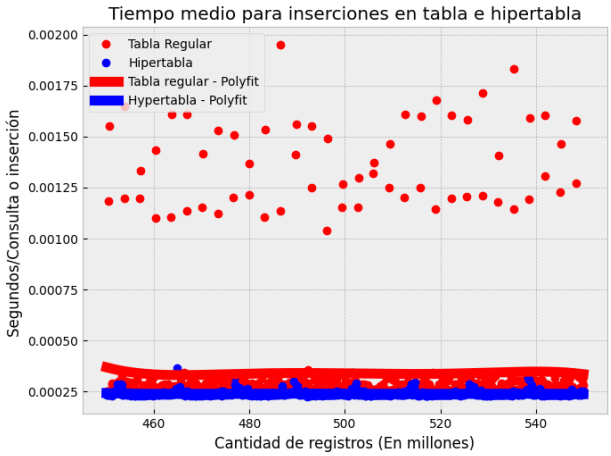

Fig 3 Inserciones entre los 450 y 550 millones de registros

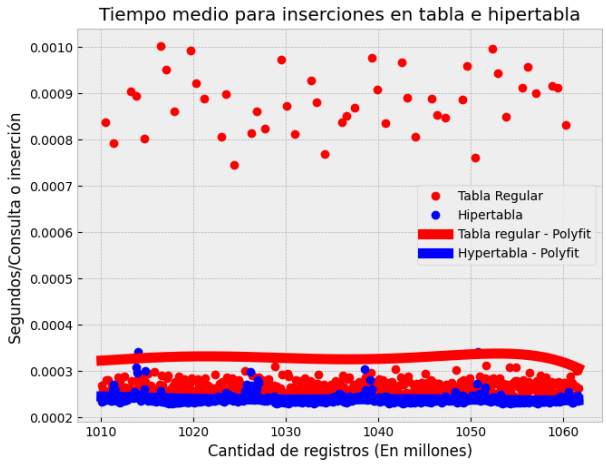

Fig 4 Inserciones entre los 1010 y 1060 millones de registros

Las Fig 5 Consultas de tiempo real entre 450 y 550 millones de registros y Fig 6 Consultas en tiempo real entre 1010 y 1060 millones de registros muestran un mejor rendimiento para la hipertabla, las consultas demoran de media un tiempo inferior que trabajando con tablas regulares.

ISSN: 2525-1333. Vol.: 5 - Nro. 2 (DICIEMBRE-2020)

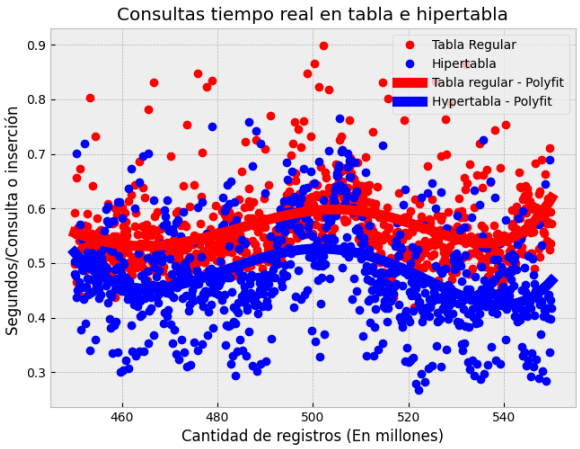

Fig 5 Consultas de tiempo real entre 450 y 550 millones de registros

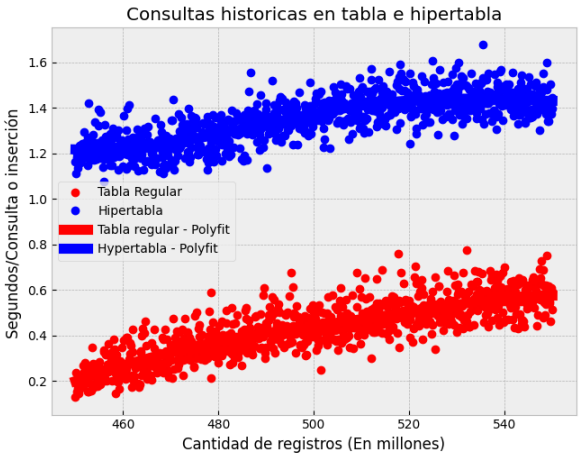

Fig 7 Consultas históricas entre 450 y 550 millones de registros

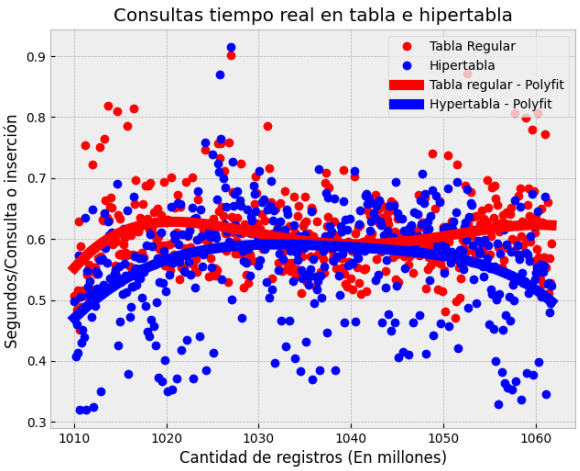

Fig 6 Consultas en tiempo real entre 1010 y 1060 millones de registros

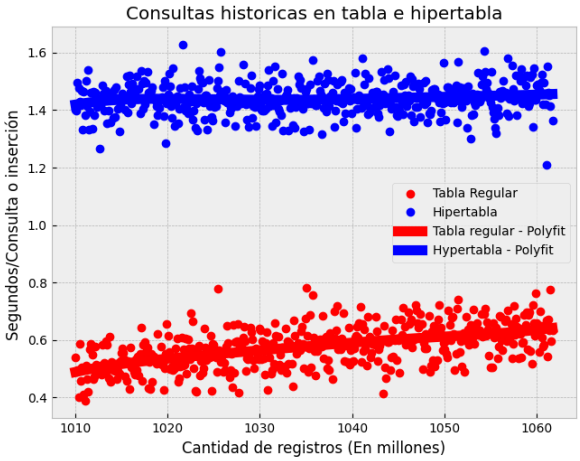

Fig 8 Consultas históricas entre 1010 y 1060 millones de registros

Las figuras Fig 7 Consultas históricas entre 450 y 550 millones de registros y Fig 8 Consultas históricas entre 1010 y 1060 millones de registros muestran, en contraste con las figuras previas, un mejor rendimiento para la tabla regular, llegando incluso a reducir a la mitad el tiempo de consulta.

ISSN: 2525-1333. Vol.: 5 - Nro. 2 (DICIEMBRE-2020)

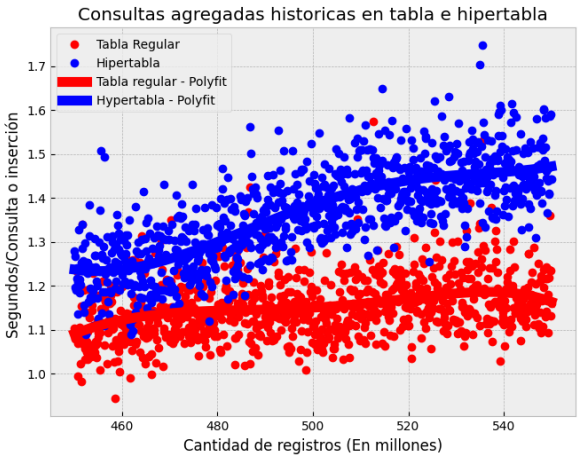

Fig 9 Consultas agregadas entre 450 y 550 millones de registros

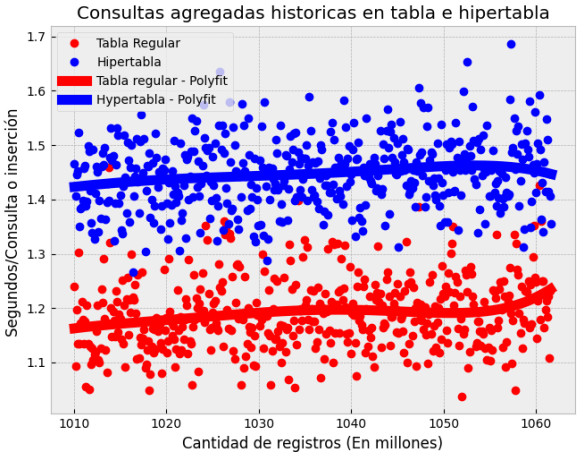

Fig 10 Consultas agregadas entre 1010 y 1060 millones de registros

Al igual que con las consultas históricas las figuras Fig 9 Consultas agregadas entre 450 y 550 millones de registros y Fig 10 Consultas agregadas entre 1010 y 1060 millones de registros muestran un mejor rendimiento sobre tablas regulares que sobre hipertablas.

V. CONCLUSIONES

Los resultados muestran que las inserciones son más eficientes en todos los casos sobre la hipertabla. Para las consultas el escenario es mixto con mejores rendimientos en uno u otro caso según la naturaleza de la consulta. Hasta la cantidad de registros simulados (1100 millones) ambas estrategias se mostraron estables, no hay una degradación marcada en ninguna de ellas ni tampoco rendimientos que ofrezcan una clara ventaja en esta instancia. Si bien las inserciones pueden ser un cuello de botella, por su naturaleza, el sistema espera recibir muchas consultas principalmente de carácter histórico. Este escenario no deja claro que una opción ofrezca mejores resultados generales por sobre la otra.

Con los datos obtenidos a la fecha se continuará trabajando sobre una versión postgresql estándar. No se descarta a futuro la migración a timescale pero la misma se considera injustificada según las simulaciones realizadas.

VI. REFERENCIAS Y BIBLIOGRAFÍA

A. Referencias bibliográficas:

[1] https://ugs.unlam.edu.ar, *UNLaM Ground Segment*, 2020.

[2] P. Soligo y J. S. Ierache, «Software de segmento terreno de próxima generación,» de *XXIV Congreso Argentino de Ciencias de la Computación (La Plata, 2018).*, 2018.

[3] P. Soligo y J. S. Ierache, «Segmento Terreno Para Misiones Espaciales de Próxima Generación,» *WICC 2019.*

[4] P. Soligo y J. S. Ierache, «Arquitectura de segmento terreno satelital adaptada para el control de límites de telemetría dinámicos,» de *XXV Congreso Argentino de Ciencias de la Computación (La Plata, 2019).*, 2019.

ISSN: 2525-1333. Vol.: 5 - Nro. 2 (DICIEMBRE-2020)

[5] https://gidsa.unlam.edu.ar, *Grupo de Investigación y Desarrollo de Software Aeroespacial de la Universidad Nacional de La Matanza,* 2020.

[6] D. J. White, I. Giannelos, A. Zissimatos, E. Kosmas, D. Papadeas, P. Papadeas, M. Papamathaiou, N. Roussos, V. Tsiligiannis y I. Charitopoulos, «SatNOGS: satellite networked open ground station,» 2015.

[7] D. Colley, C. Stanier y M. Asaduzzaman, «The Impact of Object-Relational Mapping Frameworks on Relational Query Performance,» de *2018 International Conference on Computing, Electronics & Communications Engineering (iCCECE)*, 2018.

[8] C. Russell, «Bridging the object-relational divide,» *Queue,* vol. 6, pp. 18-28, 2008.

[9] T. Morel, G. Garcia, M. Palsson y J. C. Gil, «High Performance Telemetry Archiving and Trending for Satellite Control Centers,» de *SpaceOps 2010 Conference Delivering on the Dream Hosted by NASA Marshall Space Flight Center and Organized by AIAA*, 2010.

[10] G. Pace, M. Schick, A. Colapicchioni, A. Cuomo y U. Voges, «EO ON-LINE DATA ACCESS IN THE BIG DATA ERA,» de *Proceedings of the 2019 conference on Big Data from Space*, 2019.

[11] S. K. Ibrahim, A. Ahmed, M. A. E. Zeidan y I. E. Ziedan, «Machine learning methods for spacecraft telemetry mining,» *IEEE Transactions on Aerospace and Electronic Systems,* vol. 55, pp. 1816-1827, 2018.

[12] J. Serra, «Relational databases vs Non-relational databases,» *Viitattu,* vol. 16, p. 2017, 2015.

[13] M. A. Mohamed, O. G. Altrafi y M. O. Ismail, «Relational vs. nosql databases: A survey,» *International Journal of Computer and Information Technology,* vol. 3, pp. 598-601, 2014.

[14] B. McBride y D. Reynolds, «Survey of time series database technology,» 2020.

[15] https://docs.timescale.com/, *TimescaleDB Documentation,* 2017.

[16] S. N. Pilar de Teodoro y J. Salgado, «Distributing big astronomical catalogues with Greenplum,» de *Proceedings of the 2019 conference on Big Data from Space*, 2019.

[17] E. Stefancova, «Evaluation of the TimescaleDB PostgreSQL Time Series extension,» 2018.

[18] https://www.timescale.com/, *TimescaleDB,* 2017.

[19] A. Jacobs, «The pathologies of big data,» *Queue*, vol. 7, pp. 10-19, 2009.

[20] A. B. M. Moniruzzaman y S. A. Hossain, «Nosql database: New era of databases for big data analytics-classification, characteristics and comparison,» *arXiv preprint arXiv:1307.0191*, 2013.

[21] G. Graefe y J. Alger, *Electronic database operations for perspective transformations on relational tables using pivot and unpivot columns,* Google Patents, 2001.

[22] blog.timescale.com, *Time-series data: Why (and how) to use a relational database instead of NoSQL,* 2017.

Revista Digital del Departamento de
Ingeniería e Investigaciones
Tecnológicas de la Universidad
Nacional de la Matanza

ISSN: 2525-1333. Vol.: 5 - Nro. 2 (DICIEMBRE-2020)

Recibido: 2020-11-16

Aprobado: 2020-11-28

Hipervínculo Permanente: http://www.reddi.unlam.edu.ar

Datos de edición: Vol. 5 - Nro. 2 - Art. 4

Fecha de edición: 2020-12-30