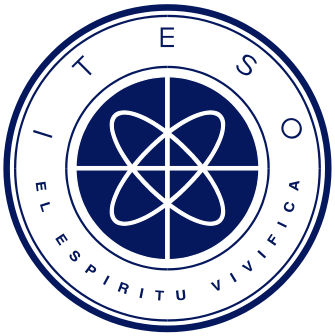
INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE OCCIDENTE

Departamento de Electrónica, Sistemas e Informática

MAESTRÍA EN SISTEMAS COMPUTACIONALES



IMPLEMENTACIÓN DE DATA LAKE EN LA NUBE PARA ANÁLISIS DE MOVILIDAD EN LA CIUDAD DE MÉXICO Y LA ZMG.

REPORTE DE TRABAJO DE OBTENCIÓN DE GRADO

Investigación, Desarrollo e Innovación 4

Presenta: Ing. Miguel Angel Ojeda Orozco

Director: Mtro. Moisés Valdovinos Toledo
Co-director: Dr. Iván Esteban Villalón Turrubiates

Tlaquepaque, Jalisco.25 de junio de 2020.

AGRADECIMIENTOS

El autor desea dar las gracias a todas aquellas personas involucradas en el desarrollo de este trabajo de obtención de grado. En primer lugar, me gustaría mencionar a mi familia, quién ha sido la base sólida sobre la cual he ido forjando mi camino durante toda mi vida y que siempre han estado conmigo para apoyarme en todas las decisiones que he tomado. Así mismo, un gran agradecimiento a todas aquellas personas que me han inspirado de una u otra forma a continuar desarrollándome y nunca permanecer en mi zona de confort, entre ellos me gustaría mencionar a mis ex compañeros de Oracle, quienes fueron los que me motivaron con su ejemplo a superarme y nunca poner límites, a mis tutores y profesores que siempre mantuvieron la mejor de las actitudes cuando de enseñanza se trató y, por supuesto, en especial al Mtro. Moisés Valdovinos quién, desde el día número uno, mostró una impecable actitud hacía mi proyecto.

De igual manera me gustaría agradecer al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico otorgado, mismo que fue entregado a través de la beca con número 640700 la cuál fue solicitada y aceptada con éxito.

Así mismo, extiendo un total agradecimiento a mi institución educativa, el Instituto Tecnológico y de Estudios Superiores de Occidente (ITESO) donde, con mucha felicidad y orgullo, llevé a cabo mis estudios, los cuáles, fueron apoyados por el descuento empresarial aplicado a mi colegiatura, el cuál ayudó con un porcentaje del pago de esta.

Sin más por el momento, dejo un extenso agradecimiento a todas y cada una de las personas que de una u otra forma apoyaron el desarrollo y entrega de este trabajo. La Dra. Mildreth y el Dr. Iván quienes fueron los coordinadores de la maestría durante mi tiempo de estudio, mis profesores, compañeros y personal académico sin los cuales este trabajo no podría haberse realizado y completado con éxito.

DEDICATORIA

El autor dedica esta tesis a ese motor que hace girar mi mundo día con día, mi familia. Desde el inicio de mis estudios, no ha existido un día en donde ellos no hayan estado presentes para mí y, con mucho orgullo y felicidad, hoy puedo darles una pequeña muestra de lo que su apoyo ha significado y el impacto que ha tenido en mi carrera y desarrollo profesional. Gracias por siempre hacer de mi mundo un lugar feliz.

Existe una palabra en el lenguaje japonés llamada "*ikigai"* la cuál en su traducción más cercana al español, podría traducirse como "la razón por la que te levantas todos los días" y, sin duda, cuando supe de la existencia de esta palabra, inmediatamente identifiqué como mi familia es mi ikigai sin dudarlo ni un segundo. Gracias por ser mi ikigai, gracias por darme las energías suficientes cuando se presentan días difíciles en mi vida. No habrá nunca nada como ustedes, los llevo en mi corazón y en mi alma.

RESUMEN

Se presenta una breve introducción al proyecto **data lake en la nube para análisis de movilidad en la Ciudad de México y la Zona Metropolitana de Guadalajara**. El cual tiene como objetivo principal generar una infraestructura que es creada con servicios de computación en la nube ofrecidos por *Amazon Web Services* con el fin de cargar, integrar, procesar y analizar datos de movilidad urbana registrados por los organismos que ofrecen dichos servicios, para de esta manera poder detectar y generar métricas y análisis que a su vez puedan dar soluciones y/o propuestas a los actuales problemas de movilidad que ambas ciudades presentan.

Es posible observar que la implementación del data lake trae consigo una gran mejora a los procesos comúnmente utilizados para llevar a cabo este tipo de análisis, pues se utilizan los datos de diferentes servicios de movilidad sin importar su estructura y tipo de almacenamiento, a fin de tener una integración total, generando un catálogo de metadatos el cual tiene como tarea principal el mapear los procesos de análisis con los datos almacenados físicamente en S3.

Finalmente, tras completar la fase de procesamiento de la información, se llevan a cabo diferentes tipos de análisis haciendo uso de tecnologías comunes, como lo son las consultas SQL, generación de dashboards y predicciones con ML, los cuales permiten confirmar y validar la importancia de tener una tecnología centralizada de análisis que sea capaz de producir resultados válidos sin tener que lidiar directamente con las diferencias estructurales inherentes de los datos utilizados.

TABLA DE CONTENIDO

1. INTRODUCCIÓN 10

1.1. ANTECEDENTES 10
1.2. JUSTIFICACIÓN 11
1.3. PROBLEMA 12
1.4. OBJETIVOS 13
1.4.1. Objetivo General: 13
1.4.2. Objetivos Específicos: 13
1.5. NOVEDAD CIENTÍFICA, TECNOLÓGICA O APORTACIÓN 14

2. ESTADO DEL ARTE O DE LA TÉCNICA 15

2.1. ESTADÍSTICAS DE MOVILIDAD EN MÉXICO 16
2.2 ESTADÍSTICAS DE MOVILIDAD EN LA ZMG Y LA CDMX 20
2.3 IMPACTO SOCIAL DE LA MOVILIDAD EN MÉXICO 24
2.4 CONDICIONES POLÍTICAS DE PROGRAMAS DE MOVILIDAD EN MÉXICO 27

3. MARCO TEÓRICO/CONCEPTUAL 28

3.1. MODELADO DE DATOS 28
3.2. PROCESAMIENTO DE DATOS 28
3.3. BASES DE DATOS 29
3.4. SQL 29
3.5. DATA WAREHOUSE 30
3.6. ETL 30
3.7. DATA LAKE 31
3.8. ELT 32
3.9. APACHE HADOOP 33
3.9.1. HADOOP DISTRIBUTED FILE SYSTEM (HDFS) 34
3.9.2. MAPREDUCE 36
3.9.3. YARN 37
3.9.3.1. RESOURCEMANAGER 37
3.9.3.2. NODEMANAGER 38
3.9.3.3. APPLICATIONMASTER 38
3.9.3.4. CONTAINER 38
3.9.4. HIVE 39
3.9.5. SPARK 40
3.10. AMAZON WEB SERVICES 42
3.10.1. S3 43
3.10.2. AMAZON GLUE 44
3.10.3. AMAZON EMR 45

4. DESARROLLO METODOLÓGICO 46

4.1. EXTRACCIÓN DE INFORMACIÓN 46
4.2. ARQUITECTURA DEL PROYECTO 48
4.3. IMPLEMENTACIÓN EN AWS 50

4.3.1. S3 50
4.3.2. GLUE 52
4.3.3. DATA CATALOG (LAKE FORMATION) 55
4.3.4. ATHENA 56
4.3.5. ELASTIC MAPREDUCE 57
4.3.6. REDSHIFT 62
4.4. ANÁLISIS DE LA INFORMACIÓN CON QUICKSIGHT 64
4.5. ANÁLISIS DE LA INFORMACIÓN CON MACHINE LEARNING 66
5. RESULTADOS Y DISCUSIÓN 70
5.1. RESULTADOS 70
5.2. DISCUSIÓN 78
6. CONCLUSIONES 79
6.1. CONCLUSIONES 79
6.2. TRABAJO FUTURO 81

LISTA DE FIGURAS

Figura 1. Número de Automóviles en circulación en el 2018 16
Figura 2. Porcentaje de población que se desplaza al trabajo en bicicleta 19
Figura 3. Viajes realizados en bicicleta durante el año 2018 en la ZMG 20
Figura 4. Porcentaje de tiempo adicional en las 25 ciudades más congestionadas del mundo 22
Figura 5. Datasets sobre la ubicación de las estaciones del programa Mi Bici en la ZMG 47
Figura 6. Ejemplo de almacenamiento de una estación de bici en la CDMX 47
Figura 7. Arquitectura general del proyecto 48
Figura 8. Nivel base de estructuración de carpetas en S3 para almacenamiento de datasets 50
Figura 9. Configuración de propiedades para datasets cargados. 51
Figura 10. Proceso de carga de archivos 51
Figura 11. Base de datos en Glue para almacenar la información sobre los datasets cargados en S3 52
Figura 12. Extracto de script utilizado para la generación de tablas de metadatos en AWS Glue 52
Figura 13. Configuración de parámetros relacionados al crawler que procesa archivos tipo CSV 53
Figura 14. Fases de ejecución del crawler 54
Figura 15. Consola de monitoreo para ejecuciones de crawlers 54
Figura 16. Bases de datos creadas en Lake Formation 55
Figura 17. Tablas creadas por los jobs de ETL ejecutados en los crawlers 55
Figura 18. Consulta de información en Athena de las tablas creadas con Glue 56
Figura 19. Extracto de código utilizado en el procesamiento de archivos .JSON 57
Figura 20. Configuración del software a instalar en el clúster 59
Figura 21. Configuración general del clúster creado en EMR 60
Figura 22. Instancias creadas en EC2 por EMR 60
Figura 23. Ejecución del clúster en tiempo real por EMR 61
Figura 24. Role en IAM creado para la completa integración de Redshift 62
Figura 25. Configuración de nodos del warehouse 63
Figura 26. Consola de administración del clúster 63
Figura 27. Consola de configuración de Amazon QuickSight 64
Figura 28. Parte de script utilizado para llevar a cabo la integración de datos 66
Figura 29. Extracto de script generado para aplicar regresión lineal a datos de MiBici 68
Figura 30. Arquitectura utilizada en AWS 71
Figura 31. Ejemplo de query ejecutada en Amazon Athena 72
Figura 32. Número de usuarios de UBER en el primer trimestre del 2019 en la CDMX 73
Figura 33. Regresión lineal aplicada para el número de usuarios del programa MiBici en el 2019 75
Figura 34. Comparativa de valores reales vs valores calculados 75
Figura 35. Estimación de usuarios activos del programa MiBici en la ZMG 76
Figura 36. Regresión lineal y predicción de usuarios de Uber en la CDMX y la ZMG 76

LISTA DE TABLAS

Tabla 1. Comparación de viajes realizados en bicicleta en la ZMG durante el año 2019. 21
Tabla 2. Resultados de encuesta de satisfacción a usuarias del programa de bici pública en Jalisco..... 25
Tabla 3. Gasto anual por utilización de plataformas de taxis ejecutivos 26
Tabla 4. Links de descarga de información 46
Tabla 5. Comparativa de motores de procesamiento utilizados 74

LISTA DE ACRÓNIMOS Y ABREVIATURAS

DA	Data Analytics
CA	Cloud Analytics
AWS	Amazon Web Services
WRI	World Resources Institute
DW	Data Warehouse
DL	Data Lake
OS	Open Source
ZMG	Zona Metropolitana de Guadalajara
CDMX	Ciudad de México
JSON	JavaScript Object Notation
ML	Machine Learning
EMR	Elastic MapReduce
AZ	Availability Zone
SSD	Solid State Disk
ML	Machine Learning
LR	Linear Regression

1. INTRODUCCIÓN

Resumen: *En este capítulo se presenta brevemente los antecedentes del objeto de estudio, justificación del objeto de estudio, justificación y la definición del problema.*

1.1. Antecedentes

Históricamente hablando, el 90% de los datos que existen en el mundo han sido creados en los últimos años y el término "Big Data" se ha utilizado desde principios de los años 90s. Aunque no es exactamente conocido quién fue la primera persona en utilizarlo, es claro que, en esencia, el término no es algo que sea completamente nuevo o algo de las últimas dos décadas. Por el paso de los siglos, diferentes culturas han estado intentando utilizar el análisis de datos y técnicas de análisis para apoyar su proceso de toma de decisiones. Los antiguos egipcios alrededor del año 300 A.C. ya habían capturado toda la información existente en la librería de Alejandría. Además, el imperio romano solía utilizar cuidadosamente el análisis estadístico de su ejército para determinar la distribución óptima de sus tropas.

Sin embargo, en las últimas dos décadas, el volumen y la velocidad con la que los datos se han generado ha cambiado más allá de la compresión humana. Un aproximado de la cantidad total de datos existentes en el mundo para el año 2013 es de 4.4 zettabits de información, para este año 2020, el total es de 44 zettabits. Para poner estos números en perspectiva, 44 zettabits es equivalente a 44 trillones de gigabytes, aún y con la más avanzada tecnología, es imposible poder analizar toda esta información. La necesidad de procesar y analizar estos datos crecientes (y no estructurados) estableció como el análisis tradicional de información se transformó a "Big Data" en la última década.

Desde el inicio de nuevo milenio, el internet y la web empezaron a ofrecer colecciones de datos únicos y oportunidades para analizarlos. Con la expansión del tráfico web y ventas en línea, compañías como Yahoo, Amazon y eBay empezaron a analizar el comportamiento del cliente analizando la frecuencia de clics, ubicación de su dirección IP y búsqueda de *logs*. Aunque el contenido no estructurado de la web es todavía el objetivo principal para muchas organizaciones en el análisis de datos y big data, las posibilidades de obtener información valiosa está emergiendo desde los dispositivos móviles.

Todo esto nos permite ver que el big data no es un fenómeno nuevo y aislado, sino que es parte de una larga evolución de captura y uso de datos. Como cualquier otro desarrollo en el almacenamiento de datos, procesamiento de datos y el Internet, big data es un paso más que traerá cambios a la forma en que se manejan los negocios y, por consiguiente, la sociedad. Al mismo tiempo, sentará las bases sobre las que se construirán muchas evoluciones en nuestra sociedad.

1.2. Justificación

Hoy en día, con el uso de la tecnología, distintos organismos tienen la capacidad de obtener información detallada de cada punto del usuario que hace uso de sus servicios. Esta información puede incluir el uso de aplicaciones móviles, “clics” digitales, interacciones en redes sociales y mucho mas, todo contribuyendo a una huella digital que es única de cada usuario. Sin embargo, en algún punto no muy lejano del tiempo, el pensamiento de que los usuarios compartieran información como la hora en la que se levantan, a qué hora desayunan, o a donde van de vacaciones, hubiera sido una consideración bastante bizarra por decir lo menos.

Aún y que el big data se puede considerar como un desarrollo reciente, el interés por implementar la computación en este servicio se ha vuelto un interés que continúa creciendo dentro de las empresas y organismos que conforman nuestra sociedad.

Las normas sociales del usuario ciertamente han cambiado con el uso de la tecnología, lo que ha permitido que dichos organismos puedan tener información detallada de sus usuarios de una manera legal, fidedigna y confiable, para así, poder obtener datos a manera que puedan ser utilizados como retroalimentación sobre los servicios que ofrecen, con el fin de poder identificar y mejorar las distintas áreas de oportunidad que puedan existir después de dicho análisis. Sin embargo, el hecho de que se tenga la oportunidad de acceder a esta información no implica necesariamente que se tengan las herramientas, métodos y conocimiento para hacerlo, por lo que en la gran mayoría de los casos, las empresas y organismos no tienen la capacidad de poder hacer un correcto uso de ella.

Un pensamiento compartido dentro de la sociedad es que para poder acceder, transformar y cargar todos estos datos con el fin de analizarlos, requerirá una gran inversión y tiempo que, en la gran mayoría de las veces, no se cuenta o no se tiene considerado para este tipo de proyectos. Sin embargo, el uso de herramientas y métodos de big data en conjunto con la nube, han proporcionado una opción sumamente confiable y económica para poder llevar a cabo todo este proceso de extracción, transformación y análisis de información sin tener que realizar una gran inversión de tiempo y dinero.

Un ejemplo claro y fundamental, es el aspecto de movilidad dentro de las dos ciudades más grandes e importantes del país (la Ciudad de México y Guadalajara), las cuales presentan índices sumamente altos respecto al tiempo de traslado de los ciudadanos entre dos o más puntos. Ambas ciudades cuentan con una gran cantidad de información proporcionada por parte de distintas organizaciones como Uber y los programas de bici que cada ciudad ofrece; en donde se recaba información sobre el traslado de sus ciudadanos, mismo que se encuentra disponible para su uso abierto y el público en general

Con todo esto dicho, es que se justifica el propósito de este trabajo, el cual se basa en utilizar los registros generados por los usuarios de estos programas, a fin de poder ser procesados y analizados utilizando la computación en la nube para poder identificar y resolver las actuales necesidades de movilidad que los usuarios han expresado de acuerdo con su experiencia.

1.3. Problema

La implementación y uso de los programas de movilidad tiene como propósito principal el ofrecer a la ciudadanía, una opción alterna de desplazamiento mucho más económica y, en muchas ocasiones, mucho más efectiva. Estos programas son ubicados normalmente dentro de las zonas con mayor importancia y afluencia, haciendo usos de renta de bicicletas (disponibles en estaciones dispuestas a manera de red en las centralidades urbanas más importantes de la ciudad) y/o taxis ejecutivos. Cada sistema cuenta con distintas horas de operatividad, ajustándose a los horarios de otros sistemas de transporte masivo, mismos que recaban información de todos los viajes realizados por los usuarios.

De igual manera Uber, la empresa de transporte de los llamados “taxis ejecutivos”, ofrece sus servicios en ambas ciudades, recabando de igual manera, toda la información disponible de todos los viajes realizados por los usuarios, mismos que se encuentran disponibles en su página de internet.

Ambas ciudades han presentado ciertas problemáticas respecto a los tiempos de traslado tal y como los usuarios lo han expresado en las distintas redes sociales, organismos y centros de atención donde se trabajan este tipo de inconformidades. Analizando a detalle dichas problemáticas, se puede identificar que no se logra satisfacer las necesidades de movilidad que se tienen dentro de la sociedad por una carencia de análisis dentro de los servicios que se ofrecen. Citando un ejemplo, algunos usuarios no tienen la oportunidad de hacer uso del servicio de bicicleta por falta de unidades, mientras que, en otras estaciones, la existencia de unidades rebasa el permitido sin tener una necesidad real de que sea así.

En lo que respecta al uso de automóviles para el traslado del ciudadano, en México, más del 80% de la población vive en las ciudades, esto es un porcentaje considerablemente mayor al promedio de los otros países en el mundo. Esto se traduce en que, la población supera con creces la capacidad de transporte por lo que los tiempos de traslado superan la media del mundo.

1.4. Objetivos

1.4.1. Objetivo General:

El presente trabajo pretende generar un infraestructura tipo *Data Lake* la cuál será creada con servicios de computación en la nube ofrecidos por Amazon Web Services con el fin de cargar, integrar, procesar y analizar datos de movilidad urbana, motorizada y no motorizada, registrados por los organismos que ofrecen dichos servicios (mismos que se encuentran abiertos al público en general sin ningún costo) dentro de la zona metropolitana de Guadalajara y de la Ciudad de México con el fin de detectar y generar métricas y patrones por medio Machine Learning para de esta manera dar soluciones y/o propuestas a los actuales problemas de movilidad que ambas ciudades presentan día con día.

1.4.2. Objetivos Específicos:

1. Obtener los datos de los programas, registrados automáticamente por el uso de los servicios, teniendo detalles del usuario, tiempos, así como la ruta utilizada para su desplazamiento.

2. Carga y limpieza de dichos datos con un proveedor de servicio de nube (Amazon Web Services) para de esta manera poder realizar el análisis sin tener la necesidad de almacenar localmente dicha información.

3. Creación de una infraestructura de tipo "Data Lake" con el fin de generar un proceso establecido y definido el cual podrá tener la capacidad de procesar la información generada en distintos formatos de una manera confiable y segura.

4. Procesamiento de la información almacenada utilizando servicios de DA como EMR (Elastic Map Reduce), Amazon Athena, Amazon QuickSight, entre otros, con el fin de obtener información que pueda ser usada por modelos de Machine Learning para la generación de posibles *insights*.

5. Generación de análisis, reportes estadísticos y patrones a partir de la información procesada con el fin de visualizar de una manera mucho más clara como es el uso actual de los programas y su impacto dentro de la movilidad de ambas ciudades, así como sus principales características definidas por el uso continuo del usuario.

6. Creación de una infraestructura auto escalable donde se pueda mantener un registro histórico de la información a fin de que todos los nuevos datos que sean generados puedan ser igualmente almacenados y analizados automáticamente.

7. Implementación de una tecnología como *Terraform* que permita crear toda la infraestructura en la nube a manera de código.

1.5. Novedad científica, tecnológica o aportación

El desarrollo del presente trabajo pretende ofrecer una solución completa, escalable y segura al problema de movilidad que la ciudad metropolitana de Guadalajara y la Ciudad de México presentan a través de una infraestructura que será capaz de almacenar, integrar procesar y analizar los datos a fin de generar posibles soluciones a partir de un análisis mucho más profundo que será posible gracias a la integración de toda la información existente.

Dicha solución se pretende lograr a través del uso de los últimos servicios de computación en la nube creados y ofrecidos por AWS, mismos que han demostrado ser confiables, seguros y escalables. De esta manera, se podrá tener las herramientas necesarias para resolver una problemática social actual de ambas ciudades utilizando servicios de última tecnología con una infraestructura segura y auto escalable que podrá ser usada y entendida por cualquier persona autorizada con un costo relativamente bajo.

La solución presentada en este trabajo podría ser generada por otros métodos que sin duda alguna involucrarían la contratación de personal y recursos que muchas veces no se tienen considerados dentro del presupuesto de los programas de movilidad por parte de las instituciones encargadas de tratar temas de movilidad. Por tal motivo, a fin de satisfacer las problemáticas actuales sin alterar la planeación y presupuesto, se ofrece una solución completa, segura y de bajo costo que puede ser implementada con total seguridad y fiabilidad a fin de resolver necesidades actuales, así como futuras de una manera casi automática.

2. ESTADO DEL ARTE O DE LA TÉCNICA

Resumen: *En este capítulo se presenta un resumen de los trabajos relacionados con los problemas de movilidad implementando soluciones dentro del campo del Big Data, así como su impacto dentro de la sociedad en la zona metropolitana de Guadalajara y la Ciudad de México*.

Al adentrarse en los temas relacionados a problemas de movilidad, dentro de las dos ciudades más grandes de este país como lo son la zona metropolitana de Guadalajara y la Ciudad de México, es posible identificar un gran número de fuentes de información que tratan y discuten distintos aspectos que forman parte o afectan en cierto modo a este tema.

El presente trabajo pretende abordar el tema desde una perspectiva diferente a la que normalmente se utiliza para hablar sobre esta problemática. Tal y como se ha expresado anteriormente, el objetivo principal de este trabajo es el de poder implementar una arquitectura completamente basada en aspectos fundamentales del Big Data utilizando servicios de computación en la nube para llevar a cabo un almacenamiento, integración y procesamiento de diferentes datos registrados por usuarios que se desplazan a través de estas dos ciudades en medios de transporte motorizados, por medio de programas de bicicletas implementados por el gobierno, y no motorizados como lo son el servicio de taxis ejecutivos que Uber pone a disposición de la ciudadanía.

Durante la exploración de esta problemática cada vez más fuerte y buscando información que sea objetiva, ajustada a la realidad, sensata y plausible de desarrollar, se procedió a tratar e investigar la literatura existente, buscando antecedentes de estudios, investigaciones, reportajes o artículos similares o relacionados que permitan construir una base sólida para el desarrollo del presente trabajo de obtención de grado, de modo que las conclusiones y resultados obtenidos al final de este sean fiables y seguros, permitiendo la generación de conocimiento y valor agregado para todas las entidades interesadas en esta problemática común.

Consultando la información y bibliografía reciente sobre los programas y servicios de los que el ciudadano hace uso en su día a día, es posible identificar y clasificar la documentación existente dentro de diferentes categorías, (mismas que serán descritas a detalle en esta sección) las cuales tienen diferentes enfoques. Mientras que por un lado existen trabajos que enfocan su investigación desde el lado estadístico, hay otros que lo abordan desde el aspecto social, tratando de generar características que describen como este tema tiene un impacto dentro de la sociedad y como la sociedad tiene un impacto dentro de su continuo desarrollo.

2.1. Estadísticas de movilidad en México

El primer aspecto por tratar es el que está relacionado al tema estadístico. Tal y como ya se ha mencionado previamente, la mayoría de los estudios y trabajos actuales sobre los programas y servicios de movilidad son aquellos que tienen como principal enfoque la generación de estadísticas, las cuales, en su mayoría, sirven como una herramienta de medición.

Para el mes de octubre del año pasado, de acuerdo con las estadísticas relativas a los vehículos de motor en México, el Instituto Nacional de Estadística y Geografía (INEGI) tiene registros administrativos de 32 milllones 291 mil 454 automóviles en circulación en todo el país. De éstos, 31.53 millones son automóviles de uso particular; 698 mil 595 son vehículos de uso público de transporte de pasajeros o utilitarios de empresas, y 58 mil 255 son vehículos "oficiales", es decir, forman parte de la flota de instituciones públicas, ya sean federales, estatales o municipales.

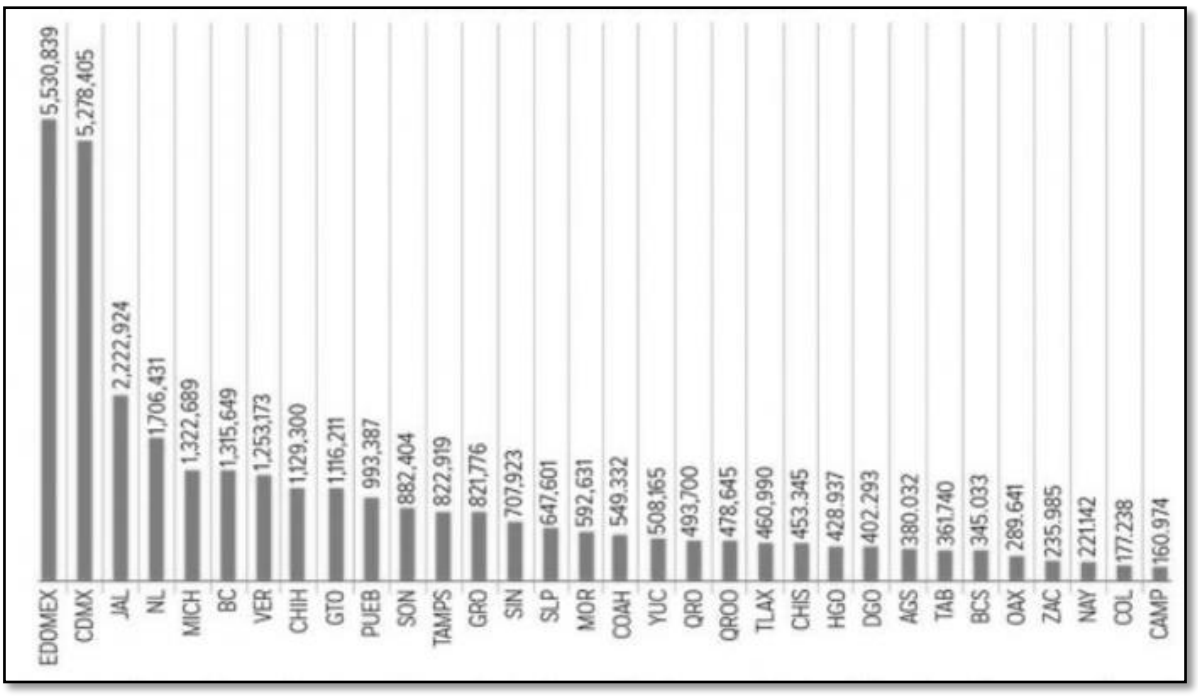


Figura 1. Número de Automóviles en circulación en el 2018

En este estudio, el grupo Imagen identifica como de acuerdo con el propio INEGI, hay nueve entidades en el país que cuentan con parques vehiculares que superan el millón en cada una de ellas; éstas son:

1. **Ciudad de México:** 5,278,405 vehículos
2. **Jalisco:** 2,222,924 vehículos
3. **Nuevo León:** 1,706,431 vehículos
4. **Michoacán:** 1,322,689 vehículos
5. **Baja California:** 1,315,649 vehículos
6. **Veracruz:** 1,253,173 vehículos
7. **Chihuahua:** 1,129,300 vehículos

Con estos datos, lo que se observa es que dos de cada tres automóviles registrados en circulación en el país, se concentran en estas entidades.

Es interesante observar que las entidades con mayor número de automóviles no son necesariamente las que tienen un mayor número de estos vehículos para el servicio público.

El uso de automóviles para transporte público de pasajeros no puede desvincularse de los desastrosos servicios de transporte público colectivo, y el inmenso desorden urbano que existe en todo el país. Lo anterior encuentra un indicador en el número de autobuses para transporte público que hay en México, pues frente a los más de 698 mil automóviles de uso público que hay en el territorio nacional, se dispone de 432 mil 240 autobuses para transporte de pasajeros, y en ellos se incluyen los de transporte federal. [1]

De igual manera, tal y como ya se habló de las ciudades con más problemas de movilidad, es de suma importancia hablar de aquellas que, por el contrario, presentan el menor índice, en cuanto a esta problemática se refiere. De esta manera, la revista Forbes (en su edición mexicana) realizó un estudio de las ciudades con mejor movilidad en México.

En este artículo se muestra el resultado general de la primera herramienta que mide la movilidad en ciudades mexicanas de manera integral, el Índice de Movilidad Urbana (IMU), elaborada por el Instituto Mexicano de Competitividad (IMCO).

Las ciudades "verdes" son las que tienen mejor desempeño en producción y bienestar social, tasas de crecimiento 2.3 veces más rápidas y salarios más altos. El salario promedio en estas ciudades es de 7,472 pesos al mes.

En cambio, las "rojas" son las que se percibe una tasa de homicidios cuatro veces mayor por cada 100,000 habitantes que en las que tienen mejor movilidad y salarios más bajos. En promedio, el salario mensual en estas localidades es de 6,165 pesos al mes. [2]

"Esto son señales para autoridades y ciudadanos de que la agenda se movilidad es una prioridad", afirma Fátima Masse, investigadora del IMCO. Destaca en ese sentido que las entidades con 10 mil o más unidades de este tipo son:

1. **Coahuila**, con 98 mil 685
2. **Guerrero**, con 58 mil 335
3. **Estado de México**, con 54 mil 472
4. **Ciudad de México**, con 30 mil 901
5. **Guanajuato**, con 29 mil 354
6. **Michoacán**, con 13 mil 678
7. **Veracruz**, con 13 mil 126
8. **Puebla**, con 11 mil 061.

De igual manera, es posible observar como la cobertura de MiBici Pública se extendió en diciembre del 2018 en donde se sumaron 38 estaciones y se ampliaron 24 ya existentes por la alta demanda.

En el 2018 la Secretaría de Desarrollo Agrario, Territorial y Urbano del Gobierno Federal publicó el libro Anatomía de la movilidad en México, en el que se analizan diversos aspectos de movilidad en el país en el periodo 2013-2018. El libro presenta datos relevantes que permiten comprender de manera más amplia algunas de las implicaciones de los patrones de movilidad en México.

¿Cómo nos movemos? En México, 55 por ciento de las personas que van a la escuela realizan sus viajes caminando; 26 por ciento en autobús, taxi, combi o colectivo; 18 por ciento en vehículo particular y uno por ciento en bicicleta. Asimismo, 35 por ciento de las personas que van a trabajar realizan sus viajes en autobús, taxi, combi o colectivo; 28 por ciento en vehículo particular; 23 por ciento caminando y 5 por ciento en bicicleta. Lo anterior sugiere entonces que la mayoría de los viajes a la escuela y al trabajo en el país se realizan caminando o en transporte público, y no necesariamente en vehículo particular, aunque la inversión en infraestructura vehicular es desproporcionadamente mayor, e incluso se estima que el espacio destinado para la circulación vehicular representa cerca del 40 por ciento de la superficie urbanizada del país.

¿Cuál es el estado de la seguridad vial? La manera en que nos movemos está costando vidas humanas; en 2016 en México murieron 16,185 personas en hechos de tránsito. Las principales víctimas son usuarios vulnerables de la vía pública: 44 por ciento de las muertes eran peatones; 34 por ciento ocupantes de vehículos; 20 por ciento motociclistas; y 2 por ciento ciclistas. Estos datos demuestran que las personas más vulnerables de la vía pública, o sea, los peatones y ciclistas, representan la mayor proporción de los decesos en las calles, es importante mencionar que las muertes por hechos de tránsito son siempre prevenibles, y, por lo tanto, las cifras anteriores no se deberían de normalizar. [3]

Las estadísticas oficiales nos dicen que la mayor parte de los viajes en México son caminando y en transporte público. Sin embargo, la preocupación por la movilidad en la esfera pública se ha concentrado en la forma alarmante del crecimiento del parque vehicular privado en México, y no en las condiciones en las que ofrecemos el transporte público o la seguridad de las redes peatonales. En el último reporte del INEGI, durante los últimos diez años, el parque de vehículos motorizados privados se ha incrementado un 5.4 por ciento, mientras que aquellos destinados al transporte de pasajeros han aumentado tan sólo un 2.37 por ciento, en promedio. El espacio público dedicado a la circulación de estos vehículos es, aproximadamente, del 40 por ciento de la superficie urbanizada, siendo así la infraestructura para el movimiento de vehículos motorizados a la que se destina la mayor parte del gasto público y prioridad por aumentar la capacidad vial, lo que lleva a impactos graves en la salud humana y urbana. A pesar de lo alarmante de las cifras de crecimiento vehicular, la mitad de la población en México usa el transporte público concesionado y cerca del 30 por ciento de las personas hace sus traslados a pie.

En el país, 17 millones 925 mil 774 personas se trasladan a la escuela caminando. El estado con el primer lugar, con 73.9 por ciento de la población de 3 años y más que se desplaza a su la escuela caminando, es Chiapas. Resalta la posición de la Ciudad de México entre los cinco estados con menor cantidad de población que se desplaza a pie, con el 41.4 por ciento, un 13.5 por ciento menos que la media nacional, por lo que comparte dinámicas similares a los estados del norte del país. Esta gráfica es totalmente opuesta cuando se analizan los desplazamientos en automóvil. Además, 9 millones 192 mil 364 personas se trasladan al trabajo caminando. Chiapas mantiene el liderazgo en esta modalidad, estando 25.5 puntos porcentuales arriba de la media nacional. Por su lado, Nuevo León tiene los últimos lugares, con 12.2 puntos porcentuales abajo de la media nacional.

No cabe duda de que, en los últimos años, el impulso a la movilidad en bicicleta ha aumentado de manera considerable en todo el país. Esto se nota en las áreas institucionales dedicadas a las políticas de movilidad no motorizada, al aumento de los proyectos de infraestructura que logran financiamiento local y federal, así como en las acciones de cultura social en favor de calles compartidas. Un resultado interesante es que, en México, existe un mayor porcentaje de población que se traslada al trabajo en bicicleta que para ir a estudiar.

Se observa que, en el sureste del país, específicamente en los estados de Yucatán, Campeche y Quintana Roo, existe una cultura del uso de la bicicleta como un modo de transporte sustentable para ir a la escuela o al trabajo, incluso siendo estados con clima cálido-húmedo la mayor parte del año.

En el país, 488 mil 678 personas utilizan la bicicleta como medio de traslado para ir a sus centros de enseñanza o de estudio. Mientras Yucatán tiene el mayor porcentaje de población de 3 años y más que asiste a la escuela en bicicleta, en el estado de Nuevo León sólo el 0.29 por ciento usa este medio de transporte para ir a estudiar. Dos millones 196 mil 854 personas utilizan la bicicleta como medio de traslado para ir a su trabajo. El estado de Guanajuato tiene el mayor porcentaje de población ocupada que se traslada de esta forma, con un 14.41 por ciento. Mientras que en Jalisco y la Ciudad de México se encuentran el lugar 16 y 29 respectivamente aún y que son las dos ciudades con mayor densidad demográfica.

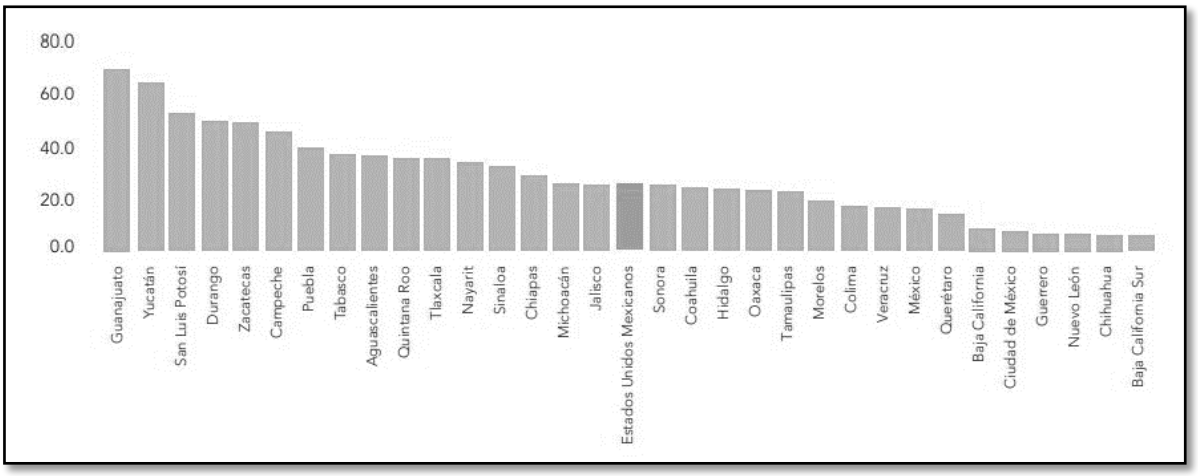


Figura 2. Porcentaje de población que se desplaza al trabajo en bicicleta

2.2 Estadísticas de movilidad en la ZMG y la CDMX

Hablando más a detalle sobre las dos ciudades principales en las cuáles se enfoca este trabajo, se puede fácilmente identificar los trabajos realizados hasta el día de hoy con respecto al tema de movilidad dado que no hay una gran variedad de ellos. Como primer tema a tratar, se habla sobre el programa de Bici pública que el Gobierno de Jalisco en conjunto con los gobiernos municipales puso a disposición de la ciudadanía de la ZMG.

Para finales del mes de enero el programa tenía una red de 274 estaciones y dos mil 446 bicicletas que cubrían una extensión de mil 100 hectáreas en los municipios de Guadalajara, Zapopan y Tlaquepaque.

Dado que parte del actual trabajo pretende generar resultados con los datos históricos del programa y no solo con los más recientes, es posible encontrar una investigación realizada por de 3 empleados del Centro de Desarrollo de México de la empresa Oracle. Esta investigación muestra un procesamiento de datos sobre el año 2018 en donde se obtuvieron características implícitas de la información generada por los usuarios en ese año.

Esta investigación tenía como principal propósito el de generar modelos que pudieran predecir la duración y el destino de cada nuevo viaje utilizando información histórica para poder entrenar dichos modelos. En este proceso se muestran las distintas fases utilizadas, mismas que se listan a continuación:

1. Exploración de información
2. Limpieza de datos
3. Ingeniería de características
4. Entrenamiento de modelos

Al consultar algunos de los resultados preliminares de dicho trabajo, es posible identificar que durante el año 2018 se realizaron un total de 3,400,000 viajes a través de 25,643 usuarios utilizando 274 estaciones, lo que generó un aumento del 35% respecto al 2017. En el siguiente gráfico generado por este grupo de ingenieros se puede observar los patrones de los viajes realizados en el 2018.

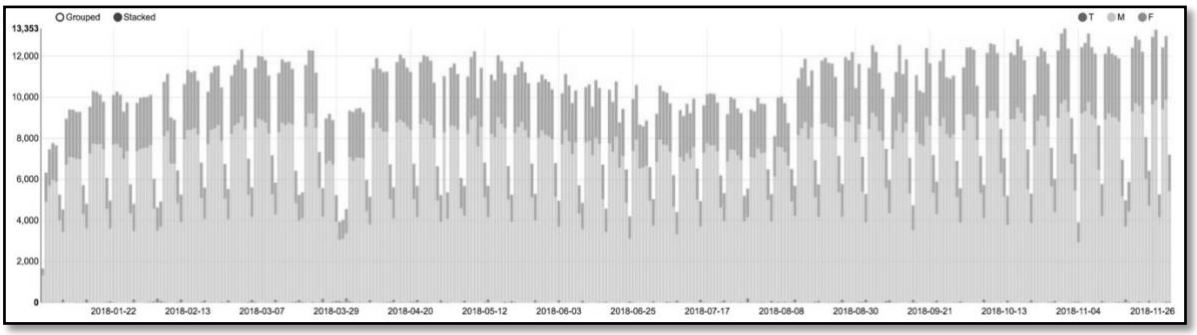


Figura 3. Viajes realizados en bicicleta durante el año 2018 en la ZMG

Otro aspecto importante considerado en este trabajo es aquel que habla sobre las características de los usuarios mismos que hacen uso del programa, ya que, tal y como se muestra en dicho estudio, hay ciertas características entre los usuarios que afectan de una manera directa el programa.

El procesamiento de datos en este trabajo se llevó a cabo utilizando Python como el lenguaje de programación, implementado la librería “Pandas” así como Spark para la limpieza de información así como la generación de características.

El sábado 1 de diciembre de 2019, el sistema de bicicletas públicas del Área Metropolitana de Guadalajara, Mi Bici, cumplió 5 años de operación y con los datos públicos que ofrecen en su sitio web y el levantamiento de una nueva encuesta de percepción, fue posible conocer las características de los usuarios y sus hábitos de movilidad. El Instituto de Recursos Mundiales México (WRI México, por sus siglas en inglés) en conjunto con el Instituto Metropolitano de Planeación del Área Metropolitana de Guadalajara (IMEPLAN) y la empresa operadora del sistema BKT Bicicleta Pública, han desarrollado una primera investigación y análisis sobre los impactos y beneficios de movilidad, salud pública y calidad del aire, entre otros.

Analizando la información publicada por Iván De la Lanza y Diana Amezola en el portal de WRI México, es posible obtener un número estadístico exacto sobre el uso del programa MiBici en el año 2019. Dentro de estos datos cabe resaltar que para el último mes del año se tenía reportado un total de 2,446 bicicletas en las mismas 274 estaciones con un total de 73,446 usuarios históricos registrados. Comparando el número de usuarios y viajes realizados durante el mes de enero entre el año 2018 y 2019, es posible observar un aumento significativo en el uso del programa. [4]

Mes	Enero	Febrero
Viajes mensuales	198,653	405,169
Usuarios registrados	11,841	17,453

Tabla 1. Comparación de viajes realizados en bicicleta en la ZMG durante el año 2019.

Sin duda alguna que es importante considerar factores sociales externos al programa que pudieron haber tenido un efecto directo al incremento registrado, como lo fue el que provocó el desabasto de gasolina en el estado de Jalisco (y otros más) debido a decisiones tomadas por el gobierno federal para combatir el robo de combustible en el país. Dicha decisión provocó un desabasto generalizado en varias zonas de la republica que provocó que una gran cantidad de ciudadanos recurriera a vías alternas para su desplazamiento. Sin embargo, aún y teniendo en cuenta aspectos como el ya mencionado, también es real que dicho número no se redujo en gran medida para los meses subsecuentes. Por ejemplo, en febrero del mismo año (2019) se registró un total de 388,993 viajes con un total de 17,465 usuarios registrados. Si bien el número de viajes disminuyó en comparación con el mes pasado, el registro de usuarios tuvo un comportamiento contrario en donde se sumaron al programa un total de 12 nuevos usuarios aún y que, para el mes de febrero, el desabasto de combustible había terminado y su consumo había regresado a la normalidad.

Con respecto a la ciudad de México, donde hay poco más de dos millones de carros y circulan a diario poco más de un millón, con un transporte público colapsado y caro, la bicicleta es el transporte público más barato y el que se desplaza a mayor velocidad en horas pico.

La mitad de los viajes en bicicleta en esta urbe son de menos de 8 kilómetros y duran menos de 20 minutos, y, en promedio, la bicicleta se usa entre 25 y 35 minutos. Las distancias en la ciudad son muy largas; por eso nadie sugeriría transportarse en bicicleta desde lugares como La Villa hasta Tlalpan. Para ese tipo de viajes, el "Plan de movilidad en bicicleta para la Ciudad de México" propone no solo incrementar y mejorar la infraestructura, sino también alternar la bicicleta con otros medios de transporte.

Actualmente, del total de la movilidad vial en la Ciudad de México, el transporte en bicicleta representa 2.5% (poco menos de 300 mil viajes); hace 10 años representaba 1% (98 mil viajes). Con la duplicación de la infraestructura vial ciclista y una adecuada promoción se espera que los viajes en bicicleta se dupliquen en 2024. La bicicleta es una opción viable, eficiente, de bajo costo y rápida en distancias cortas, que contribuye a mejorar la movilidad y a hacer de las ciudades espacios sustentables. [5]

En los últimos 30 años en México, el crecimiento de la población y la mala planificación urbana han provocado el congestionamiento de las grandes urbes, ocasionando que el traslado de un punto a otro dentro de la ciudad y hacia las periferias se convierta en un recorrido promedio de entre dos y tres horas, cuando son trayectos que normalmente se podrían realizar entre treinta minutos a una hora. Un claro ejemplo es la Ciudad de México que "de acuerdo con el Índice de Tráfico de Tom Tom, fabricante de sistemas GPS, en 2016 obtuvo el primer lugar de 295 ciudades en congestión vehicular" (IMCO, 2016: 01). En donde se puede apreciar que trasladarse en la Ciudad de México implica invertir 59% más del tiempo estimado, porcentaje que aumentó considerablemente el año siguiente al subir a 66%.

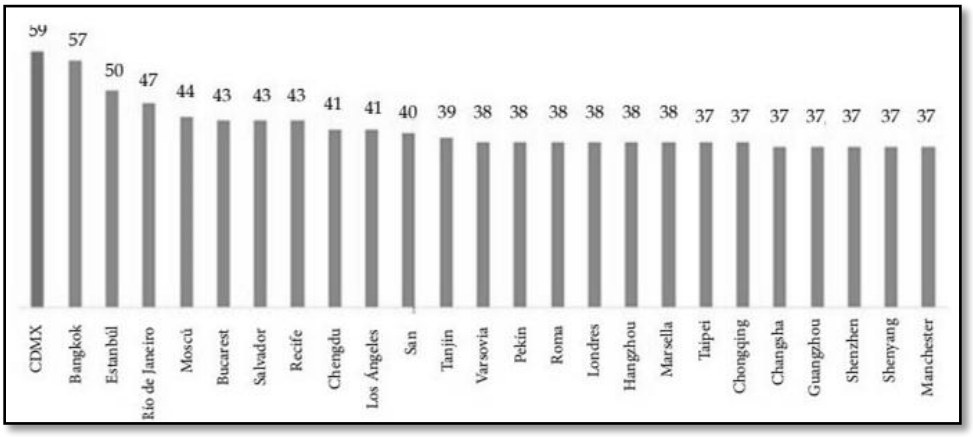


Figura 4. Porcentaje de tiempo adicional en las 25 ciudades más congestionadas del mundo

Frente a estas circunstancias, la movilidad sustentable pasó a ser un tema de gran relevancia para planificar los sistemas de movilidad urbana, ya que es un modelo que promueve la utilización de diferentes medios de transporte que son amigables con el medio ambiente, incluyentes y accesibles. [6]

En lo que respecta a información estadística sobre medios de transporte como Uber, cabe mencionar que, durante la búsqueda de información más detallada sobre el uso de este tipo de aplicaciones, es posible enfrentarse a una carencia de detalles, pues toda la información de estas plataformas no se encuentra disponible para el público en general. Fue hasta el mes de septiembre del año pasado que la Ciudad de México y la ZMG se sumaron a la lista de urbes donde Uber abrió su base de datos que permite conocer patrones de viaje, zonas de alta de manda o comparar tiempos de traslado entre diferentes zonas.

Se trata del sitio web movement.uber.com, donde cualquier persona puede tener acceso de manera gratuita. Esta herramienta se alimenta de los viajes que se han realizado en la plataforma en la Ciudad de México desde el 1 de enero de 2016 a la fecha.

"Uber Movement muestra datos agregados y anonimizados sobre los viajes realizados con la aplicación de Uber. Esta herramienta muestra tendencias de tiempos de traslado promedio, mínimo y máximo en trayectos dentro de las principales zonas metropolitanas de México", explicó la marca.

El filtro de los datos se hace considerando las *Áreas Geoestadísticas Básicas* definidas por el Instituto Nacional de Estadística y Geografía (INEGI). La actualización de *Movement* de Uber para la Ciudad de México se hace cada cuatro veces al año.

La Ciudad de México fue la primera del país en contar con la plataforma de libre acceso, actualmente disponible en más de 30 ciudades del mundo y fue la tercera en América Latina, después de Bogotá y Sao Paulo. Esta plataforma se desarrolló con apoyo del Instituto de Geografía de la Universidad Nacional Autónoma de México, quienes además realizan una investigación para determinar el grado de conectividad y accesibilidad en las diferentes zonas de la Ciudad de México.

"*Uber Movement* es un valioso ejemplo de cómo los datos generados por los usuarios de las empresas de transporte pueden socializarse y utilizarse para generar conocimiento sobre la movilidad de nuestras ciudades, ayudando a identificar oportunidades versus desigualdades y buscar estrategias para equilibrarlas", comentó Héctor Reséndiz, Coordinador Técnico de la Unidad GITS del Instituto de Geografía de la UNAM.

En tanto María Eugenia Zurita, directora de Comunicación de Uber para México menciono que en el presente año estarán disponibles nuevas capas en su plataforma Movement, como velocidad promedio a nivel de cuadras y avenidas, lo que permitirá un análisis más detallado de los viajes que se realizan en esta aplicación en la Ciudad de México. [7]

Es por tal motivo que el procesamiento e integración de esta información se haga a la brevedad posible, ya que hasta el momento hay pocos trabajos y estudios que hagan uso de esta información a fin de poder generar valor que sea utilizado por las diferentes estructuras y organizaciones encargadas de resolver problemas de movilidad en el país.

2.3 Impacto social de la movilidad en México

Una vez tratado el tema anterior, es posible ver e identificar a través de la bibliografía existente, que los temas de análisis y estudio respecto a la movilidad no se enfocan únicamente en lo estadístico, sino que va más allá y se plantean proyectos y trabajos que involucran otros aspectos. El propósito de este subtema es el de hablar sobre los trabajos relacionados a mi proyecto, mismos que tienen una perspectiva social moderna del programa.

¿Cuánto gastamos en transporte? Para la mayoría de las familias mexicanas que habitan en zonas urbanas, el transporte es el segundo concepto de gasto familiar con 19 por ciento, seguido de la educación, con 12 por ciento, duplicando el gasto en vivienda y servicios básicos con 10 por ciento, y superando casi siete veces el gasto en salud, con 3 por ciento.

La movilidad ecológica está echando raíces con la implementación de estos sistemas de préstamo de bicicletas que se han implementado en distintas ciudades, teniendo como resultado, un aumento en los viajes realizados por medio de la bicicleta y cumpliendo con el propósito fundamental de desincentivar el uso del automóvil como medio de transporte.

Un reportaje realizado por Violeta Mélendez en el mes de enero del 2018 para el periódico “el diario” muestra como para inicios del 2018, el programa de bicicletas en la ZMG había cumplido tres años de operaciones, en los que se reportó un impacto positivo en el tránsito motorizado y, por ende, en la reducción de emisiones contaminantes: alcanzando un total de 3.8 millones de viajes realizados en bicicleta, de los cuales 31 por ciento evitó el uso de un automóvil público o privado, que equivale a 1 millón 178 mil traslados.

De acuerdo con el reporte de Violeta, este 31 por ciento se conforma de 25 por ciento de autos privados que habrían sido usados para completar traslados y de 6 por ciento que correspondería al alquiler de un taxi. En total conjuntaron más de un millón de vehículos motorizados que dejaron de circular en calles de Guadalajara, Zapopan y Tlaquepaque, donde se extiende la red de este sistema de bicicletas públicas.

De esta manera es posible ver como estos programas también han funcionado como una opción de transporte multimodal, debido a que 47 por ciento los usuarios lo utilizan para completar sus traslados en transporte público, es decir, se acercan a la parada del autobús o usan la bicicleta al bajarse de éste para llegar a su destino final. El otro 37 por ciento la combina con medios no motorizados, es decir, caminan o usan su propia bici. Considerando la saturación vehicular que ha alcanzado a la fecha, la aceptación del sistema en Guadalajara ha sido favorable, y es que la oportunidad en el avance del uso de bicicleta como medio transporte, posibilita que se conciba desde un punto de vista más amplio, como un sistema de movilidad, indicó Rodrigo Vázquez, Director y Gestor de Proyectos Especiales de Mobiliario Urbano para el Espacio Público de BKT, la empresa que opera y da el mantenimiento al sistema de bicicleta pública de Guadalajara, MiBici: “Hay más de 600 ciudades en el mundo con sistemas de bici pública; hay mucha tecnología y hay buena tecnología de oferta, pero yo creo que el gran reto por lo menos en nuestra empresa BKT Bici Pública, es evidenciar las ventajas de sistemas lo más confiables posibles, ese sería el principal reto, que las ciudades entiendan que no es comprar bicicletas, sino comprar sistemas confiables de transporte de sistemas de bici pública”.

De esta manera es posible analizar que el alcance de este programa se extiende a mucho más que solo números ya que tiene un impacto directo para los usuarios e indirecto para aquellas personas que prefieren seguir utilizando los medios típicos de transporte.

Sin embargo, en la implementación de nuevos sistemas de movilidad como este surgen diversos obstáculos. En la primera etapa de MiBici, Felipe Reyes Lara, Director de Movilidad No Motorizada del Instituto de Movilidad y Transporte de Jalisco mencionó para un reportaje (realizado por Araceli Robles el 10 de mayo del 2016) que la socialización fue difícil ya que hubo que explicarle a la gente de qué se trata, explicarle que es un sistema de transporte que puede utilizar y con el que puede generar intermodalidad. Es un proceso de entendimiento que tiene que ver con implementar el sistema e ir explicándolo poco a poco.

De esta manera, al presentarse un programa como este a una sociedad en donde no se había tenido antes, surgen distintas problemáticas que tienen afectaciones en la sociedad. Un inconveniente que algunos residentes vieron en un principio y no tomaron como una buena señal, fue la instalación de las "bici-estaciones", ya que observaron de manera negativa que les quitaran un espacio de la calle que no se podría utilizar mas. Por otra parte, en la opinión pública intervienen los señalamientos de algunas investigaciones en movilidad, que indican que la bici pública genera un problema para los automotores, quitando capacidad a la vía, sin justificar los resultados de viajes realizados por hora del sistema.

De igual manera, dentro del marco del 5to Encuentro Latinoamericano de Sistemas de Bicicletas Públicas y compartidas que se llevo a cabo en la ciudad de Guadalajara entre los días 16 y 17 de mayo del 2019 se presentó un informe sobre otra problemática social que trata el tema de las afectaciones que sufren las mujeres que hacen uso de este programa. Este informe presenta los resultados del estudio Usuarias del Sistema de Bicicletas Públicas en el Área Metropolitana de Guadalajara y su Experiencia en Movilidad.

Este estudio parte de la premisa relacionada al enfoque de este subtema en donde se busca poner en evidencia que la movilidad no motorizada, además de tener un impacto ambiental, tiene también un importante impacto de orden social. Dentro de los resultados de este informe, es posible analizar como el acoso callejero (traducido en silbidos, comentarios sexuales, alusiones con falta de respeto al cuerpo de las usuarias, contacto físico, entre otros) no se encuentra excluido del programa que forma parte de una nueva alternativa de movilidad en la zona metropolitana. Dada esta situación, es posible identificar ciertos aspectos y características. Como ejemplo, se reportó que el 92.5% de las usuarias encuestadas expresaron haber sido acosadas al ser usuarias del programa. La mayoría (42.9%) respondió que su reacción fue huir del lugar lo más pronto posible, seguido por enojarse (22.8%) y gritarle al agresor (12.9%). Sólo 1.5% dijo haberse paralizado al usar MiBici.

	Ha mejorado	Permanecen igual	Ha empeorado
Gesto de transporte	82.2%	16.2%	1.6%
Estado de ánimo	81.1%	17%	1.9%
Tiempo de traslado	76.9%	20.5%	2.6%
Comodidad	64.4%	32.2%	3.4%
Seguridad	27.9%	66.5%	5.6%
Acoso callejero	25.3%	64.6%	10.1%

Tabla 2. Resultados de encuesta de satisfacción a usuarias del programa de bici pública en Jalisco

En lo que respecta a plataformas de movilidad por medio de taxis ejecutivos, es sumamente importante mencionar a Uber, que ha sido la más usada de todas y la que ha llegado para formar parte de la movilidad del día a día de los mexicanos.

La solución a la movilidad en las grandes metrópolis no pasa por el automóvil, sino por sistemas de transporte público eficiente y seguro. Ciudad como Londres o Tokio tienen un sistema de Metro que es igual o más funcional que cualquier taxi. Y la movilidad no pasa por tener más automóviles, sino por reducirlos e incentivar al ciudadano a usar el transporte público y otros medios como la bicicleta. [8]

Tal y como ya se mencionó, México es uno de los países en el dónde más se gasta en transporte en todo el mundo, según la publicación *The EU in the Word 2015*, la cual proporciona datos estadísticos de la Unión Europea, en comparación con el resto del mundo. Con datos del INEGI, el reporte indica que los hogares mexicanos destinan el 18.78% de sus ingresos mensuales para transportarse, mientras que los expertos recomiendan destinar entre 7% y 10%.

Una de las múltiples causas de esta problemática es que, en varias ciudades de la república el sistema de transporte público es deficiente, está saturado y es de baja calidad; confirma la Encuesta Nacional de Movilidad y Transporte 2015, realizada por la UNAM. La solución para muchos ciudadanos ha sido utilizar servicios de transporte como Uber, Cabify o, ahora, Didi. No por nada, el país ocupa el tercer puesto en volumen de viajes para Uber, después de Estados Unidos y Brasil. Tan solo en la capital circulan 55,000 vehículos de dicha aplicación, de acuerdo con información de CNNMoney. Pero ¿cuánto nos está costando a los mexicanos al año los servicios de transporte privado como Uber, Cabify o Didi?

El eufórico apoyo a Uber, que creció 800% en sus descargas, cuando cientos de taxistas protestaron en su contra en la Ciudad de México, germina en medio de sistemas de transporte ineficientes e inseguros. Parecería ser más fácil tomar un auto con aire acondicionado que surfear entre lamentables sistemas de transporte. Sin embargo, tomemos la euforia con calma. La aplicación de Uber requiere de una tarjeta de crédito, y menos del 40% de los mexicanos tienen una, sin hablar que a más de la mitad le resulta casi imposible costearlo. El servicio es muy bueno, pero no es para todos.

De acuerdo con el sondeo aplicado por la comparadora de servicios financieros Coru.com y la encuestadora Brad.Feebbo, en febrero de 2019; el 30% de los mexicanos utilizan estas plataformas u otro servicio similar tres veces a la semana, el 30% usa este tipo de transporte una vez cada 15 días; el 20% usa este tipo de transporte una vez a la semana: el 10% lo usa más de tres veces a la semana; y 10% asegura no utilizar este tipo de transporte.

Ahora bien, según datos de Finerio, la primera aplicación de finanzas personales automática y gratuita en México, en promedio, sus usuarios han gastado en Uber $145.94, en Cabify $75.42 y en Didi $88.83 por cada viaje. La siguiente tabla resume los gastos por servicios de taxi mediante estas aplicaciones:

Servicio	Costo promedio por viaje	Una vez cada 15 días	Una vez a la semana	Dos veces a la semana	Tres veces a la semana
Uber	$145.94	$3,502.56	$7,588.88	$15,177.76	$22,766.64

Tabla 3. Gasto anual por utilización de plataformas de taxis ejecutivos

2.4 Condiciones políticas de programas de movilidad en México

Si bien el enfoque de este proyecto no está relacionado ni ligado al ámbito político, la bibliografía encontrada sobre el tema expone que este aspecto tiene un impacto directo dentro del problema de movilidad ya que el manejo y distribución de los sistemas se encuentra a cargo del gobierno, mismo que puede o no cambiar cada cierto periodo.

En México, los estados de Jalisco, Nuevo León, Ciudad de México y Querétaro cuentan con una Ley de Movilidad que contempla la movilidad sustentable, en las cuales consideran, entre otros, al peatón, a los ciclistas y a los usuarios de transporte público. Es importante destacar que para el cuadro analítico sólo se compararan la Ley de movilidad de los tres primeros estados. Las leyes actuales hacen referencia al uso de la bicicleta y los beneficios que esta tiene, entre ellos se promueve la actividad física y mejora la calidad de vida de los usuarios, ya que, al ser un medio de transporte principalmente impulsado por la fuerza humana, no consume más energía de la que el ciclista pueda proveer, por lo tanto, no emite ruido, no genera gases y ocupa mucho menos espacio que un auto. Sin embargo, se requiere tener consideraciones más precisas respecto a los ciclistas que fortalezcan el sistema de movilidad, la planificación de vialidades, la seguridad vial, los derechos que estos tienen y la responsabilidad de los gobiernos locales y federales para diseñar los marcos normativos que rigen su funcionamiento.

Dentro del desarrollo de la zona metropolitana de Guadalajara, era inminente la necesidad de un programa de movilidad no motorizada debido a la gran cantidad de contaminación ambiental y gran afuero vehicular, por lo que surgió la idea de este programa como una buena solución. Una investigación realizada en el año 2015 por estudiantes del Instituto Politécnico Nacional analizó la licitación pública LA-914012998-I11-2015 para abastecer el programa MiBici en toda la zona metropolitana de Guadalajara, donde se identificó que ningún proveedor nacional pudo participar dado que una especificación requerida en la licitación es la descripción de una patente que pertenece a “Urban Solutions”. Con base en las leyes establecidas en el estado de Jalisco, (articulo 41 de la Ley de Adquisiciones, Arrendamientos y Servicios del Sector Público) fue posible realizar una adjudicación directa, sin embargo, el hecho de que solo existiera un licitante que cumplía con las especificaciones solicitadas por la licitación, tornó en sin sentido el hecho de que existiera.

En el caso de la Ley de movilidad de la Ciudad de México y la de Jalisco, definen a la bicicleta como todo aquel vehículo no motorizado que es impulsado a base de fuerza humana y al ciclista el que conduce a no más de 25 kilómetros por hora. También dentro de sus disposiciones consideran las ciclovías, al estipular las autoridades encargadas de planificar, ampliar y construir redes de movilidad para este medio de transporte, así como crear los espacios exclusivos para guardar o estacionar las bicicletas.

Las dos entidades federativas incluyen la intermodalidad dentro de su Ley de movilidad, aunque cada una la plantea de forma distinta. La Ley de la CDMX hace alusión a vincular los diferentes medios de transporte y facilitar que los ciclistas ingresen con su bicicleta al Sistema Integrado de Transporte y la Ley de Jalisco da la posibilidad de facilitar el cambio de un transporte privado a la bicicleta. [9]

3. MARCO TEÓRICO/CONCEPTUAL

Resumen: *En este capítulo se presentan las bases teóricas y conceptuales sobre el procesamiento y análisis de información, así como las tecnologías implementadas para dichas actividades*.

3.1. Modelado de datos

El método de crear un modelo de almacenamiento de datos es llamado **Modelado de datos** (DM por sus siglas en inglés) en una base de datos. Esto introduce objetos de datos teóricos y conexiones entre diferentes objetos de datos.

El modelado de datos es un proceso de formulación en un formato estandarizado en un sistema de información el cual ayuda a analizar datos de una manera rápida y efectiva que, a su vez, ayuda a satisfacer las necesidades del negocio. El proceso de modelado de datos requiere de analistas que trabajen en conjunto con las partes interesadas y usuarios para desarrollar un modelo de datos que apoye y soporte la infraestructura de los sistemas de información del negocio.

Normalmente el modelado ocurre en tres distintas capas:

1. **Modelo Físico:** Es un esquema que proporciona información sobre cómo están almacenados los datos físicamente en la base de datos.
2. **Modelo conceptual:** Es la visión del usuario sobre los datos
3. **Modelo lógico:** Se ubica entre el modelo lógico y el modelo conceptual, y representa la lógica que los datos en conjunto forman entre sí dejando de lado la manera es que estos se encuentran almacenados. [10]

3.2. Procesamiento de datos

El procesamiento de datos es la colección y manipulación de los datos en una forma deseada y utilizada. La manipulación no es otra cosa más que procesamiento, el cual, puede ser manual o automático en una secuencia de operaciones predefinida. En el pasado, todo este proceso era realizado manualmente lo cual demandaba mucho tiempo y podía llevar a una posibilidad mayor de errores durante el procesamiento, por lo que hoy en día se realiza de manera automática con el uso de computadoras y recursos en la nube que a su vez conducen a un procesamiento altamente veloz y eficaz con resultados ampliamente confiables.

El procesamiento de datos es simplemente la conversión de datos crudos a información significativa a través de un proceso definido. Los datos son técnicamente manipulados para producir resultados que puedan llevar a una resolución de un problema o a la mejora de una situación actual. Similar al proceso de producción de una empresa, se sigue un ciclo donde la entrada (datos crudos) es alimentada a un proceso (sistemas computacionales, software, etc.) para producir una salida deseada (información e ideas). [11]

3.3. Bases de datos

En el mundo actual existe una cada vez mayor demanda de datos. Esta demanda siempre ha sido patente en empresas y sociedades, pero en los últimos años, la demanda se ha disparado más debido al acceso multitudinario a las redes integradas en Internet y a la aparición de dispositivos móviles que requieren y generan información.

Desde su nacimiento, la informática se ha encargado de proporcionar herramientas que faciliten la gestión de datos. Antes de la aparición de las aplicaciones informáticas, las empresas tenían como únicas herramientas de gestión de datos a los cajones, carpetas y fichas en las que almacenaban los datos. En este proceso manual, el tiempo requerido para manipular estos datos era enorme. Sin embargo, el proceso de aprendizaje era relativamente sencillo ya que se usaban elementos que el usuario reconocía perfectamente. Por esta razón, la informática ha adaptado sus herramientas para que los elementos que el usuario maneja en el ordenador se parezcan a los que se utilizaban manualmente. [12]

Las bases de datos permiten mejorar la calidad de las prestaciones de los sistemas informáticos y aumentar su rendimiento:

- ✓ **Independencia de los datos**
- ✓ **Menor redundancia**
- ✓ **Integridad**
- ✓ **Coherencia**
- ✓ **Mayor seguridad**
- ✓ **Acceso más eficiente**
- ✓ **Reducción del espacio de almacenamiento**
- ✓ **Fácil acceso**

3.4. SQL

SQL (Structured Query Language), por sus siglas en inglés, es el medio estándar de manipulación y consulta de datos en bases de datos relacionales. Es un lenguaje de dominio específico usado en programación el cual es particularmente útil en el manejo de datos estructurados.

Fue desarrollado en los 70's por investigadores de IBM que buscaban un lenguaje que les permitiera transformar y consultar conjuntos de datos usando una sintaxis expresiva la cual se basara en los conceptos de teoría de conjuntos en lugar de un lenguaje de programación típico como C o Java.

SQL permite leer, insertar, actualizar y borrar registros en un Sistema de Manejo de Bases de Datos Relacional (RDBMS por sus siglas en inglés) a través de "consultas" que están divididas en dos categorías:

1. Lenguaje de Manipulación de Datos (DML): SELECT, INSERT, UPDATE y DELETE
2. Lenguaje de Definición de Datos (DDL): CREATE, ALTER, DROP

3.5. Data Warehouse

Un Data Warehouse (DWH) es un almacén electrónico donde generalmente una empresa u organización mantiene una gran cantidad de información. Los datos de un DW deben almacenarse de forma segura, fiable, fácil de recuperar y fácil de administrar. DW es una arquitectura de almacenamiento de datos que permite a los ejecutivos de negocios organizar, comprender y utilizar sus datos para tomar decisiones estratégicas.

Es un repositorio unificado para todos los datos que recogen diversos sistemas de una empresa. El repositorio puede ser físico o lógico y hace hincapié en la captura de datos de diversas fuentes sobre todo para fines analíticos y de acceso. Normalmente, un DW se aloja en un servidor corporativo o cada vez más, en la nube. Los datos de diferentes aplicaciones de procesamiento de transacciones Online (OLTP) y otras fuentes se extraen selectivamente para su uso por aplicaciones analíticas y de consultas por usuarios. [13]

La arquitectura de un DW puede ser dividida en tres estructuras simplificadas: básica, básica con un área de ensayo y básica con área de ensayo y data marts.

- **Con una estructura básica:** Sistemas operativos y archivos planos proporcionan datos en bruto que se almacenan junto con metadatos. Los usuarios finales pueden acceder a ellos para su análisis, generación de informes y minería.
- **Al añadir un área de ensayo:** Se puede colocar entre las fuentes de datos y el almacén, esta proporciona un lugar donde los datos se pueden limpiar antes de entrar al almacén. Es posible personalizar la arquitectura del almacén para diferentes grupos dentro de la organización.
- **Se puede hacer agregando data marts:** Son sistemas diseñados para una línea de negocio en particular. Se pueden tener data marts separados para ventas, inventario y compras, por ejemplo, y los usuarios finales pueden acceder a datos de uno o de todos los data marts del departamento.

La centralización de la información combina los registros históricos con otros datos más actuales y, de esta forma, la función de *reporting* se ve enriquecida ya que cualquier informe se elabora a partir de datos procedentes de diferentes fuentes. Contar con un DW reduce el tiempo necesario para encontrar y analizar los datos importantes, consiguiendo que las operaciones sean más eficientes. [14]

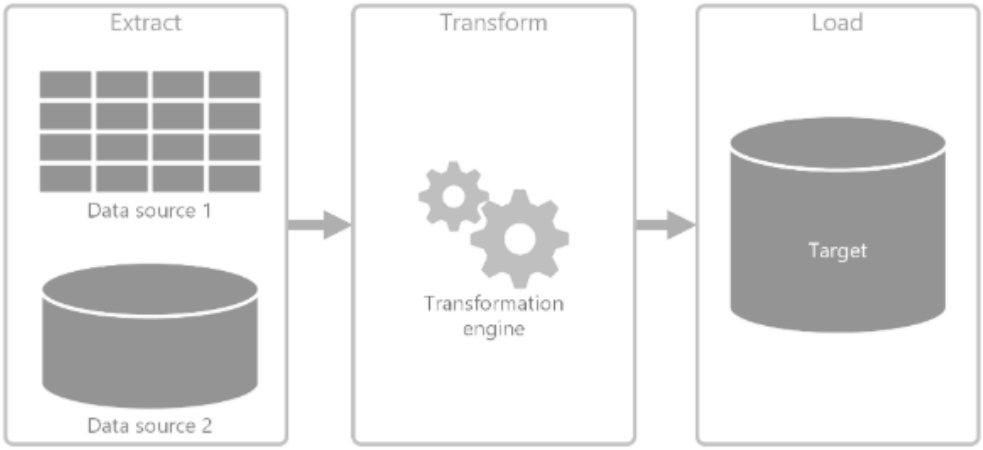
3.6. ETL

Un problema común que las empresas y organizaciones enfrentan es el hecho de obtener información de distintas fuentes, en múltiples formatos y moverla a uno o más sistemas de almacenamiento de información. El sistema destinado a almacenar toda esta información puede no tener la misma estructura que la fuente, y ocasionalmente, el formato es diferente o los datos necesitan ser modificados y "limpiados" antes de ser almacenados.

Varias herramientas, servicios y procesos han sido desarrollados a través de los años para ayudar enfrentar todos estos retos. Sin importar el proceso utilizado, hay una necesidad común de coordinar el trabajo y aplicar un cierto nivel de transformaciones dentro del "pipeline" de datos.

ETL (Extract, Transform and Load) por sus siglas en inglés, se define como un pipeline de datos usado para recaudar datos de distintas fuentes, transformar dichos dantos de acuerdo con las reglas y especificaciones del negocio y, finalmente, cargar toda la información dentro un sistema de almacenamiento especifico. El trabajo de transformación del ETL toma lugar en un "motor" especializado que constantemente involucra el uso de tablas estacionales para almacenar la información temporalmente a medida que estos son transformados y cargados en su destino.

La transformación de datos que se implementa, usualmente involucra varias operaciones como filtrado, ordenamiento, agregaciones, unión de datos, deduplicación y validación de datos.



En ocasiones, las tres fases de un proceso de ETL son ejecutadas en paralelo para ahorrar tiempo. Por ejemplo, mientras que los datos son extraídos, un proceso de transformación podría estarce aplicando en datos ya recibidos y son preparados para ser cargados en su destino. [15]

3.7. Data Lake

Un Data Lake (DL) es un repositorio donde se pueden almacenar todos los datos de una empresa u organización, tantos datos en bruto, estructurados, sin estructurar, sin ningún tipo de procesamiento y sin ningún tipo de esquema pueden ser almacenados hasta que son necesarios para ser analizados.

La información que se almacena en un Data Lake procede de diversas fuentes de datos por lo que se tienen diferentes formatos y tipos; procedentes de bases de datos, documentos ofimáticos, registros de servidores, recursos extraídos de Internet, redes sociales, etc. Con el objetivo de ser estudiados y analizados posteriormente.

Las empresas vierten los datos en estos almacenes y los recuperan cuando son necesarios. Es en ese momento, cuando las empresas tienen la necesidad de los datos, que estos son ordenados y se diseña una

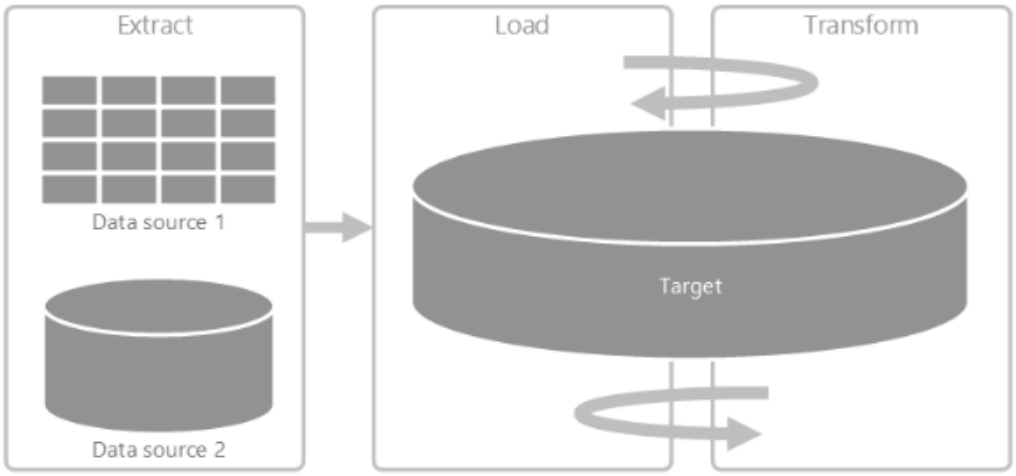
estructura de análisis apropiada. Podríamos describir el data lake como un almacenamiento de bajo coste y el acceso a la información original es directo al disponer de todos los datos en bruto.

Un DL funciona de la siguiente manera: se asigna un identificado único a cada elemento del DL y se etiqueta con un conjunto de etiquetas de metadatos extendidas. Cuando en la empresa se presenta una cuestión sobre el negocio que debemos resolver y requerimos de datos, podemos solicitarle al DL los datos que están relacionados con esa cuestión. Una vez obtenidos podemos analizar ese conjunto de datos más pequeño para ayudar a obtener una respuesta. [16]

3.8. ELT

Extraer, Cargar y Transformar (ELT por sus siglas en inglés) difiere del proceso ETL únicamente en lugar donde las trasformaciones toman lugar. En un proceso de ELT, la transformación es realizada en el sistema de almacenamiento destino. En lugar de usar un proceso de trasformación separado, las capacidades de procesamiento del sistema de almacenamiento destino son utilizadas para transformar los datos. Esto simplifica la arquitectura al remover este proceso intermedio de transformación.

Otro beneficio de utilizar este proceso es el hecho de que al escalar el sistema de almacenamiento destino, el rendimiento de procesamiento y transformación escala de igual manera, por lo que, este proceso debe ser considerado cuando se tiene un sistema destino eficiente.



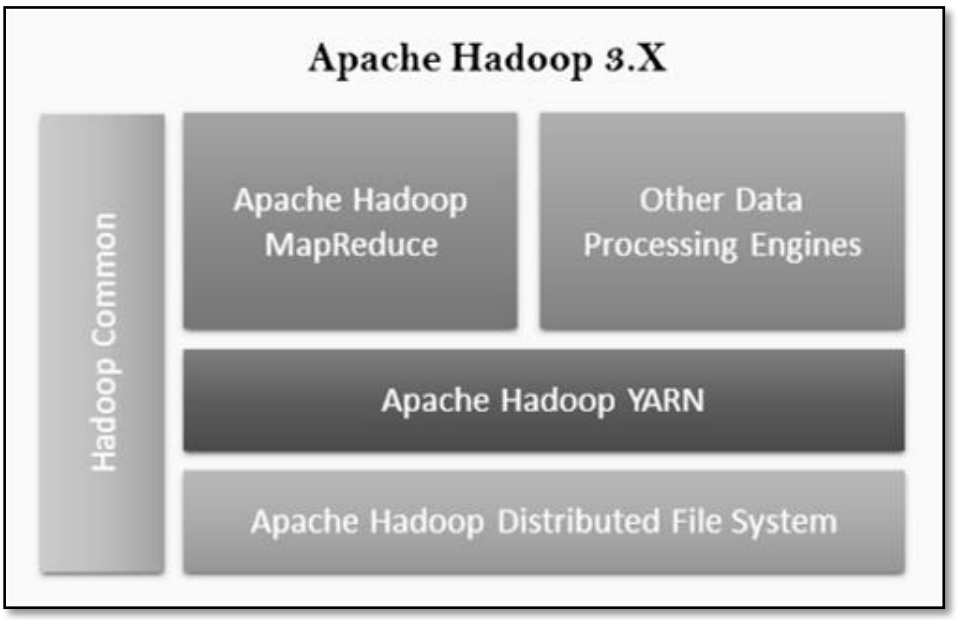
Los casos de uso típicos para ELT caen dentro del ámbito del *Big Data*. Por ejemplo, es posible comenzar extrayendo los datos e insertarlos en archivos planos en almacenamiento escalable, como el sistema de archivos distribuido de Hadoop (HDFS). Las tecnologías como Spark o Hive se pueden utilizar para consultar los datos de origen. El punto clave con ELT es que el almacén de datos utilizado para realizar la transformación es el mismo almacén de datos donde finalmente se consumen los datos. Este almacén de datos lee directamente desde el almacenamiento escalable, en lugar de cargar los datos en su propio almacenamiento patentado. Este enfoque omite el paso de copia de datos presente en ETL, que puede ser una operación que requiere mucho tiempo para grandes conjuntos de datos. [17]

3.9. Apache Hadoop

Apache Hadoop es una colección de software tipo Open Source (OS) que permite procesamiento y almacenamiento distribuido de grandes datasets a través de un cluster de diferentes tipos de sistemas computacionales. El framework de Apache Hadoop consiste en los siguientes 4 módulos principales:

1. Apache Hadoop Common
2. Apache Hadoop Distributed File System (HDFS)
3. Apache Hadoop MapReduce
4. Apache Hadoop YARN

Cada uno de estos módulos cubre diferentes capacidades del framework de Hadoop. El siguiente diagrama representa su posición en términos de aplicabilidad para Hadoop.



Apache Hadoop fue inventado para resolver grandes problemas de datos que ningún sistema o software comercial pueden resolver. Con la ayuda de este software, los datos que antes solían ser almacenados o perdidos, son ahora utilizados en este sistema. Como todo software que es utilizado, existen ciertas características que le den su gran valor a esta herramienta, en donde las principales son:

- ✓ **Fiabilidad:** El sistema distribuido de archivos de Hadoop ofrece replicación de datos (3 por default) lo que asegura que no hay pérdida de datos a pesar de posibles fallas que pueda presentar el hardware.
- ✓ **Flexibilidad:** La gran mayoría de datos que los usuarios se encuentran en el día a día, son no estructurados. Tradicionalmente, este tipo de datos pasan desapercibidos, sin embargo, con Apache Hadoop, distintos datos (incluyendo estructurados y no estructurados) pueden ser procesados, almacenados y analizados para tomar mejores decisiones futuras.

- ✓ **Efectividad de costos:** Apache Hadoop es completamente OS, lo que implica que no tiene costo alguno. A diferencia del software tradicional, este puede correr en cualquier hardware o sistema y no requiere servidores de alta gama; La inversión general y el costo total de implementar Hadoop es mucho menor que el que se requeriría para procesar el mismo tipo de información.
- ✓ **Escalabilidad:** Hadoop es completamente un sistema distribuido. Con el aumento de información, la implementación de clústeres de Hadoop pueden añadir dinámicamente más nodos o inclusive quitar basándose en la demanda procesamiento y almacenamiento de datos.
- ✓ **Alta disponibilidad:** Con la replicación de datos y el masivo cómputo paralelo corriendo en un hardware multinodo, las aplicaciones que utilizan Hadoop, proveen de un ambiente con alta disponibilidad para todas las implementaciones.
- ✓ **Espacio ilimitado de almacenamiento:** Hadoop 3.x y sus versiones subsecuentes soportan más de 10,000 nodos en un cluster, mientras que Hadoop 2.x soporta hasta 10,000 nodos. Con este procesamiento paralelo masivo, Apache Hadoop ofrece poder computacional ilimitado para todas las aplicaciones.
- ✓ **Compatibilidad con la nube:** Hoy en día, casi todos los proveedores de servicios en la nube soportan Hadoop directamente como un servicio, lo que significa que una configuración completamente automatizada está disponible como servicio On-Demand. De igual manera soporta escalamiento masivo, por lo que en general, se vuelve un modelo atractivo para ser utilizado. [18]

3.9.1. Hadoop Distributed File System (HDFS)

HDFS es un sistema de archivos distribuido diseñado para correr en hardware básico. Tiene muchas similitudes con sistemas de archivos distribuidos existentes. Sin embargo, las diferencias con otros sistemas de archivos distribuidos son significativas. HDFS es altamente tolerante a fallas y está diseñado para ser ejecutado en hardware de bajo costo. HDFS provee de un alto rendimiento de acceso a los datos de la aplicación y es adecuado para aplicaciones que tienen grandes conjuntos de datos.

Falla en el hardware es la norma más que la excepción. Una instancia de HDFS puede consistir de cientos o miles máquinas servidoras, cada una almacenando parte de los datos del sistema de archivos. El hecho de que haya un gran número de componentes y que cada uno tenga una probabilidad no trivial de falla significa que algunos componentes de HDFS son siempre no funcionales. Por lo tanto, la detección de fallas y rápida automática recuperación de ellos es una meta núcleo arquitectónica de HDFS.

Aplicaciones que corren en HDFS tienen grandes conjuntos de datos. Un archivo típico en HDFS es de gigabytes a terabytes en tamaño por lo que HDFS está configurado para soportar grandes archivos. Provee un gran ancho de banda para datos agregados y escala a cientos de nodos en un solo clúster además de que soporta cientos de millones de archivos en una sola instancia. Las aplicaciones en HDFS necesitan un modelo de una sola escritura y muchas lecturas para los archivos por lo que una vez que un archivo es creado, escrito y cerrado, no necesita cambio alguno. Está asunción simplifica la coherencia de errores en los datos y permite un gran rendimiento de acceso.

HDFS tiene una arquitectura tipo maestro/esclavo. Un cluster con HDFS consiste de un llamado **NameNode**, un servidor maestro que maneja el espacio de nombre del sistema de archivos y regula el acceso a los archivos por los clientes. Además, hay un número de **DataNodes,** usualmente uno por nodo en el cluster, que manejan el almacenamiento asignado a los nodos en los que corren. HDFS expone un espacio de nombre del sistema de archivos y permite, a los datos de los usuarios, el ser almacenados en archivos. Internamente, un archivo es divido entre uno o más bloques y estos bloques son almacenados en un conjunto de DataNodes. El NameNode ejecuta las operaciones del sistema de archivos como abrir, cerrar y renombrar archivos y directorios. De igual manera, determina el mapeo de los bloques a los DataNodes. Los DataNodes son responsables de solicitudes de lectura y escritura por parte de los clientes del sistema de archivos. También llevan a cabo la creación, eliminación y replicación de bloques sobre una instrucción del NameNode.

HDFS soporta una organización de archivos jerárquica tradicional. Un usuario o una aplicación pueden crear directorios y almacenar archivos dentro de estos directorios, el espacio de nombres del sistema de archivos es similar a la mayoría de los otros sistemas de archivos existentes; se puede crear y remover archivos, mover un archivo de un directorio a otro, o renombrar un archivo. El NameNode mantiene el nombre del espacio del sistema de archivos. Cualquier cambio al sistema o a sus propiedades es almacenado por el NameNode. Una aplicación puede especificar el número de réplicas de un archivo que deberían ser mantenidas por HDFS.

En lo que a replicación de datos se refiere, este sistema está diseñado para almacenar de forma confiable archivos muy grandes en un clúster de igual tamaño. Almacena cada archivo como una secuencia de bloques; todos los bloques en un archivo, exceptuando el último bloque, son del mismo tamaño. Los bloques de un archivo son replicados para ser tolerantes a fallas. El tamaño del bloque y el factor de replicación son configurables por archivo. Una aplicación puede especificar el número de réplicas de un archivo. El factor de replicación puede ser definido en la creación del archivo y puede ser cambiado después. El NameNode toma todas las decisiones sobre la replicación de los bloques. Periódicamente recibe una señal (Heartbeat) y un reporte (Blockreport) de cada DataNode en el clúster, por lo que el recibir esta señal, implica que el DataNode se encuentra funcionando adecuadamente. El reporte contiene una lista de todos los bloques en el DataNode.

Los protocolos de comunicación están en capas sobre el protocolo TCP/IP. Un cliente establece una conexión a un puerto TCP configurable en la máquina del NameNode. Este habla el protocolo del cliente con el NameNode. El DataNode habla con el NameNode usando el protocolo del DataNode. [19]

El objetivo principal de HDFS es el de almacenar datos de una manera confiable aún y con la presencia posible de fallas, misma que pueden ser de tres tipos:

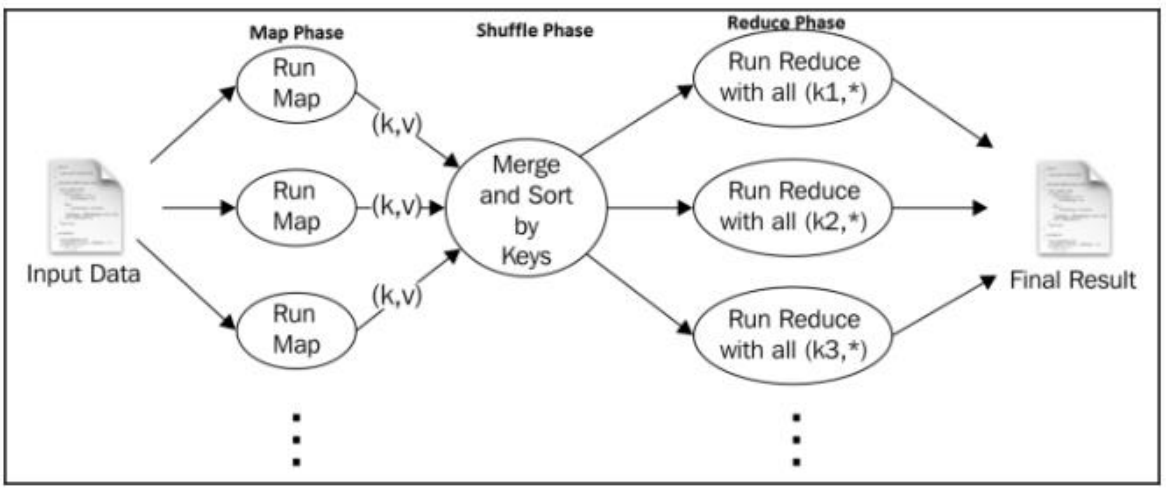
1. Fallas en el NameNode
2. Fallas en los DataNodes
3. Particiones de red

3.9.2. MapReduce

Hadoop MapReduce es un framework de procesamiento de datos que puede ser utilizado para procesar cantidades masivas de información almacenada en HDFS. Como se ha mencionado anteriormente, el procesamiento distribuido de cantidad masiva de información en una manera confiable y eficiente no es una tarea fácil. Hadoop MapReduce apunta el hacerlo fácil para los usuarios previendo de una abstracción limpia para programadores por medio de paralelización automática de los programas y previendo de soporte de tolerancia a fallas administrado por el framework.

El modelo de programación de Mapreduce consiste en funciones Map y Reduce. La función Map recibe cada registro de los datos de entrada (líneas de un archivo, filas de una base de datos, etc.) como pares tipo *key-value* como entrada y como salida. Por diseño, la invocación de la función Map es independiente de otras, permitiendo al framework utilizar el método divide y conquistaras para ejecutar el cómputo en paralelo. Esto de igual manera permite ejecuciones o re-ejecuciones de tareas Map en caso de fallas o cargas inbalanceadas sin afectar el resultado del cómputo. Típicamente, Hadoop crea una sola instancia de una tarea Map para cada bloque de datos de entrada almacenados en HDFS. El número de invocaciones de funciones Map dentro de una instancia de tarea es igual al número de registros del bloque con los datos de entrada de la instancia de la tarea.

Hadoop MapReduce agrupa los registros *key-value* de salida de todas las tareas Map de un cómputo por su llave (key) y los distribuye a las tareas Reduce. Esta distribución y transmisión de datos a las tareas Reduce es llamada fase *Shuffle* del cómputo MapReduce. Los datos de entrada para cada tarea Reduce es también ordenado y agrupado por su llave. La función Reduce es invocada para para cada llave y el grupo de valores ordenados de esa llave. En un programa MapReduce típico, los usuarios solo tiene que implementar las funciones Map y Reduce y Hadoop se encargará de programarlas y ejecutarlas en paralelo. Hadoop re-ejecutará cualquier tarea fallida y también proveerá medidas para mitigar cualquier cómputo inbalanceado. El siguiente diagrama muestra de una mejor manera el flujo computacional de datos sobre tareas MapReduce.



En la primera versión de Hadoop, los componentes de MapReduce consistían en el proceso *JobTracker* el cual corre en el nodo maestro del clúster manejándolo y coordinando los trabajos, y de *TaskTrackers* los cuales corrían en cada nodo asignado a cómputo ejecutando y coordinando las tareas ejecutadas en cada nodo. Ninguno de estos procesos existió en las versiones 2.X de MapReduce (MR2), en donde el coordinador responsable de trabajos es manejado por un *ApplicationMaster* que es instalado *on-demand* a través de YARN. [20]

3.9.3. YARN

YARN es un acrónimo de **Yet Another Resource Negotiator** (por sus siglas en inglés) fue introducido como un componente gestionador de recursos en la segunda generación de Hadoop, YARN fue agregado como un subproyecto. Con MapReduce enfocado únicamente en el procesamiento de batch, YARN está diseñado para proveer una plataforma de procesamiento genérico para datos almacenados en un clúster y así como un framework manejador de recursos para un cluster robusto.

YARN divide sus responsabilidades en diferentes componentes, cada uno con una tarea en específica por hacer. En la primera versión de Hadoop *JobTracker* era el componente que se encargaba del manejo de recursos, programación de trabajo (job scheduler) y monitoreo de trabajos. YARN divide estas responsabilidades en *ResourceManager* y *ApplicationMaster.*

3.9.3.1. ResourceManager

Este componente es un servicio que maneja la programación de los recursos computacionales a las aplicaciones. Optimiza la utilización del cluster en términos de memoria, núcleos del CPU, respuesta a solicitudes y SLA (Service-level agreement) por sus siglas en inglés. Este servicio tiene a su vez dos componentes:

1. **Programador (scheduler)**: Este es un componente que únicamente es responsable de asignar recursos a las aplicaciones enviadas al clúster, aplicando condiciones de capacidades y colas. No provee ninguna garantía de finalización y/o monitoreo de los trabajos (Jobs) enviados, únicamente asigna los recursos del clúster.
2. **ApplicationManager (AsM):** Este es un servicio utilizado para manejar las aplicaciones a través del clúster que es responsable de aceptar una solicitud de una aplicación, proveyendo de recursos para que la aplicación pueda empezar, monitoreando el progreso de la aplicación y reiniciando, en caso de falla.

3.9.3.2. NodeManager

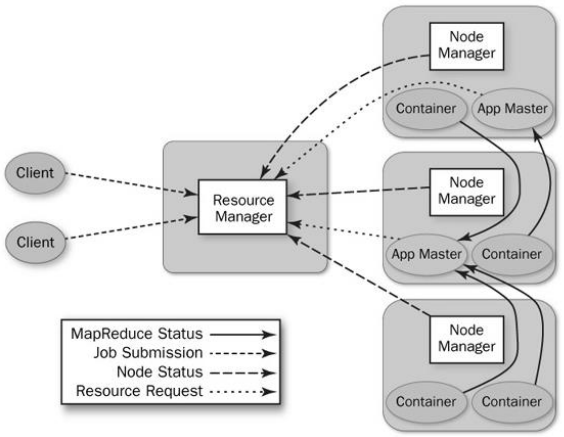
El NodeManager es el servicio responsable de la ejecución de contenedores (Containers) basándose en la capacidad del nodo. Dicha capacidad es calculada basado en la memoria instalada y el número de núcleos del CPU. El NodeManager envías señales continuas (heartbeats) al ResourceManager para actualizar constantemente su "estado de salud". De igual manera, también envía su estatus, el cual puede ser el estatus del nodo donde está corriendo o el estatus de las tareas ejecutadas en él.

3.9.3.3. ApplicationMaster

Es una librería específica por aplicación que maneja cada instancia de una aplicación que corre en YARN. YARN trata a ApplicationMaster como una biblioteca de terceros responsable de la negociación de los recursos del ResourceManager y trabaja con el NodeManager para ejecutar las tareas. El ResourceManager asigna los contenedores para la ApplicationMaster y estos contenedores son después usados para correr los procesos específicos de una aplicación. De igual manera rastrea el estatus de la aplicación y monitorea el progreso de los contenedores. Cuando la ejecución de un contenedor es completada, la librería se encarga de anular el registro con el ResourceManager así como de ella misma cuando la ejecución de la aplicación es completada.

3.9.3.4. Container

Un contenedor es un paquete lógico de recursos en términos de memoria, CPU, disco, etc. Que están vinculados a un nodo en particular. En la primera versión de YARN, un contenedor es equivalente a un bloque de memoria. El servicio de programación del ResourceManager dinámicamente asigna recursos como contenedores. Un contenedor otorga privilegios a una ApplicationMaster para usar una cantidad específica de recursos de un host en específico. Una ApplicationMaster es considerada como el primer contenedor de una aplicación y él maneja la ejecución lógica de una aplicación en contenedores asignados. [21]



3.9.4. Hive

Hive es un proyecto de standard de software construido sobre Hadoop para proveer análisis y consultas de información con un lenguaje tipo SQL (Structured Query Language). Provee una manera de acceder a los datos en HDFS, permitiéndole a Hadoop el ser usado como un Warehouse. El lenguaje de Hive, HQL (Hive Query Language), tiene semánticas y funciones similares al SQL estándar en las bases de datos relacionales, por lo que analistas de bases de datos experimentados pueden fácilmente acceder y utilizar la información de HDFS. HQL puede correr en diferentes motores de ejecución como Map Reduce, Tez y Spark.

Algunas características de Hive son:

- ✓ Indexación que proporciona rapidez a las consultas
- ✓ Almacenamiento de diferentes tipos de datos como RCFile, HBase, ORC entro otros
- ✓ Almacenamiento de metadatos en un RDBMS lo que le permite reducir el tiempo de análisis durante la ejecución de una consulta.
- ✓ Operaciones sobre datos comprimidos almacenados dentro del ecosistema de Hadoop usando algoritmos que incluyen DEFLATE, BWT, snappy, etc.
- ✓ Funciones definidas por usuario para manipular fechas, textos y otras herramientas de minería de datos.
- ✓ Consultas estilo SQL, las cuales son convertidas automáticamente a MapReduce o Tez, o tareas Spark.

Por default, Hive almacena los metadatos en una base de datos Apache Derby embebida, pero cualquier otra puede ser usada.

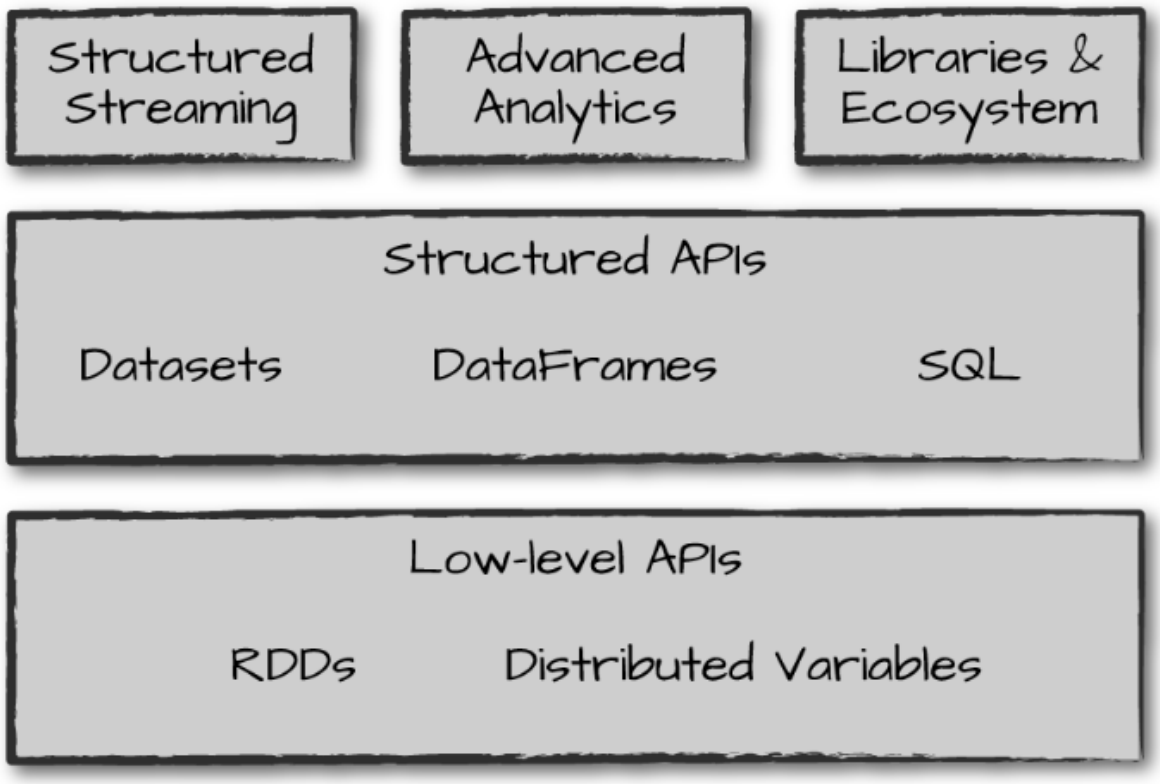
Los principales componentes dentro de la arquitectura de Hive son:

- **Metastore:** Almacena los metadatos como su esquema y ubicación para cada una de las tablas. De igual manera incluye metadatos de las particiones lo que ayuda que el controlador registre el progreso de varios conjuntos de datos distribuidos a través del clúster. Tener el registro de los datos es crucial, por lo que un servidor de respaldo continuamente replica los datos los cuales pueden ser utilizados como respuesta a una pérdida de información.
- **Driver:** Actúa como el controlador que recibe las consultas (HQL). Empieza la ejecución de estas creando sesiones y monitoreando el ciclo de vida y el progreso de la ejecución. Almacena los metadatos necesarios generados durante la ejecución de la consulta. De igual manera, actúa como un punto de colección de datos o resultados de las consultas en las operaciones de reducción.
- **Compiler:** Realiza compilaciones de consultas, mismas que son convertidas a un plan de ejecución. Este plan contiene las tareas y pasos necesarios para ser ejecutados por Hadoop MapReduce.
- **Optimizer:** Realiza varias transformaciones en el plan de ejecución para un obtener un grafo optimizado. Las transformaciones pueden ser combinadas a fin de obtener un mejor rendimiento durante la ejecución. De igual manera puede dividir las tareas, como aplicar una transformación a los datos antes de realizar una operación de reducción.

- **Executor:** Después de la compilación y optimización, se ejecutan las tareas. Este componente interactúa con el *job travker* de Hadoop para programar la ejecución de las tareas. Toma cuidado de establecer un "pipeline" a fin de asegurar que las tareas con dependencias sean ejecutadas solo si los pre-requisitos se cumplen.
- **CLI (Command Line Interface):** Provee una interfaz a través de la red (Similar a los protocolos JDBC o ODBC) para que un usuario externo pueda interactuar con Hive enviando consultas, instrucciones y monitoreando el estatus de los procesos. [22]

3.9.5. Spark

Spark es un motor computacional unificado y un conjunto de librerías para procesamiento paralelo de datos en clústeres computacionales. Spark es el motor open source activamente más desarrollado, haciéndolo una herramienta estándar para cualquier desarrollador o científico de datos interesado en Big Data. Spark soporta una amplia variedad de lenguajes de programación (Python, Java, Scala y R), incluye librerías para diversas tareas yendo desde SQL, Streaming y Machine Learning, y corre en una gran variedad de sistemas desde una laptop hasta un clúster o cientos de servidores. Esto lo hace un sistema fácil para empezar a escalar a procesamientos de big data.



La presente figura ilustra los componentes y librerías que Spark ofrece a los usuarios.

Una de las preguntas más frecuentes sobre Spark es su relación con Hadoop. ¿Se trata acaso de otra tecnología competencia del famoso framework? En realidad, Spark es la evolución natural de Hadoop, cuya funcionalidad es muy rígida y limitada en el sentido de que no aprovecha al máximo las capacidades del procesamiento distribuido.

Algunas de las evoluciones que supone Spark frente a su predecesor son el procesamiento en memoria que disminuye las operaciones de lectura/escritura, la posibilidad de análisis interactivo con SQL (similar a Hive en cierto modo) y la facilidad para interactuar con múltiples sistemas de almacenamiento persistente.

Apache Spark es un motor de procesamiento distribuido responsable de orquestar, distribuir y monitorizar aplicaciones que constan de múltiples tareas de procesamiento de datos sobre varias máquinas de trabajo, que forman un cluster.

Como ya hemos mencionado, es posible leer los datos desde diferentes soluciones de almacenamiento persistente como Amazon S3 o Google Storage, sistemas de almacenamiento distribuido como HDFS, sistemas key-value como Apache Cassandra, o buses de mensajes como Kafka.

A pesar de ello, Spark no almacena datos en sí mismo, sino que tiene el foco puesto en el procesamiento. Este es uno de los puntos que lo diferencian de Hadoop, que incluye tanto un almacenamiento persistente (HDFS) como un sistema de procesamiento (MapReduce) de una manera muy integrada.

Es importante hablar de la velocidad de procesamiento: la clave es la posibilidad que ofrece Spark para realizar el procesamiento en memoria. Esto, y la extensión del popular MapReduce para permitir de manera eficiente otros tipos de operaciones: Consultas interactivas y Procesamiento en Streaming.

Respecto a su propósito general, la virtud de Spark es estar diseñado para cubrir una amplia gama de cargas de trabajo que previamente requerían sistemas distribuidos diferentes. Estos sistemas incluyen procesamiento batch, algoritmos iterativos, consultas interactivas, procesamiento Streaming, entre otros más. Spark es flexible en su utilización, y es que ofrece una serie de APIs que permiten a usuarios con diferentes campos de trabajo poder utilizarlo. Incluye APIs de Python, Java, Scala, SQL y R, con funciones integradas y en general una performance razonablemente buena en todas ellas.

Permite trabajar con datos más o menos estructurados (RDDs, dataframes, datasets) dependiendo de las necesidades y preferencias del usuario. Además, se integra de maneras muy cómodas con otras herramientas Big Data, aquellas procedentes del proyecto Apache. En particular, como era de esperar, cabe destacar la integración con Hadoop: Spark puede ejecutarse en clústeres Hadoop y acceder a los datos almacenados en HDFS y otras fuentes de datos de Hadoop (Cassandra, Hbase, Kafka).

Apache Spark es un framework de programación para procesamiento de datos distribuidos diseñado para ser rápido y de propósito general. Como su propio nombre indica, ha sido desarrollada en el marco del proyecto Apache, lo que garantiza su licencia Open Source. [23]

3.10. AMAZON WEB SERVICES

Amazon Web Services, también conocida como AWS, es un conjunto de herramientas y servicios de cloud computing de Amazon. Este servicio se lanzó oficialmente en 2006 y para junio de 2007 AWS ya contaba con una base de usuarios de aproximadamente 180 mil personas. Entre las empresas que la utilizan se encuentran algunas como Reddit, Foursquare, Pinterest, Netflix, la NASA o la CIA, y algunas españolas como Mapfre, el FC Barcelona o Interflora. Esto se debe principalmente a la madurez del servicio frente a otros similares y las posibilidades que ofrece el amplio abanico de herramientas disponibles. En la Guía de Cloud Computing podrá encontrar una comparativa de todas las herramientas de Amazon Web Services con las de otras plataformas similares.

La tendencia general para las plataformas en la nube es la de ofrecer la mayor cantidad posible de herramientas y servicios, para que así se pueda crear todo un entorno de computación en una misma nube. Al igual que otras plataformas como Microsoft Azure o Google Cloud Platform, Amazon dispone de una gran cantidad de herramientas para la gestión de diferentes elementos dentro de la empresa. Los servicios de AWS están preparados tanto para autónomos, como pequeñas y medianas empresas o grandes corporaciones, ya que existen posibilidades para escalar las instancias o el almacenamiento según su empresa vaya también creciendo.

Amazon Web Services ofrece herramientas en las siguientes categorías:

- ✓ Cloud computing: todo lo necesario para la creación de instancias y el mantenimiento o el escalado de las mismas. Amazon EC2 es el rey indiscutible dentro de los servicios de computación en la nube de Amazon.
- ✓ Bases de datos: distintos tipos de bases de datos pueden permanecer en la nube mediante el servicio Amazon RDS, que incluye distintos tipos a elegir, como MySQL, PosgreSQL, Oracle, SQL Server y Amazon Aurora, o Amazon DynamoDB para NoSQL.
- ✓ Creación de redes virtuales: permite la creación de redes privadas virtuales a través de la nube, gracias principalmente al servicio Amazon VPC.
- ✓ Aplicaciones empresariales: Amazon WorkMail es el servicio de correo empresarial que ofrece Amazon, al que pueden unirse otros servicios como Amazon WorkDocs y Amazon WorkSpaces.
- ✓ Almacenamiento y gestores de contenido: tipos de almacenamiento diferentes, tanto para archivos con acceso regular, poco frecuente o incluso como archivo.
- ✓ Inteligencia de negocios o Business Intelligence (BI): sistemas para análisis de datos empresariales a gran escala y otros servicios para la gestión de flujos de datos.
- ✓ Gestión de aplicaciones móviles: herramientas como Amazon Mobile Hub permiten la gestión, creación, testeo y mantenimiento de aplicaciones móviles a través de la nube.
- ✓ Internet de las cosas (Internet of Things): para establecer conexiones y análisis de todos los dispositivos conectados a internet y los datos recogidos por los mismos.
- ✓ Herramientas para desarrolladores: para almacenar código, implementarlo automáticamente o incluso publicar software mediante un sistema de entrega continúa.

- ✓ Seguridad y control de acceso: se pueden establecer autenticaciones en varios pasos para poder proteger el acceso a sus sistemas internos, ya estén en la nube o instalados de forma local en sus instalaciones.

Estos son principalmente los servicios que se podrán encontrar en Amazon Web Services, aunque es cierto que Amazon actualiza periódicamente la oferta de servicios y herramientas disponibles en su plataforma. No obstante, hay que tener en cuenta que cada empresa puede tener particularidades que no se puedan cubrir con Amazon Web Services. Para comprobar las diferencias de esta y otras plataformas en la nube, consulte la Guía de Cloud Computing. [24]

3.10.1. S3

Amazon Simple Storage Service (Amazon S3) es un servicio de almacenamiento de objetos que ofrece escalabilidad, disponibilidad de datos, seguridad y rendimiento líderes en el sector. Esto significa que clientes de todos los tamaños y sectores pueden utilizarlo para almacenar y proteger cualquier cantidad de datos para diversos casos de uso, como sitios web, aplicaciones móviles, procesos de copia de seguridad y restauración, operaciones de archivado, aplicaciones empresariales, dispositivos IoT y análisis de big data. Amazon S3 proporciona características de administración fáciles de utilizar que le permiten organizar los datos y configurar sofisticados controles de acceso con objeto de satisfacer sus requisitos empresariales, organizativos y de conformidad. Amazon S3 está diseñado para ofrecer una durabilidad del 99,999999999 % (11 nueves) y almacena datos de millones de aplicaciones para empresas de todo el mundo.

Amazon S3 admite varios niveles de acceso a los datos con sus tarifas correspondientes. Es posible utilizar el análisis de clases de almacenamiento de S3 para detectar datos que se deben mover a una clase de almacenamiento más barata en función de los patrones de acceso, así como configurar una política de ciclo de vida de S3 para llevar a cabo la transferencia. También se puede almacenar datos con patrones de acceso cambiantes o desconocidos mediante la característica Capas inteligentes de S3, que dispone los datos en capas en función de patrones de acceso cambiantes y ofrece automáticamente un ahorro de costos.

S3 ofrece capacidades sólidas para administrar el acceso, el costo, la replicación y la protección de datos. Los puntos de acceso de S3 facilitan la administración del acceso a los datos con permisos específicos para las aplicaciones utilizando un conjunto de datos compartidos. Las funciones de replicación S3 administran la replicación de datos dentro de la región o de otras regiones. Las operaciones por lote en S3 ayudan a administrar los cambios a gran escala entre miles de millones de objetos. Dado que S3 funciona con AWS Lambda, los clientes pueden registrar actividades, definir alertas y automatizar flujos de trabajo, todo ello sin administrar ninguna infraestructura adicional.

S3 le ofrece capacidades sólidas para administrar el acceso, el costo, la replicación y la protección de datos. Los puntos de acceso de S3 facilitan la administración del acceso a los datos con permisos específicos para sus aplicaciones utilizando un conjunto de datos compartidos. [25]

3.10.2. AMAZON GLUE

AWS Glue es un servicio de extracción, transformación y carga (ETL) completamente administrado que puede utilizar para catalogar los datos, limpiarlos, completarlos y trasladarlos de manera fiable entre almacenes de datos. Con AWS Glue, puede reducir significativamente el costo, la complejidad y el tiempo dedicado a la creación de trabajos ETL. AWS Glue es un servicio sin servidor, por lo que no es necesario configurar ni administrar infraestructura. Solo paga por los recursos utilizados mientras se ejecutan sus trabajos.

El catálogo de datos de AWS Glue es su almacén de metadatos persistente para todos sus activos de datos, independientemente de dónde se encuentren. El catálogo de datos contiene definiciones de tablas, definiciones de trabajos y otra información de control para ayudar a administrar el entorno de AWS Glue. Procesa las estadísticas y registra las particiones automáticamente para realizar consultas en sus datos de manera eficaz y rentable. También mantiene un historial de versiones de esquemas exhaustivo para que pueda entender cómo han cambiado sus datos con el tiempo.

Los rastreadores de AWS Glue se conectan con cualquier almacén de datos de origen o de destino, avanzan a lo largo de una lista priorizada de clasificadores para determinar los esquemas para los datos y crean metadatos en el catálogo de datos de AWS Glue. Los metadatos se almacenan en tablas en el catálogo de datos y se utilizan en el proceso de autoría de todos los trabajos ETL. Puede ejecutar rastreadores (crawlers) de acuerdo con un programa, bajo demanda o activarlos en función de un evento para garantizar que los metadatos están actualizados.

AWS Glue genera automáticamente el código para extraer, transformar y cargar los datos. Simplemente es necesario apuntar AWS Glue hacia el origen y el destino de los datos, y el servicio creará secuencias de comandos ETL para transformar, acoplar y completar los datos. El código se genera en Scala o Python y se escribe para Apache Spark.

Este servicio ayuda a limpiar y preparar datos para análisis al proporcionar una transformación de aprendizaje automático que se llama FindMatches y sirve para deduplicar y encontrar registros coincidentes. Se puede utilizar FindMatches de AWS Lake Formation para encontrar registros duplicados en una base de datos de restaurantes, como, por ejemplo, cuando un registro dice "Pizza José" en "Calle 121 Main St." y otra muestra "Pizzería de José" en "121 Main". No es necesario que sepa algo de aprendizaje automático para poder hacer esto. FindMatches simplemente pedirá que se etiquete un conjunto de registros como "coincidente" o "no coincidente". Luego, el sistema aprenderá su criterio para determinar la coincidencia de dos registros y creará una transformación de aprendizaje automático que usted puede utilizar para encontrar los registros duplicados dentro de una base de datos o registros coincidentes entre dos bases de datos.

AWS Glue proporciona puntos de enlace de desarrollo para que se edite y pruebe el código que se genera y elimine errores. Puede usar cualquier IDE o bloc de notas preferido. Puede escribirse lecturas, escrituras o transformaciones personalizadas e importarlas en trabajos de ETL de AWS Glue como bibliotecas personalizadas. También se puede usar y compartir código con otros desarrolladores en un repositorio GitHub. [26]

3.10.3. AMAZON EMR

Amazon EMR es la plataforma para big data en la nube líder de Amazon destinada al procesamiento de grandes volúmenes de datos mediante el uso de herramientas de código abierto como Apache Spark, Apache Hive, Apache HBase, Apache Flink, Apache Hudi y Presto. Con EMR se puede ejecutar análisis a escala de petabytes a menos de la mitad del costo de las soluciones locales tradicionales y con una velocidad 3 veces superior que el Apache Spark estándar. Para trabajos de corta duración, es posible iniciar y finalizar clústeres y pagar por segundo en función de las instancias utilizadas. Para cargas de trabajo de larga duración, se puede crear clústeres de alta disponibilidad que se escalan automáticamente para satisfacer la demanda. Si se tiene implementaciones locales de herramientas de código abierto, tales como Apache Spark y Apache Hive, también se puede ejecutar clústeres de EMR en AWS Outposts.

A diferencia de la infraestructura rígida de los clústeres locales, EMR desacopla el cómputo y el almacenamiento, lo que brinda la capacidad para ajustar la escala de cada uno de forma independiente y aprovechar el almacenamiento en niveles de Amazon S3. Gracias a EMR, es posible aprovisionar una instancia de cómputo o cientos o miles de ellas para procesar datos a cualquier escala. Aumentar o reducir la cantidad de instancias automáticamente con Auto Scaling (que administra los tamaños de los clústeres en función del uso).

EMR establece automáticamente los ajustes del firewall de EC2 para controlar el acceso de red a las instancias y lanza clústeres en una Amazon Virtual Private Cloud (VPC). El cifrado del lado del servidor o el cifrado del lado del cliente se puede utilizar con AWS Key Management Service o las claves administradas por el cliente. EMR facilita la habilitación de otras opciones de cifrado, como el cifrado en tránsito y en reposo, y la autenticación sólida con Kerberos. De igual manera se puede usar AWS Lake Formation o Apache Ranger para aplicar controles de acceso a los datos específicos para bases de datos, tablas y columnas.

EMR se puede utilizar para realizar de forma rápida y rentable cargas de trabajo de transformación de datos (ETL), como por ejemplo ordenar, agregar e incluir, en conjuntos de datos de gran tamaño. Una de las herramientas de aprendizaje automático integradas de EMR, son Apache Spark MLlib, TensorFlow y Apache MXNet para los algoritmos de aprendizaje automático escalables, y las AMI personalizadas y las acciones de arranque para agregar bibliotecas y herramientas preferidas con facilidad y crear un propio conjunto de herramientas de análisis predictivo.

Con este servicio se tiene pleno control sobre cualquier clúster con acceso raíz a cada instancia. Se puede iniciar clústeres de EMR con las AMI de Amazon Linux personalizadas e instalar fácilmente aplicaciones adicionales con acciones de arranque. EMR permite reconfigurar aplicaciones en clústeres en ejecución sobre la marcha, sin la necesidad de reiniciar los clústeres. Además, con Hadoop 3.0, se puede empaquetar las dependencias de la biblioteca en contenedores Docker y enviarlas a personales trabajos para simplificar las dependencias del entorno. [27]

4. DESARROLLO METODOLÓGICO

Resumen: *En este capítulo se presenta en detalle el desarrollo metodológico relacionado a la infraestructura generada para el objeto de estudio, así como todo el proceso relacionado al procesamiento de la información.*

4.1. Extracción de información

Como todo nuevo proyecto, siempre se inicia con un primer paso. En este caso, para la realización del presente trabajo, y una de las motivaciones principales, fue el aspecto relacionado a la extracción de la información.

Tras indagar y analizar la información disponible por los diferentes organismos y proveedores de servicios de movilidad, fue fácilmente identificable como la estructuración, tipos de datos y almacenamiento de dicha información variaba de un organismo a otro, evidentemente, cada uno siguiendo los mejores lineamientos para poder ofrecer dicha información al público en general de una manera sencilla y confiable.

Tal y como se mencionó anteriormente, el presente trabajo se generó utilizando información que fuera un buen indicativo de inicio para poder llevar a cabo un análisis completo de la movilidad en las principales dos ciudades más importantes del país. De esta manera, toda la información utilizada en el presente trabajo se encuentra disponible en los portales de Internet de cada proveedor en específico.

La siguiente tabla muestra a detalle las direcciones generadas utilizadas para la descarda de los datos. Cabe mencionar que la información detallada (uso del servicio por mes y por año) se encuentra disponible en diferentes URLs por lo que la descarga se realizó de una manera paulatina y gradual ya que el peso de los archivos es bastante considerable.

Servicio	Página web de descarga
Bici GDL	https://datos.jalisco.gob.mx/search/type/dataset
Bici CDMX	https://www.ecobici.cdmx.gob.mx/es/informacion-del-servicio/open-data https://datos.cdmx.gob.mx/explore/?sort=modified&refine.keyword=movilidad
Uber GDL	https://movement.uber.com/explore/guadalajara/travel-times/
Uber CDMX	https://movement.uber.com/explore/mexico_city/travel-times/

Tabla 4. Links de descarga de información

Una vez descargada la información fue posible observar los diferentes formatos, así como las estructuras utilizadas por cada proveedor.

Para el caso de los datos referentes al uso de la bicicleta en Guadalajara, se tienen los registros agrupados por mes y por año en formato CSV en lo que a viajes individuales respecta además de contar con archivos .dbf, (el cual es un archivo de la base de datos **dBase**) donde se tienen los detalles de todas las estaciones instaladas para el aparcamiento de las bicicletas que no se están en uso.

Figura 5. Datasets sobre la ubicación de las estaciones del programa Mi Bici en la ZMG

De igual manera, los datos referentes a los viajes individuales realizados en bicicleta en la CDMX se encuentran agrupados por mes en formato .CSV, mientras que el archivo que contiene información relacionada a las estaciones se encuentra en formato tipo JSON. Adicionalmente, el gobierno del estado ofrece información sobre las ciclovías (datos que no se tienen disponibles por parte del programa Mi Bici en la ZMG), mismas que se encuentran en un formato tipo JSON.


```
{
    "datasetid": "estaciones-de-ecobici",
    "recordid": "09d4f3cf20ef0636a24567db330a99d0a9004e63",
    "fields": {
        "districtcode": "1",
        "nearbystations_0": 390,
        "name": "416 RECREO-PARROQU\u00cdA",
        "location_lon": -99.177607,
        "districtname": "Ampliaci\u00f3n Granada",
        "zipcode": "3104",
        "punto_geo": [19.371308, -99.177607],
        "stationtype": "BIKE",
        "addressnumber": "S/N",
        "address": "416 - Recreo-Parroqu\u00eda",
        "nearbystations_2": 415,
        "nearbystations_1": 391,
        "location_lat": 19.371308, "id": 416},
        "geometry": {
            "type": "Point",
            "coordinates": [-99.177607, 19.371308]},
            "record_timestamp": "2018-11-12T17:10:11.133-06:00"},
```


Figura 6. Ejemplo de almacenamiento de una estación de bici en la CDMX

En lo que respecta a la información almacenada por UBER, en este caso se tiene una agrupación de todos los viajes realizados por los usuarios en cuartos por año (enero-marzo, abril-junio, julio-septiembre, Octubre-Diciembre).

4.2. Arquitectura del proyecto

Con base en los lineamientos establecidos, los servicios ofrecidos por AWS y las características y fundamentos en los cuales un Data Lake se conforma, se planteó la siguiente arquitectura.

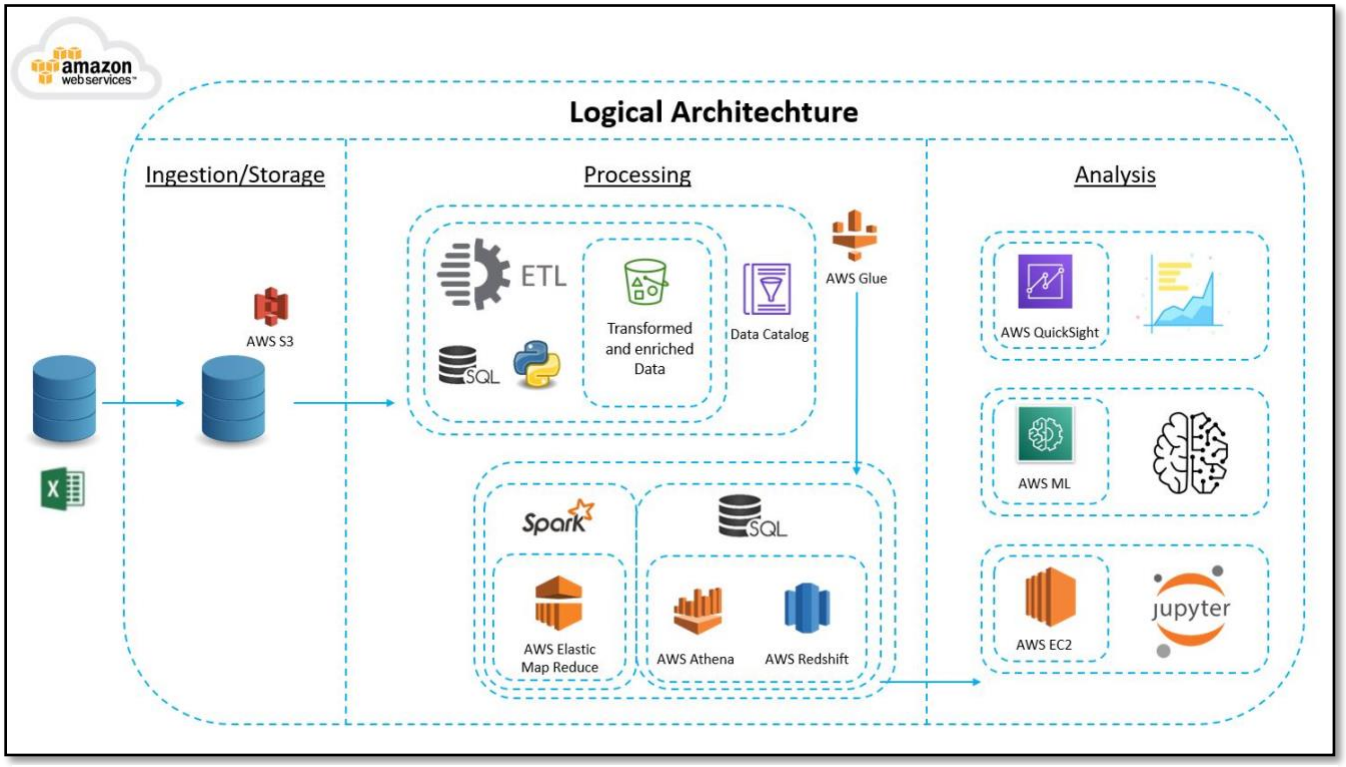


Figura 7. Arquitectura general del proyecto

La arquitectura propuesta está basada en 3 diferentes etapas, la primera de ellas corresponde a la ingesta de información, la cual, es la más sencilla de todas pues solo involucra la carga y almacenamiento de la información en el servicio de Amazon S3, el cual como ya se mencionó, es un servicio de almacenamiento distribuido el cuál presenta ciertas ventajas sobre el también posible HDFS, como lo es la escalabilidad, durabilidad, persistencia, la relación costo-eficiencia y su funcionamiento, además de ser un servicio totalmente administrado por Amazon lo que le añade cierta seguridad a su uso.

La siguiente etapa es la relacionada al procesamiento. Es aquí donde se lleva a cabo toda la integración y transformación de la información. Dentro de su composición, la información que es almacenada en S3 es posteriormente leída, analizada y transformada por procesos definidos de ETLs los cuales a su vez interactúan con los datos en su forma “cruda” para posteriormente llevar a cabo un procesamiento de los mismos a fin de que se pueda generar una base de datos, utilizando el servicio de Glue, con toda la información, ubicación y metadatos necesarios sobre los archivos almacenados en S3 para de esta manera poder lograr tener un acceso general y uniforme a cualquier tipo y formato de dato.

Con base en este catálogo es que las demás futuras etapas del proyecto se construyen y funcionan, pues todos los procesos generados de lectura de información serán a través de este catálogo y no directamente es S3, donde se tiene almacenada en forma física toda la información.

Dichos procesos se generan con base en tres servicios principales, los cuales son:

1. AWS Elastic MapReduce
2. AWS Athena
3. AWS Redshift

El primero de ellos será utilizado para generar procesos que requieran un uso y una integración completa de toda la información almacenada. Al tratarse de un servicio de Big Data, está diseñado para trabajar con base en la creación de clústeres que utilizan instancias de EC2 como motores de procesamiento en paralelo que permite obtener resultados de una manera muy rápida aún y que se trabaje con millones de registros.

El segundo de ellos se trata de un motor SQL para llevar a cabo consultas sencillas de la información almacenada, teniendo la posibilidad de generar estadísticas y reportes los cuáles no necesitan un procesamiento a detalle, por lo que es una manera sencilla y rápida para interactuar con la información.

Finalmente, AWS Redshift está integrado en la arquitectura como un servicio que fungirá como un servicio extra de almacenamiento en paralelo para poder tener un registro histórico de toda la información futura que sea generada, a fin de ofrecer una alternativa al servicio de S3, el cual está planeado para almacenar la información de interés.

La última etapa de la arquitectura planteada es la que corresponde al análisis de la información. Una vez que todos los datos son procesados e integrados es posible utilizarlos para poder llevar a cabo un análisis completo de la información a fin de generar conocimiento el cual pueda ser utilizado de muy distintas maneras.

En esta etapa se plantean 3 vertientes, la primera de ellas es la que corresponde a la creación y uso de reportes y *dashboards*, los cuales son comúnmente utilizados por analistas de negocio y altos mandos para formar y tomar decisiones sobre los servicios ofrecidos. La segunda hace referencia al uso del Machine Learning (ML) para poder generar conocimiento de una manera mucho más profunda, haciendo uso de distintos enfoques y algoritmos los cuales ofrecen soluciones que no sería posible general de una manera sencilla con otras tecnologías. Para este caso, no se hará uso de los servicios de Amazon, ya que no ofrecen las respuestas que el presente trabajo pretende genera, por lo contrario, todo el procesamiento se lleva a cabo por medio del lenguaje de programación Python en conjunto con el framework Spark, haciendo uso de algunas librerías como Numpy y MlLib.

Finalmente, se considera una posible tercera fase la cual pretende ofrecer una herramienta pre-configurada, la cual pueda ser usada por cualquier persona interesada en generar nuevo conocimiento a partir de la información almacenada y/o procesada, haciendo uso de la herramienta Jupyter Notebook.

4.3. Implementación en AWS

Una vez definida la arquitectura del proyecto, se procedió a trabajar con los servicios de cloud de AWS. Evidentemente para esta parte fue necesario contar con una cuenta personal para poder acceder a dichos servicios. Una vez que la cuenta estaba configurada correctamente, como primer paso, se llevó a cabo la etapa de la ingesta de datos.

4.3.1. S3

Como parte fundamental de la ingesta de información, fue necesario llevar a cabo la carga de todos los datasets descargados de las distintas plataformas por las que se compone el presente trabajo. Siguiendo las recomendaciones y lineamientos de AWS para poder hacer un mejor uso de la información almacenada en este servicio, se generó una estructura jerárquica de carpetas en donde cada data set se encuentra en una carpeta individual.

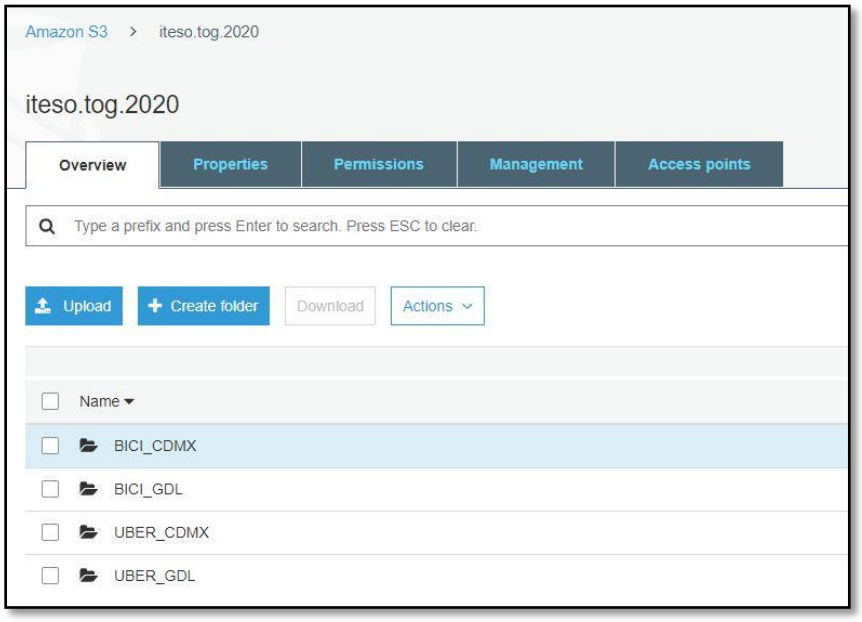


Figura 8. Nivel base de estructuración de carpetas en S3 para almacenamiento de datasets

Tal y como se aprecia en la imagen, antes de llevar a cabo la carga de archivos, fue necesario generar todas estas carpetas. Es posible identificar las carpetas para cada servicio, sin embargo, dentro de estas se generaron una para cada dataset en específico, ya que, de otra manera, la comunicación con Spark generaría problemas al compartir el mismo destino dos datasets diferentes.

Una vez definidas todas las carpetas, fue posible acceder a cada una de ellas para llevar a cabo la carga de datos en su respectivo espacio de almacenamiento generado. Dicha carga se llevó a cabo de una manera muy sencilla, pues la interfaz del servicio de S3 es bastante amigable y funcional.

Para dicho proceso, es necesario definir ciertos parámetros como el tipo de almacenamiento a utilizar (mismos que varían en tiempo de respuesta, costos, velocidad, etc.) en donde se definió un tipo *Estándar* el cuál es un balance entre costos y efectividad de lectura de información. Así mismo, fue necesario especificar los permisos de los datasets para de esta manera poder definir quién y cómo podrían acceder a los datos. Al tratarse de información que se encuentra pública sin ninguna restricción, dichos permisos fueron configurados sin ningún tipo de encriptación o seguridad especial (a diferencia de los datos generados en la etapa de procesamiento y análisis).

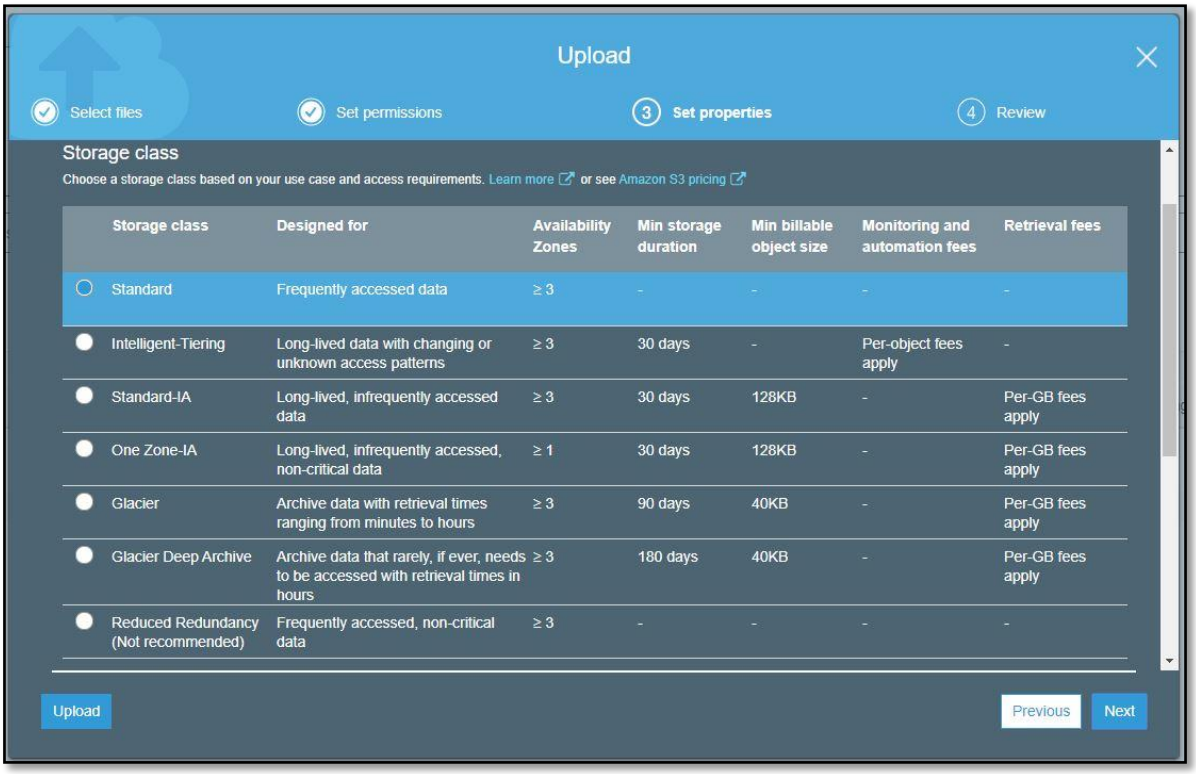


Figura 9. Configuración de propiedades para datasets cargados.

Finalmente, al tener todos los parámetros definidos, así como las configuraciones deseadas para los datasets, finalmente se procedió a aceptar y empezar a llevar a cargo el proceso de transferencia de archivos hacía S3.

Figura 10. Proceso de carga de archivos

4.3.2. Glue

La implementación del servicio de Glue en el proyecto presentó una parte importante, pues fue el servicio encargado de llevar a cabo la recopilación de todos los datasets previamente cargados en S3 para de esta manera poder generar un catálogo de metadatos el cual permitió tener un "mapeo" sobre toda la información a utilizar y procesar.

Para llevar a cabo dicha recopilación, fue necesario crear una base de datos en este servicio, la cual es la encargada de almacenar toda la información generada en forma de tablas.

Figura 11. Base de datos en Glue para almacenar la información sobre los datasets cargados en S3

Una vez definida lógicamente esta base de datos (pues AWS se encarga de todo el back-end por lo que no es necesario configurar absolutamente nada de lo que conlleva instalar una base de datos), el siguiente paso, y uno de los más importantes en lo que corresponde a este servicio, consistió en generar los procesos de ETL los cuales fueron los encargados de leer los datasets de S3 y generar las tablas de metadatos. Para llevar a cabo estos procesos, se generaron scripts utilizando Spark como herramienta de lectura y procesamiento de información.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
## @type: DataSource
## @args: [database = "tog_db_csv_tables", table_name = "csv", transformation_ctx = "datasource0"]
## @return: datasource0
## @inputs: []
datasource0 = glueContext.create_dynamic_frame.from_catalog(database = "tog_db_csv_tables", table_name = "csv", transformation_ctx = "datasource0")
## @type: ApplyMapping
## @args: [mapping = [("viaje_id", "long", "viaje_id", "long"), ("usuario_id", "long", "usuario_id", "long"), ("genero", "string", "genero", "string"), ("a�o_de_nacimiento", "long", "a�o_de_nacimiento", "long"),
## @return: applymapping1
## @inputs: [frame = datasource0]
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [("viaje_id", "long", "viaje_id", "long"), ("usuario_id", "long", "usuario_id", "long"), ("genero", "string", "genero", "string"), ("a�o_de_nacim
## @type: ResolveChoice
## @args: [choice = "make_struct", transformation_ctx = "resolvechoice2"]
## @return: resolvechoice2
## @inputs: [frame = applymapping1]
resolvechoice2 = ResolveChoice.apply(frame = applymapping1, choice = "make_struct", transformation_ctx = "resolvechoice2")
## @type: DropNullFields
## @args: [transformation_ctx = "dropnullfields3"]
## @return: dropnullfields3
## @inputs: [frame = resolvechoice2]
dropnullfields3 = DropNullFields.apply(frame = resolvechoice2, transformation_ctx = "dropnullfields3")
## @type: DataSink
## @args: [connection_type = "s3", connection_options = {"path": "s3://iteso.tog.2020/BICI_GDL/Viajes/2019/Enero/Parquet"}, format = "parquet", transformation_ctx = "datasink4"]
## @return: datasink4
## @inputs: [frame = dropnullfields3]
datasink4 = glueContext.write_dynamic_frame.from_options(frame = dropnullfields3, connection_type = "s3", connection_options = {"path": "s3://iteso.tog.2020/BICI_GDL/Viajes/2019/Enero/Parquet"}, format = "parquet"
job.commit()
```

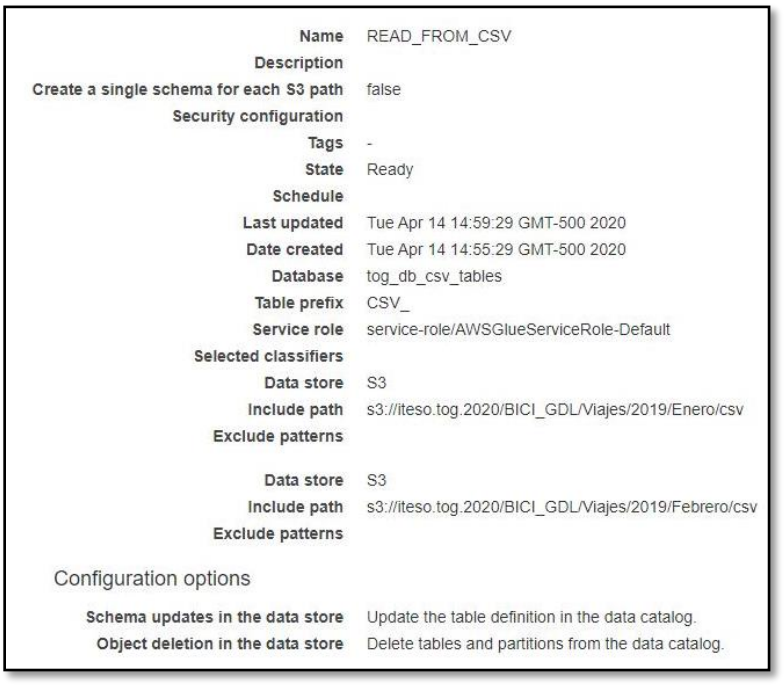
Figura 12. Extracto de script utilizado para la generación de tablas de metadatos en AWS Glue

Dentro de la lógica definida en el proyecto, y, con base en los formatos más comunes y utilizados en un *data lake*, se optó por generar dos bases de datos en el servicio de Glue. La primera para almacenar los metadatos sobre los datasets en un formato .CSV, y la segunda para un formato tipo PARQUET de los mismos. De esta manera, es posible tener dos opciones que pueden ser utilizadas según el caso de uso.

El formato tipo CSV es un tipo de dato comúnmente utilizado y con el que la mayoría de la gente está familiarizada, sin embargo, en cuestiones de procesamiento no presenta la mejor opción. Para este caso es que se optó por crear la segunda opción con base en un formato tipo parquet, el cual, al ser un tipo de dato columnar y binario, ofrece un mejor procesamiento y uso de la información. De esta manera, fue que se generaros dos procesos diferentes, ambos para leer la misma información, sin embargo, diferentes en la transformación de los tipos de datos.

Para llevar a cabo la ejecución de estos procesos, se hizo uso de *crawlers* (rastreadores según AWS) los cuales permiten la integración entre los distintos servicios así como la posibilidad de designarles una agenda de ejecución, para de esta manera tener un proceso automatizado el cual puede ejecutarse constantemente y de esta manera identificar los cambios en nuestro sistema de almacenamiento S3 para así agregar, actualizar y/o borrar datos el catálogo creado según se realicen estos en la fuente. Para la realización del presente trabajo, dichos crawlers fueron ejecutados *on-demand.*

Al tratarse de componentes que tienen una integración entre diferentes servicios, es necesario realizar ciertas configuraciones, como la especificación de la localización de los datasets, base de datos en donde insertar los resultados, roles de IAM para poder acceder a dichas ubicaciones, etc.



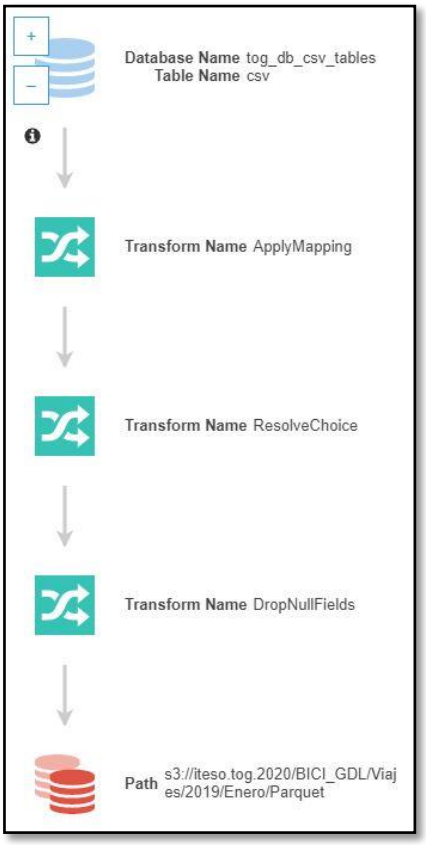
Name	READ_FROM_CSV
Description	
Create a single schema for each S3 path	false
Security configuration	
Tags	-
State	Ready
Schedule	
Last updated	Tue Apr 14 14:59:29 GMT-500 2020
Date created	Tue Apr 14 14:55:29 GMT-500 2020
Database	tog_db_csv_tables
Table prefix	CSV_
Service role	service-role/AWSGlueServiceRole-Default
Selected classifiers	
Data store	S3
Include path	s3://iteso.tog.2020/BICI_GDL/Viajes/2019/Enero/csv
Exclude patterns	
Data store	S3
Include path	s3://iteso.tog.2020/BICI_GDL/Viajes/2019/Febrero/csv
Exclude patterns	

Configuration options

Schema updates in the data store	Update the table definition in the data catalog.
Object deletion in the data store	Delete tables and partitions from the data catalog.

Figura 13. Configuración de parámetros relacionados al crawler que procesa archivos tipo CSV

Una vez completada la configuración del crawler así como el proceso a realizar en los datos a través de Spark, se procedió a generar una ejecución de los mismos. Tal y como ya se mencionó anteriormente, ejecución de los crawlers se realizó en la modalidad *on-demand*, dicha ejecución se compone de distintas fases, mismas que pueden verse en la consola que Glue pone a disposición del usuario.



Las primeras ejecuciones fueron fallidas, debido a errores y/o falta de librerías y dependencias en el código ejecutado por el servicio. Una gran ventaja que posee el servicio de Glue es que tiene una integración directa con Amazon CloudWatch el cual es un servicio de monitorización y observación creado para procesos de ejecución. CloudWatch ofrece datos e información procesable para monitorizar las aplicaciones, responder a cambios de rendimiento que afectan a todo el sistema, optimizar el uso de recursos y lograr una vista unificada del estado de las operaciones. CloudWatch recopila datos de monitorización y operaciones en formato de registros, métricas y eventos, lo cual ofrece una vista unificada de los recursos, las aplicaciones y los servicios de AWS que se ejecutan en servidores locales y de AWS.

Con este servicio, fue fácilmente posible obtener y analizar todos los errores que fueron generados por la ejecución de los scripts de replicación de datos

Figura 14. Fases de ejecución del crawler

Finalmente, una vez que todos los errores fueron corregidos, la ejecución del código, así como de las otras fases del proceso, se realizó de manera exitosa. Logrando de esta manera generar y llenar las tablas en el catálogo creado.

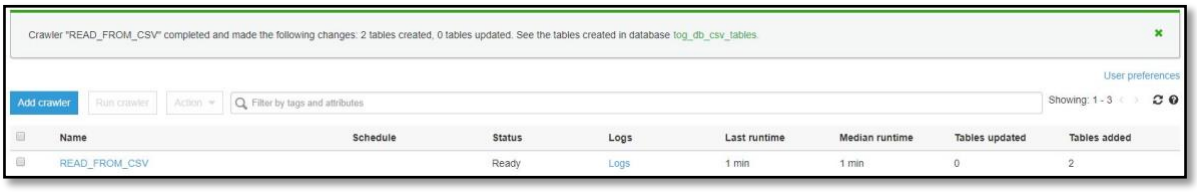


Figura 15. Consola de monitoreo para ejecuciones de crawlers

4.3.3. Data Catalog (Lake Formation)

La ejecución satisfactoria de los crawlers en Glue, permitió generar el catálogo de todos los datasets. Tal y como se mencionó previamente, se optó por tener dos bases de datos, una para los archivos csv y otra para parquet.

Figura 16. Bases de datos creadas en Lake Formation

Todos los archivos resultantes de las ejecuciones de los crawlers, fueron almacenados en estas dos bases de datos (según el formato generado). El servicio utilizado de Lake Formation facilita la configuración del data lake. El repositorio creado permite ser centralizado, seleccionado y seguro para almacenar todos los datos, tanto en su forma original como preparados para análisis. Por lo que, al tener toda esta información pre-cargada, es posible desglosar los silos de datos y combinar diferentes tipos de análisis para obtener información nueva.

Sin embargo, la configuración y administración del data lake implica muchas tareas manuales complejas y que llevan algo de tiempo. Este trabajo incluye cargar datos de diversas fuentes, monitorizar esos flujos de datos, configurar particiones, activar el cifrado y la administración de claves, definir trabajos de transformación y monitorizar su operación, reorganizar los datos en un formato de columnas, configurar los ajustes de control de acceso, deduplicar datos redundantes, relacionar registros vinculados, obtener acceso a conjuntos de datos y auditar el acceso a lo largo del tiempo.

Afortunadamente, una vez que todas estas tareas fueron completadas, la ejecución de los procesos pudo llevarse a cabo de una manera muy sencilla, por lo que cualquier otra persona que desee ejecutarlos manualmente puede hacerlo sin necesidad de llevar a cabo cambios complejos en la configuración del servicio y/o de los procesos.

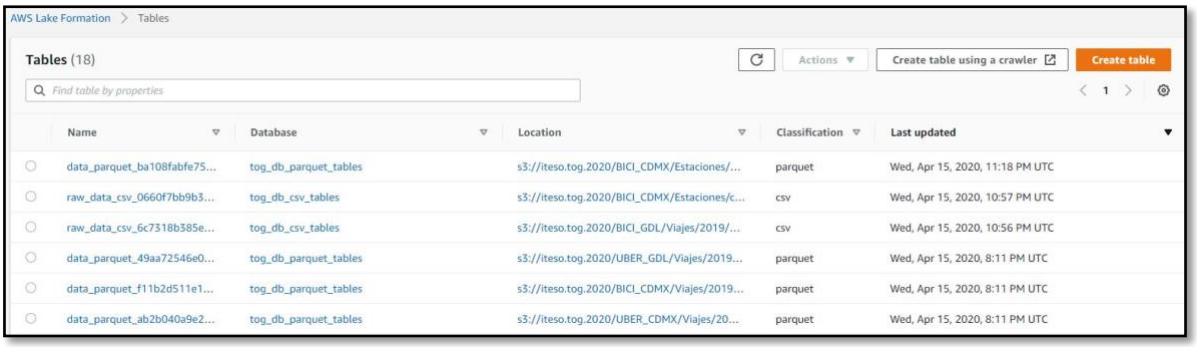


Figura 17. Tablas creadas por los jobs de ETL ejecutados en los crawlers

4.3.4. Athena

Una vez que el catálogo en Lake Formation fue creado con toda la información necesaria, tras la ejecución de los crawlers, se procedió a utilizar el servicio de Athena ya que es un servicio relativamente fácil de usar y configurar, teniendo una respuesta rápida sobre el proceso realizado anteriormente. Es un servicio de consultas interactivo que facilita el análisis de datos con SQL estándar. Ya que no fue necesario tener ni configurar un servidor, fue posible evitar administrar infraestructura específica.

Este servicio fue muy sencillo de utilizar. Simplemente fue necesario configurar la fuente de donde los datos van a ser leídos, que en este caso fueron las tablas de Lake Formation, por lo que una vez definido el *Data Source* y el esquema, fue posible comenzar a realizar consultas con SQL estándar. La mayoría de los resultados se proporciona en cuestión de segundos. No es necesario realizar trabajos complejos de ETL para preparar los datos para el análisis ya que todo esto fue realizado previamente con el servicio de Glue. Por ello, cualquier persona con habilidades SQL puede analizar conjuntos de datos a gran escala de forma rápida y sencilla.

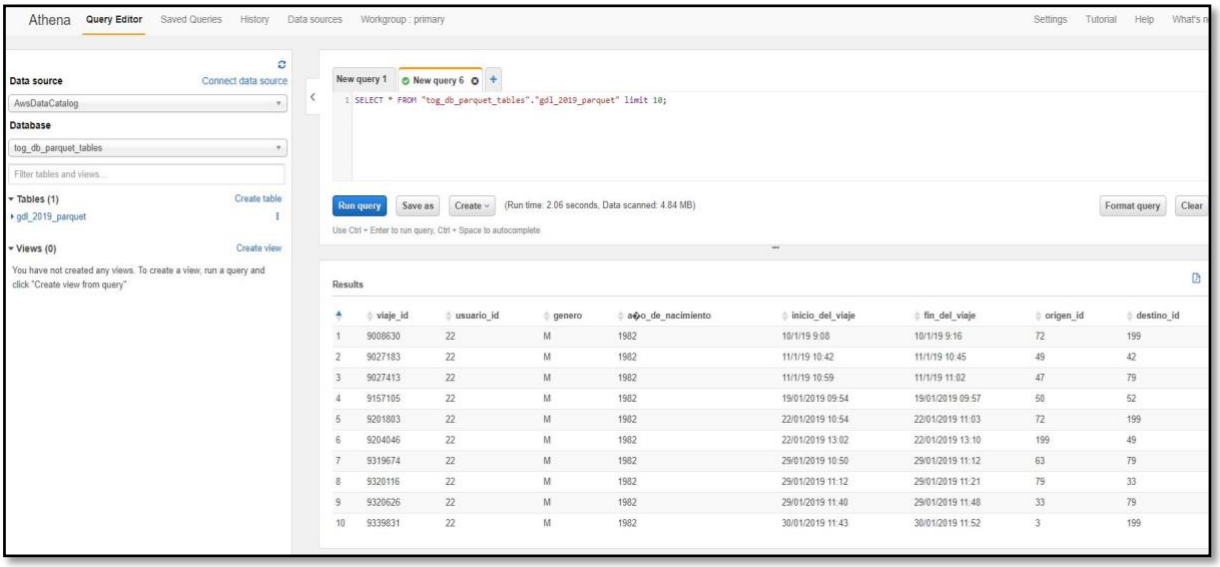


Figura 18. Consulta de información en Athena de las tablas creadas con Glue

La integración que este servicio tiene de serie con el catálogo de datos de AWS Glue, permitió utilizar el repositorio de metadatos unificado en diversos servicios. Además de rastrear orígenes de datos para descubrir esquemas y completar dicho catálogo con definiciones de particiones y tablas nuevas y modificadas, y mantener el control de las versiones de los esquemas. Para este proyecto en específico, no fue necesario llevar a cabo dichas transformaciones, pues al ser datasets muy grandes (GB) el costo por dicho uso hubiera aumentado considerablemente. Para llevar a cabo todas estas trasformaciones, se optó por utilizar un servicio de procesamiento en paralelo como lo es Elastic MapReduce

4.3.5. Elastic MapReduce

A diferencia de Athena, el uso del servicio de MapReduce, requirió mucho más esfuerzo y dedicación. Dado que es necesario tener un conocimiento amplio sobre las tecnologías de Big Data, en especial con la cartera de productos de Apache, pues son los que se utilizan principalmente para configurar el clúster con el cual se pretende trabajar.

Antes de entrar en detalles con la configuración y manejo del cluster, es necesario primero trabajar en los *jobs* que dicho clúster va a ejecutar. Para la realización de este trabajo, se optó por utilizar el framework de Spark, mismo que puede ser ejecutado en este servicio sin ningún problema. Dichos jobs en Spark son aquellos donde se hace el manejo y las transformaciones de todos los datasets en conjunto a fin de generar un procesamiento mucho más rápido y eficaz que con cualquier otra tecnología disponible.

El enfoque principal del proyecto es aquel relacionado a la integración de información entre las distintas fuentes de las cuales se extrajeron los datos, dada esta circunstancia es que el primer job desarrollado en Spark fue aquel encargado de leer todos los datasets cargados para integrarlos y unificarlos en uno solo. Tal y como se mencionó previamente, dentro de los datasets, se tenían archivos en formato .CSV, .JSON y .DFB por lo que el job desarrollado fue creado para trabajar con esta problemática. Todo el proceso de integración y generación de resultados se llevó a cabo utilizando la API de Python en donde se generaron scripts (véase un ejemplo en la imagen inferior) mismos que fueron almacenado en S3 y posteriormente leídos por EMR para ser ejecutados en el cluster.

```
march_travels_df = read_csv(spark, "s3://iteso.tog.2020/BICI_CDMX/Viajes/2019/Marzo/csv/Marzo.csv")
#Removing useless column in march data frame
marchProc_travels_df = march_travels_df.drop('_c9')

schema = StructType([
StructField("datasetid", StringType(), True),
StructField("recordid", StringType(), True),
StructField("fields", StructType([
        StructField("districtcode", StringType(), True),
        StructField("nearbystations_0", IntegerType(), True),
        StructField("name", StringType(), True),
        StructField("location_lon", DoubleType(), True),
        StructField("districtname", StringType(), True),
        StructField("zipcode", StringType(), True),
        StructField("punto_geo", ArrayType(DoubleType()), True),
        StructField("stationtype", StringType(), True),
        StructField("addressnumber", StringType(), True),
        StructField("address", StringType(), True),
        StructField("nearbystations_2", IntegerType(), True),
        StructField("nearbystations_1", IntegerType(), True),
        StructField("location_lat", DoubleType(), True),
        StructField("id", IntegerType(), True),
]), True),
StructField("geometry", StructType([
        StructField("type", StringType(), True),
        StructField("coordinates", ArrayType(DoubleType()), True),
]), True),
StructField("record_timestamp", StringType(), True)
])

stations_df = spark.read.schema(schema).json("s3://iteso.tog.2020/BICI_CDMX/Estaciones/json/Estaciones.json")
stations_columns_df = stations_df.select('fields.*')

firstQtr_travels_df = january_travels_df.union(february_travels_df).union(marchProc_travels_df)
full_info_travels_df = firstQtr_travels_df.join(stations_columns_df, firstQtr_travels_df.Ciclo_Estacion_Retiro == stations_columns_df.id)

full_info_travels_df.registerTempTable("CDMX_TRAVELS")


mostUsedStations = spark.sql("""SELECT CICLO_ESTACION_RETIRO, DISTRICTNAME, COUNT(CICLO_ESTACION_RETIRO) AS TIMES_USED
                                                        FROM CDMX_TRAVELS
                                                        GROUP BY CICLO_ESTACION_RETIRO, DISTRICTNAME
                                                        ORDER BY TIMES_USED
```

Figura 19. Extracto de código utilizado en el procesamiento de archivos .JSON

Dentro de la API de SQL en Spark, se cuenta de igual manera otras más que fueron parte importante como lo es la utilizada para la implementación de dataframes (*Structured API*) así como la librería de SQL.

Dada la diferencia de formatos con los que se contaba, fue necesario hacer uso de estas funcionalidades de Spark para poder llevar a cabo una unificación de datos. El uso de las APIs mencionadas anteriormente, permite hacer una lectura de estos archivos y posteriormente realizar una transformación hacía lo que es conocido como un *data frame.*

Los data frames son estructuras de datos de dos dimensiones (rectangulares) que pueden contener datos de diferentes tipos, por lo tanto, son heterogéneas. Esta estructura de datos fue primeramente desarrollada en lenguaje de programación R, sin embargo, posteriormente fue adquirida por la librería Pandas en Python y, de igual manera, en Spark. Podemos entender a los data frames como una versión más flexible de una matriz. Mientras que en una matriz todas las celdas deben contener datos del mismo tipo, los renglones de un data frame admiten datos de distintos tipos, pero sus columnas conservan la restricción de contener datos de un sólo tipo.

Fue utilizando data frames (como tipo de dato lógico) que se llevó a cabo la integración de los datos, haciendo una conversión implícita de todos los datasets .CSV, .JSON y .DFB. Dado que un archivo JSON contiene información en distintos "niveles", a diferencia de los CSV y DFB que mantienen una estructura relacionada a filas y columnas, fue necesario llevar a cabo una definición del "esquema" (mismo que se aprecia en la imagen anterior) que los archivos tenían, para de esta manera poder indicarle a Spark la manera en que los datos debían ser leídos. Para los casos de los archivos .CSV y .DFB no fue necesario realizar este paso por la estructura utilizada por estos formatos.

La transformación de los archivos a data frame solo se realiza en tiempo de ejecución del job de Spark en el clúster, y no es un indicativo que los datos permanecerán en este formato una vez que el job haya acabado. Sin embargo, los resultados de los procesamientos que se hayan realizado durante la ejecución del job si pueden ser almacenados en un solo formato especificado. Es por esta razón que Spark es una herramienta muy poderosa al tratar con archivos y datos de diferentes tipos.

Una vez generados los data frames a utilizar, se llevaron a cabo diferentes tareas dentro de los distintos jobs ejecutados, mismos que serán discutidos en la parte de "resultados y discusión" del presente trabajo.

Es importante mencionar que, para testear los jobs generados (utilizando solo un subset de los datos) se hizo uso de los servicios de *Databricks Community Edition,* el cual es una plataforma unificada de análisis de datos donde se hace uso de un *notebook* donde se tiene ya instalado Spark con todas sus funcionalidades. Únicamente es necesario crear un "clúster" (mismo que solo se compone de una sola instancia de Amazon EC2) y vincularlo con el notebook recién creado. De esta manera se tiene una herramienta donde se puede testear todos los jobs generados sin tener que llevar a cabo una compleja instalación local de Spark.

Una vez que los jobs fueron creados y testeados, el siguiente paso fue el correspondiente a la configuración del clúster en Elastic MapReduce (EMR).

La consola de configuración del clúster presenta dos opciones para la creación de este. La primera es un modo "Automático" donde solo es necesario definir el número de instancias y otros detalles generales, sin embargo, dado que se pretendía hacer uso de configuraciones más detalladas, se optó por la segunda opción, la cual ofrece una configuración mucho más completa y detallada del clúster a generar.

De esta manera primero se llevó a cabo la configuración de los componentes a utilizar en el clúster, mismos que se tienen identificados por AWS en la parte de *Software Configuration.* Para poder hacer uso de la API de Python en Spark, es necesario generar una configuración de comando detallada, misma que puede verse a continuación.

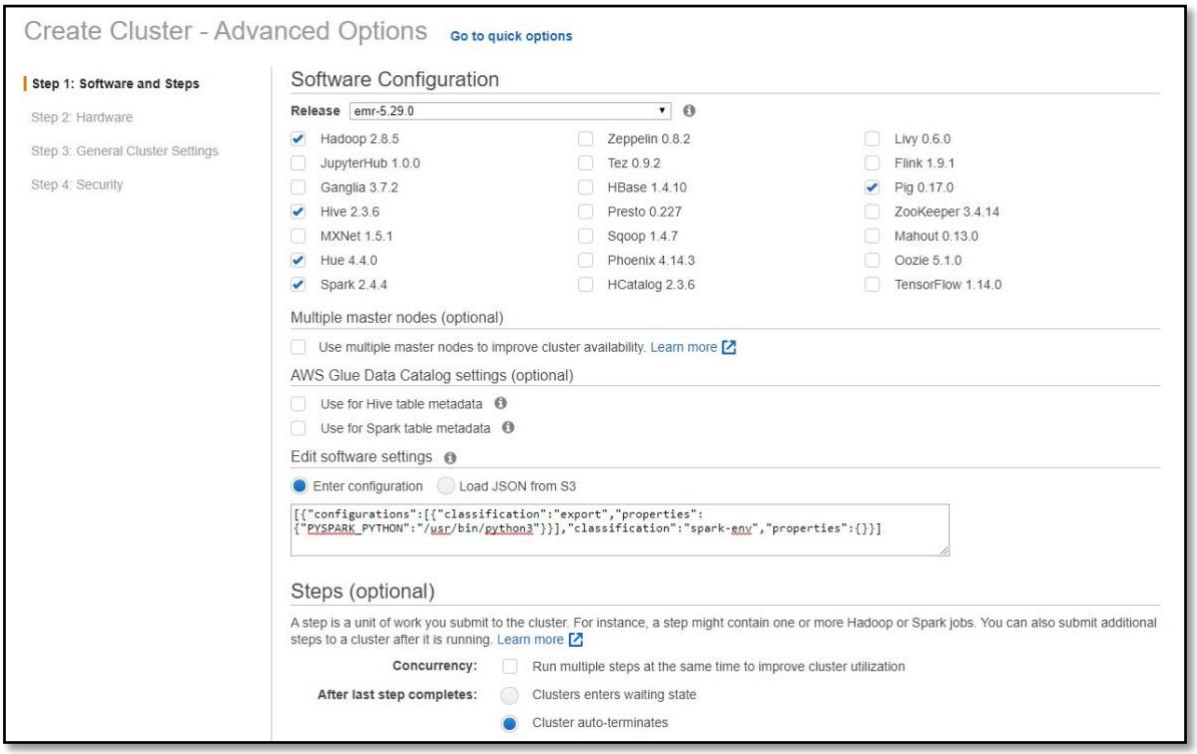


Figura 20. Configuración del software a instalar en el clúster

De igual manera, es en este apartado donde se indica a EMR la ubicación de los scripts, generados y almacenados en S3, que van a ser ejecutados por el clúster, así como los comandos de Spark a ser utilizados cuando se inicie la ejecución de estos. Los comandos utilizados en este caso fueron el modo de ejecución de Spark así como el número de *cores* a utilizar. Finalmente es posible configurar la respuesta del clúster en caso de falla, donde se optó por terminar (por cuestiones de ahorro) el clúster si alguna falla se presentaba.

En lo que respecta a la configuración del hardware del clúster, y teniendo en cuenta los costos por la creación del mismo, se definió con 3 instancias de EC2 tipo m5.xlarge las cuales cuenta con 4vCore, 16 GB de RAM con un almacenamiento de 64 GB. Una instancia dedicada para el *Master Node* y las otras dos para los Worker Nodes, mismas que fueron "adquiridas en la modalidad" *On-demand* para de esta manera contar con los mismos hosts. En lo que respecta a la configuración de la red utilizada (VPC), se utilizó la misma red (la cual fue creada específicamente para el proyecto) dejando las instancias ejecutadas en la misma *Availability Zone* (AZ).

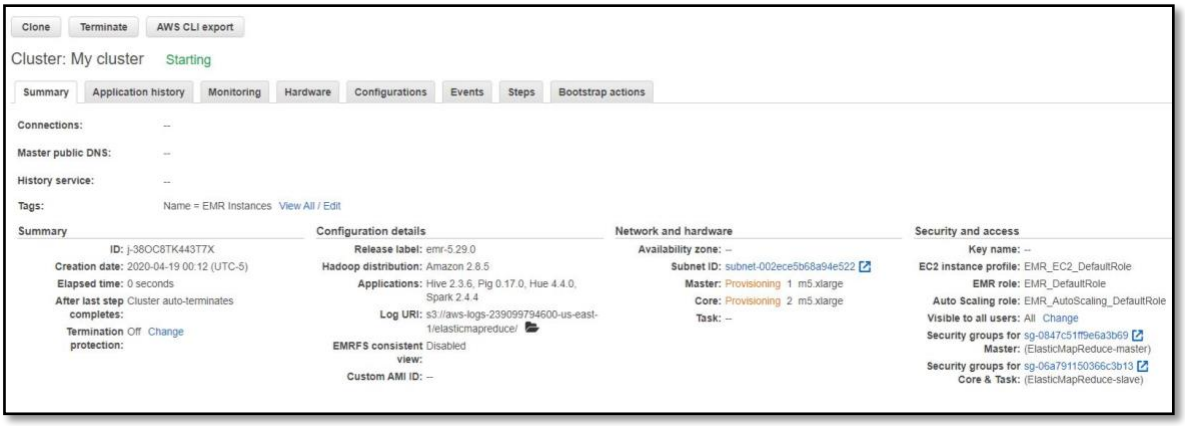


Figura 21. Configuración general del clúster creado en EMR

Finalmente, una vez definida toda la configuración necesaria para la creación del clúster, se procedió a llevar a cabo su ejecución. Evidentemente al tratarse de un componente tan complejo, el cual hace uso de distintos servicios para su creación y, además, lleva a cabo la instalación de todo el software especificado, fueron necesarios alrededor de 10 minutos para que este completara y estuviera en estado funcional. Mientras la instalación del clúster es llevada a cabo, se tiene la posibilidad de identificar todos los servicios que se requieren para su creación, por lo que no solo se tiene un componente abstracto, sino que es posible hacer uso de los componentes que lo forman. Un ejemplo claro para esto puede verse en el servicio de EC2, en donde se muestran todos los detalles de las instancias creadas, por lo que de igual manera se tiene la posibilidad de acceder y llevar a cabo configuraciones externas en cualquiera de estas, como si se tratase de una instancia independiente.

Figura 22. Instancias creadas en EC2 por EMR

Una vez que el clúster fue creado, se tuvieron algunos errores en el script de Python ejecutado, específicamente en la configuración utilizada para almacenar todos los archivos creados durante la ejecución del clúster. Afortunadamente, EMR ofrece la posibilidad de "clonar" un clúster previamente creado, permitiendo de esta manera, evitar tener que llevar a cabo toda la misma configuración realizada la primera vez.

Una vez corregidos todos los errores subsecuentes (la mayoría generados por falta de conocimiento del servicio), fue posible llevar a cabo una ejecución exitosa. Teniendo la posibilidad de tener una consola de monitorización del proceso en tiempo real.

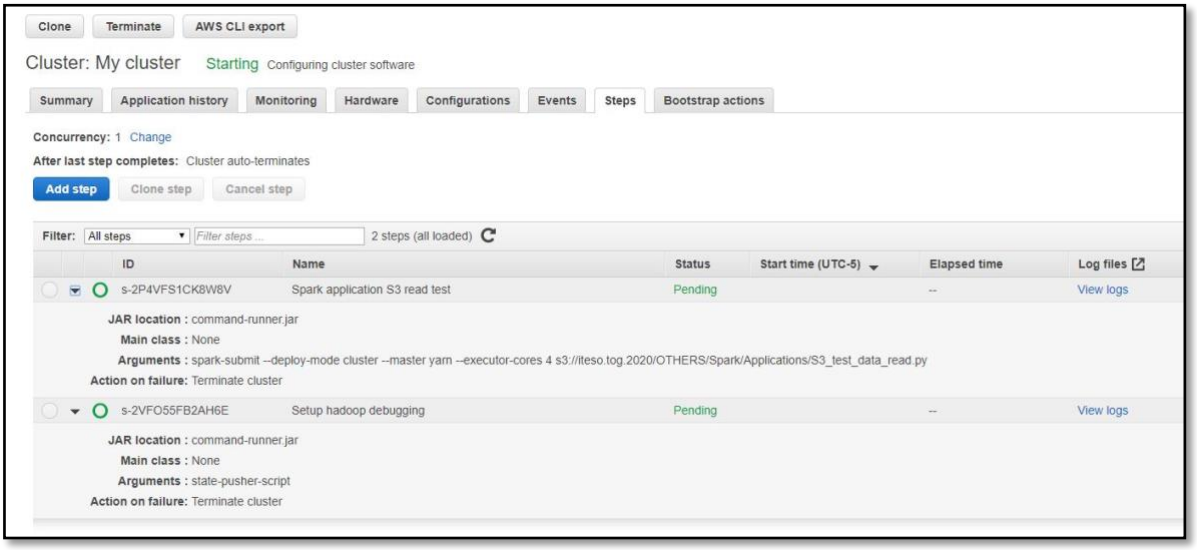


Figura 23. Ejecución del clúster en tiempo real por EMR

Todos los archivos y resultados obtenidos fueron los esperados en un lapso muy aceptable, mismos que serán discutidos a detalle en el siguiente capítulo.

Por último, tal y como se le configuró al clúster, una vez que todos los jobs fueron ejecutados y ninguna acción programada quedó pendiente, el clúster llevo a cabo su ejecución de manera automática, lo que permitió un ahorro significativo de costos en el supuesto caso de que éste siguiera activo con todos los servicios creados en continua ejecución.

4.3.6. Redshift

Como última etapa de la fase de procesamiento establecida para el proyecto, se tiene aquella relacionada al almacenamiento histórico de los datos.

Dicho almacenamiento presenta una etapa importante en el proyecto dado que actualmente el tamaño de los datos registrados y analizados no presenta una fecha de término, dado la movilidad dentro de las dos ciudades continuará en asenso conforme la población lo haga, y por tal motivo, de igual manera cada vez más será mayor el número de información generada en este ámbito social. Por tal motivo, tener la posibilidad de utilizar un *Warehouse* permitirá a los organismos responsables de este aspecto social el poder aplicar múltiples funciones a los datos históricos por lo que da la capacidad de predecir situaciones futuras en diferentes escenarios. Así pues, facilita la comunicación entre departamentos con información centralizada ya que se basa en datos integrados y globales, por lo que se favorece a la toma de decisiones en cualquier área y, da una mayor rapidez a la hora de hacer consultas y acceder a la información;

Por otra parte, otorga una optimización tecnológica de información, estadísticas y de generación de informes con excelentes retornos de la inversión, facilitando la aplicación de técnicas de análisis estadísticas y de modelización para encontrar las relaciones ocultas entre los datos en este servicio.

Dado este concepto, es que se hizo uso del servicio Redshift de AWS, en donde es posible ejecutar consultas de alto rendimiento de muy grandes cantidades de datos estructurados de forma simple y rentable, lo que permite utilizar los elementos de la fase de análisis sin ningún problema, así como también se tiene la capacidad de crear informes y paneles eficientes mediante el uso de herramientas de inteligencia empresarial existentes.

Así mismo, se tiene la posibilidad de poder combinar datos estructurados y datos semiestructurados en el *Warehouse*, como registros de aplicaciones, del data lake para obtener información operativa en tiempo real acerca de aplicaciones y sistemas.

Al realizar la configuración y creación de dicho servicio, se tuvieron únicamente problemas en lo referente al acceso que el servicio de Redshift tiene hacía los componentes de los otros servicios utilizados en el proyecto. Tras llevar a cabo un poco de investigación, fue posible ver como este problema era generado por la falta de un role que tenga todas las políticas de acceso. Dado que los roles existentes presentaban variaciones significativas unos con otros, se optó por crear un role personalizado específicamente para otorgar el acceso de Redshift hacía los servicios utilizados en el proyecto.

Figura 24. Role en IAM creado para la completa integración de Redshift

Al configurar y crear dicho role, lo único restante por hacer, es llevar a cabo la creación del warehouse para el almacenamiento de información. Dicha creación resulta prácticamente sencilla por medio de la interfaz que Amazon pone a disposición del usuario. Al tratarse básicamente de una base de datos distribuida, es necesario llevar a cabo la configuración de ciertos aspectos generales que componen una base de datos. Por cuestiones de presupuesto limitado y dado que actualmente no se tenía el registro de un dataset tan grande (Petabytes) se optó por elegir una configuración basada únicamente en dos nodos con 320 GB de almacenamiento tipo SSD.

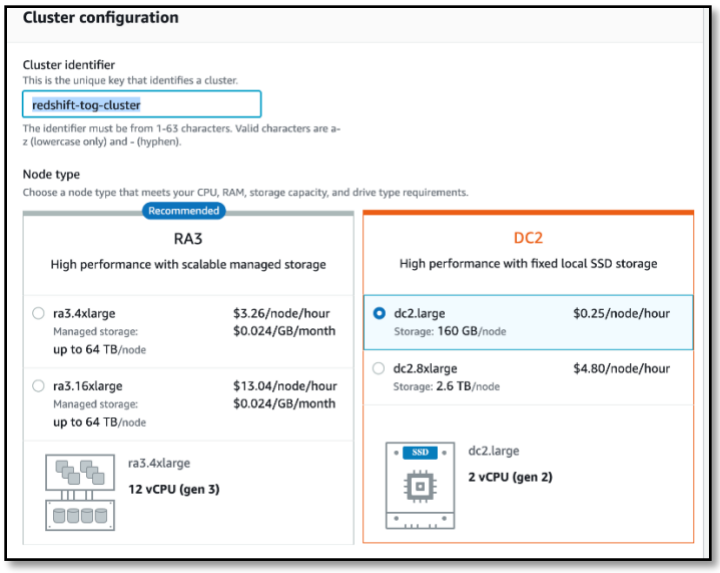


Figura 25. Configuración de nodos del warehouse

Es importante tomar en cuenta que dicha "base de datos" conlleva la creación de mínimo un usuario que tenga acceso a la misma, además de ciertos protocolos de comunicación como lo es el puerto por el que se definirá dicho proceso. Una vez terminada la configuración, es posible realizar un completo seguimiento sobre su creación a través de la consola de AWS, donde de igual manera se tiene herramientas que constante monitorean el rendimiento de su CPU, así como otros factores detallados.

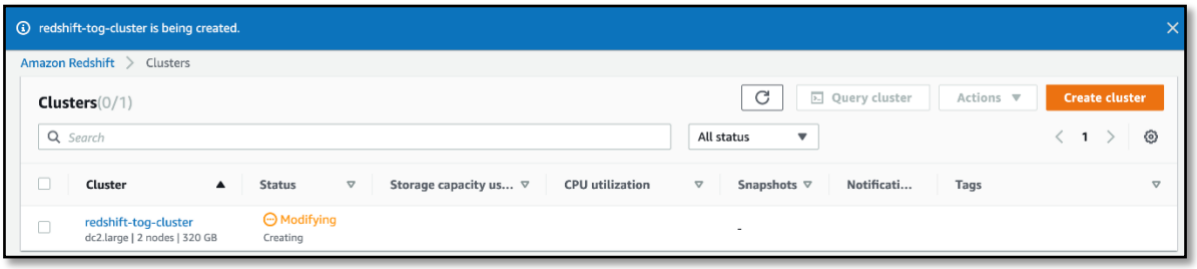


Figura 26. Consola de administración del clúster

La carga de información se hace de manera muy sencilla a través de un simple comando SQL en el editor que se tiene de este servicio. [28]

```
COPY table_name [ column_list ] FROM data_source
```

4.4. Análisis de la información con QuickSight

El inicio de la tercera fase diseñada del proyecto empieza justamente en este subtema, misma que de igual manera se trata en el capítulo de “Resultados y Discusión”. Sin embargo, los aspectos relacionados a la configuración e implementación de los procesos y métodos de análisis se discuten en este apartado.

Tal y como se mencionó previamente, un data pipeline tiene como principal objetivo el de generar y procesar la información de todas las maneras necesarias a fin de que esta pueda ser analizada. Uno de los usos más comunes es el referente al que hace uso y a la vez depende de la escritura de consultas “ad hoc” en SQL para analizar los datos y la información. La arquitectura sin servidor con precios de pago por sesión de QuickSight permite proporcionar esta información a todas las personas involucradas en el análisis. Con este servicio se puede compartir paneles integrales e interactivos con todos los usuarios interesados, que les permiten desglosar y explorar los datos para responder a preguntas y obtener la información pertinente de una manera muy sencilla.

El servicio de QuickSight, a diferencia de la gran mayoría de servicios de Amazon, requiere una subscripción independiente de la cuenta utilizada en AWS. Una vez que se completa dicha subscripción, se tiene la posibilidad de llevar a cabo la configuración del servicio, en donde básicamente es necesario ingresar un usuario y correo electrónico, para finalmente seleccionar los servicios a los cuales se tendrá acceso. Algo muy parecido a lo que se mencionó en el servicio de Redshift, sin embargo, para este no fue necesario llevar a cabo la creación de un role en específico. Todos los accesos se configuran a través de la consola propia de QuickSight.

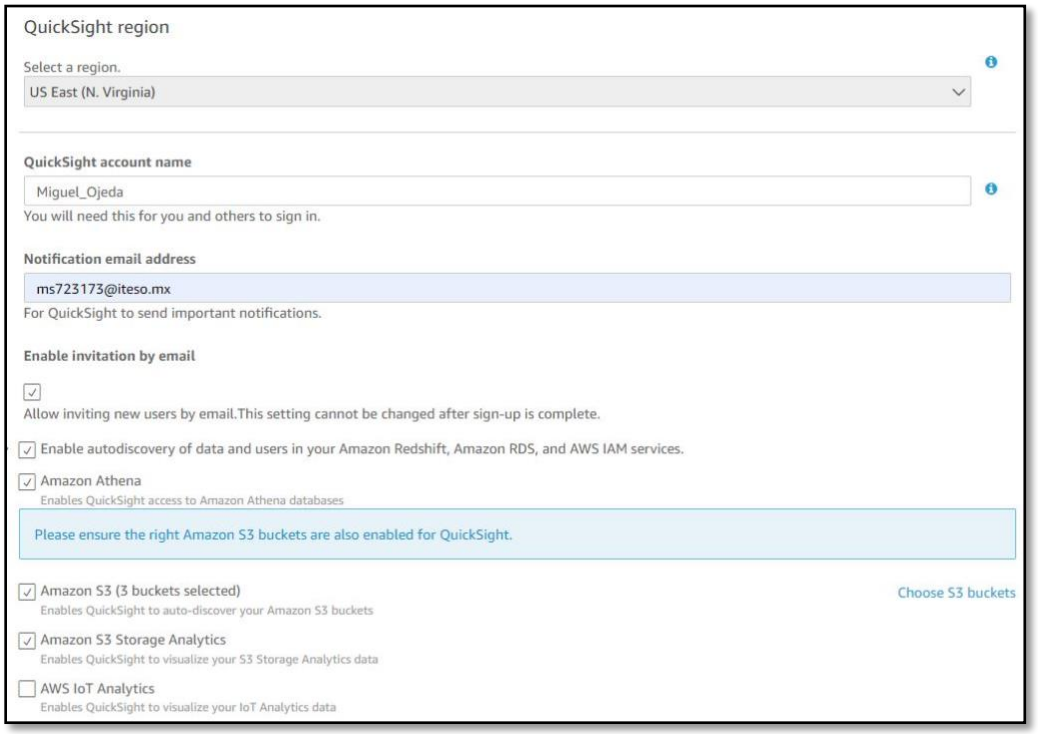


Figura 27. Consola de configuración de Amazon QuickSight

Uno de los aspectos más importantes en el ámbito del análisis de información por parte de los analistas, es el que corresponde a la posibilidad de tener una herramienta que, además de generar reportes, también tenga la capacidad de generar gráficos, los cuales presentan la información de una manera mucho más ilustrativa, los cuales, muchas veces generan conocimiento específico que sería difícil de obtener por otros medios.

Fue principalmente por esta razón que se hizo un énfasis especial en esta herramienta pues permite integrar fácilmente visualizaciones y paneles interactivos en aplicaciones y portales web. Cuando los usuarios obtienen acceso a los paneles, se paga únicamente en función del uso real que se hace de ellos por lo que no es necesario adquirir una licencia por un determinado tiempo, en donde no se toma en cuenta el uso real que se hace de la aplicación, lo que sin duda presenta un beneficio de costos. Se puede crear y publicar paneles y, después, integrarlos en aplicaciones con el inicio de sesión único y las API mediante los SDK de AWS y QuickSight. Realizar cambios en los paneles activos puede llevarse a cabo en cuestión de minutos, sin necesidad de escribir código y sin procesos de implementación complicados. [29]

Con el uso de QuickSight se puede utilizar una variedad de orígenes para análisis de datos, incluidos archivos, servicios de AWS y bases de datos en las instalaciones. La etapa previa de procesamiento de datos permite crear análisis de una manera mucho más confiable, por lo que se debe crear conjuntos de datos basados en los orígenes de estos. Un conjunto de datos identifica los campos y filas específicos que se desea utilizar. Además de los datos, un conjunto guarda los cambios que se realicen, por lo que se estará preparado la próxima vez que se deseen analizar dichos datos. Por ejemplo, se puede cambiar el nombre de campos, cambiar los tipos de datos y añadir campos calculados.

Un análisis de datos es el espacio de trabajo básico para crear e interactuar con elementos visuales que son representaciones gráficas de los datos. Cada análisis incluye un conjunto de elementos visuales que se deben reunir y organizar para distintos fines, como un análisis de problemas, un análisis de costos o un seguimiento de indicadores clave de desempeño dentro de los distintos servicios de movilidad. Cada análisis puede contener historias, que se pueden utilizar para guardar una presentación secuencial de iteraciones diferentes del análisis. Esto resulta útil si desea mostrar los cambios a lo largo del tiempo o proporcionar comparaciones visuales de los datos.

La primera vez que se crea un análisis, el flujo de trabajo típico se podría definir de la siguiente manera:

1. Añadir o cargar un origen de datos a fin de utilizarlo para crear un conjunto de datos.
2. (Opcional) Preparar los datos. Se deben tener preparados para los informes mediante la estandarización de nombres de campo o la adición de cálculos.
3. Visualización (o creación) de un nuevo análisis a partir del conjunto de datos.
4. Elección de algunos campos para crear el primer elemento visual en el análisis. Es posible utilizar AutoGraph para crear dinámicamente un elemento visual basándose en el número y el tipo de campos elegidos. Como alternativa, se puede elegir el tipo de elemento visual que se desea utilizar.
5. (Opcional) Realizar cambios en el elemento visual si así se desea (por ejemplo, añadir un filtro o cambiar el tipo de elemento visual).

4.5. Análisis de la información con Machine Learning

Finalmente, como última etapa de la fase de análisis, se lleva a cabo la implementación de Machine Learning para realizar un análisis más profundo sobre la información almacenado y procesada.

Todos los análisis y procesamientos de información extras se realizaron utilizando Spark, mediante la API de Python. Con esta API fue posible llevar a cabo la construcción de diferentes scripts en donde se llevaron a cabo diferentes acciones. Sin embargo, todos fueron generados siguiendo un camino y estructura similar, en donde como primera etapa se procede a cargar los datos dentro de Spark, posteriormente se lleva a cabo la integración de estos a fin de obtener un solo dataset sobre el cual se pueda llevar a cabo todas las transformaciones necesarias. Una vez unificado el archivo, se lleva a cabo el particionamiento de información en todos los ejecutores con base en las características del clúster utilizado para finalmente llevar a cabo las transformaciones necesarias.

Dentro de las opciones que se tienen en Spark para utilizar Machine Learning, se hizo uso básicamente de 3; *Spark MlLib*, *Sklearn* y *Numpy*. Estas tres librerías fueron las utilizadas dentro del procesamiento de la información.

Tal y como ya se mencionó, la primera etapa fue aquella correspondiente a la carga de información, por lo que los primero *jobs* realizados fueron aquellos que se encargaron de este paso.

```
from pyspark.sql import SparkSession, Row
from pyspark.sql.window import Window
from pyspark.sql.functions import row_number


def get_spark_session():
    return SparkSession.builder.appName("DATA_PROCESSING").getOrCreate()

def read_csv(spark, path):
    return spark.read.format("csv").option("header", "true").option("inferSchema", "true").load(path)

def save_df(df, path):
    return df.coalesce(1).write.option("header", "true").csv(path)


def main():

    january_details_df.createOrReplaceTempView("JANUARY_DETAILS")
    february_details_df.createOrReplaceTempView("FEBRUARY_DETAILS")
    march_details_df.createOrReplaceTempView("MARCH_DETAILS")
    april_details_df.createOrReplaceTempView("APRIL_DETAILS")
    may_details_df.createOrReplaceTempView("MAY_DETAILS")
    june_details_df.createOrReplaceTempView("JUNE_DETAILS")
    july_details_df.createOrReplaceTempView("JULY_DETAILS")
    august_details_df.createOrReplaceTempView("AUGUST_DETAILS")
    september_details_df.createOrReplaceTempView("SEPTEMBER_DETAILS")
    october_details_df.createOrReplaceTempView("OCTOBER_DETAILS")
    november_details_df.createOrReplaceTempView("NOVEMBER_DETAILS")
    december_details_df.createOrReplaceTempView("DECEMBER_DETAILS")

    january_travels = spark.sql("SELECT 'JANUARY' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM JANUARY_DETAILS")
    february_travels = spark.sql("SELECT 'FEBRUARY' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM FEBRUARY_DETAILS")
    march_travels = spark.sql("SELECT 'MARCH' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM MARCH_DETAILS")
    april_travels = spark.sql("SELECT 'APRIL' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM APRIL_DETAILS")
    may_travels = spark.sql("SELECT 'MAY' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM MAY_DETAILS")
    june_travels = spark.sql("SELECT 'JUNE' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM JUNE_DETAILS")
    july_travels = spark.sql("SELECT 'JULY' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM JULY_DETAILS")
    august_travels = spark.sql("SELECT 'AUGUST' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM AUGUST_DETAILS")
    september_travels = spark.sql("SELECT 'SEPTEMBER' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM SEPTEMBER_DETAILS")
    october_travels = spark.sql("SELECT 'OCTOBER' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM OCTOBER_DETAILS")
    november_travels = spark.sql("SELECT 'NOVEMBER' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM NOVEMBER_DETAILS")
    december_travels = spark.sql("SELECT 'DECEMBER' AS MONTH, COUNT(DISTINCT(Usuario_Id)) AS TOTAL_USERS FROM DECEMBER_DETAILS")

    total_travels_details = january_travels.union(february_travels).union(march_travels).union(april_travels).union(may_travels).union(june_travels).union(july_travels).union(august_travels).uni

    sorted_df = total_travels_details.sort("TOTAL_USERS")
    final_df = sorted_df.withColumn("ID", row_number().over(Window.orderBy("TOTAL_USERS")))
    save_df(final_df, "/Users/miguelojeda/Google Drive/Maestría/TOG/Desarrollo del proyecto/Spark/data_to_be_analyzed")


if __name__ == '__main__':
    main()
```

Figura 28. Parte de script utilizado para llevar a cabo la integración de datos

Tal y como se puede observar en la imagen superior, los datos fueron generados, y por tanto almacenados igualmente, de una manera parcial (por mes), sin embargo, al trabajar con Machine Learning, la mayoría de las veces se obtienen mejores resultados cuando los algoritmos son aplicados a todos los datos "en general" – es importante tener en cuenta que el término "en general" hace referencia al conjunto de datos en sí y no solo a partes de este. Por lo que muchas veces es necesario llevar a cabo una segmentación de los datos en donde una parte es utilizada para generar los modelos mientras otra parte es utilizada para realizar pruebas de estos – y no a subconjuntos de los datos, dado que en muchas ocasiones no se pueden considerar características importantes las cuales se pudieran encontrar en algún sub-conjunto ignorado.

Por tal motivo es que se llevó a cabo esta integración de la información, posteriormente fue posible llevar a cabo la implementación del ML en el proyecto. Para este punto, se hizo implementación principalmente de la Regresión Lineal (LR por sus siglas en inglés).

El análisis de regresión lineal es una técnica utilizada para estudiar la relación entre variables. Es posible adaptarla a una amplia variedad de situaciones. En los trabajos de investigación social como este, el análisis de regresión se utiliza para predecir un amplio rango de fenómenos, desde medidas económicas hasta diferentes aspectos del comportamiento de los programas implementados en ambas ciudades. En el contexto de la investigación puede utilizarse para determinar en cuál de diferentes medios de comunicación puede resultar más eficaz llevar a cabo modificaciones; para mejorar el uso de los servicios con base en las recomendaciones de los usuarios.

Tanto en el caso de dos variables (regresión simple) como en el de más de dos variables (regresión múltiple), el análisis de regresión lineal puede utilizarse para explorar y cuantificar la relación entre una variable llamada dependiente o criterio (Y) y una o más variables llamadas independientes o predictoras (tal y como se puede ver en la imagen inferior), así como para desarrollar una ecuación lineal con fines predictivos. Además, dicho análisis sobre la información almacenada puede llevar asociada una serie de procedimientos de diagnóstico (futuro uso de los servicios, usuarios activos en el siguiente año) que logran informar sobre la estabilidad del análisis y que proporcionan pistas sobre como perfeccionarlos.

El objetivo del presente análisis es el de proporcionar los fundamentos de los procesos generados para identificar posibles características de los servicios de movilidad. Al igual que en los capítulos anteriores, no se hace hincapié en los aspectos más técnicos de la implementación, sino que se intenta fomentar la comprensión del por qué y cómo se hizo uso de dichos análisis por medio de la regresión lineal.

Para ambos servicios se generó el estudio más formal de la regresión con el modelo de regresión lineal simple, pero conviene no perder de vista que, puesto que generalmente se está interesado en estudiar simultáneamente más de una variable predictora, este análisis es solo un punto de partida en la explicación del análisis de regresión.

Dicho esto, se llevó a cabo la identificación de dicha variable en los distintos servicios analizados (Uber y programas de Bici Pública), tal y como se puede observar en la imagen inferior (para el caso de los usuarios del programa MiBici en la ZMG) la variable independiente utilizada fue el mes mientras que la variable dependiente fue el número total de usuarios que hicieron uso del servicio en ese mismo mes.

```
def predict_values(processed_df):

    # Another option to get DF column values.
    month_id_array = processed_df.select('MONTH_ID').rdd.flatMap(lambda x: x).collect()
    users_array = processed_df.select('TOTAL_USERS').rdd.flatMap(lambda x: x).collect()

    independet_variable = np.array(month_id_array).reshape((-1,1))
    dependet_variable = np.array(users_array)

    model = LinearRegression()
    model.fit(independet_variable, dependet_variable)
    model = LinearRegression().fit(independet_variable, dependet_variable)
    r_sq = model.score(independet_variable, dependet_variable)
    # print("coefficient of determination: ")
    # print(r_sq)

    months_to_be_predicted= []
    for i in range(13, 25):
        months_to_be_predicted.append(i)

    month_id = np.array(months_to_be_predicted)
    predictions = model.predict(np.array(months_to_be_predicted).reshape((-1,1)))
    plt.bar(month_id, predictions)
    plt.xlabel("MONTH ID")
    plt.ylabel("NUMBER OF USERS")
    plt.show()

    return predictions


def main():

    spark = get_spark_session()

    # DATASET PRE-PROCESSING
    original_df = read_csv(spark, "/Users/miguelojeda/Google Drive/Maestría/TOG/Desarrollo del proyecto/Spark/data_to_be_analyzed/users_per_month.csv")
    processed_df = original_df.drop("MONTH").withColumnRenamed("ID", "MONTH_ID")

    #-------------------------------------------------------------------------------------------
    # PERFORM LINEAR REGRESSION
    month_id_list = processed_df.toPandas()['MONTH_ID'].values.tolist()
    total_users_list = processed_df.toPandas()['TOTAL_USERS'].values.tolist()
    # linear_regression_with_numpy(month_id_list, total_users_list)

    # -------------------------------------------------------------------------------------------
    # GENERATE LINEAR REGRESSION MODEL TO BE USED FOR PREDICTION
    linear_regression_with_mllib(processed_df)

    # -------------------------------------------------------------------------------------------
    #PERFORM PREDICTION
    values_predicted = predict_values(processed_df)
    print(type(values_predicted))
```

Figura 29. Extracto de script generado para aplicar regresión lineal a datos de MiBici

Dentro de MlLib se tiene la función *MLlibLinearRegression()* la cual necesita como parámetro de entrada las "features" que mejor describen a la variable independiente. Tal y como ya se mencionó, únicamente se hizo uso de regresión simple, por lo que fue solamente una variable la utilizada. Sin embargo, evidentemente como en la gran mayoría de algoritmos, es necesario seleccionar un subconjunto de todos los datos para posteriormente usarlo como prueba del algoritmo a fin de revisar su comportamiento. Por tal motivo es que se hizo uso de la función *randomSplit([0.7,0.3])* para generar dicho sub conjunto tomando de una manera aleatoria el 30% de todos los datos. El resto es el utilizado por la función de regresión lineal a fin de llevar a cabo el procedimiento.

Algo importante que se tuvo presente fue que, si la dependencia de las variables analizadas no es lineal, no significa que no tengan otro tipo de dependencia o relación, por tanto claro queda que no se correlacionan a través de una recta pero sí lo pueden hacer mediante otra función matemática.

Como tenemos dos variables, tendremos dos ecuaciones de rectas diferentes en función de la variable dependiente e independiente, a las que llamaremos de x sobre y o de y sobre x. La expresión matemática de dichas ecuaciones es así:

$$(y - \bar{y} = \frac{S_{xy}}{S_x^2}(x - \bar{x})$$

Donde $\bar{y}$ y $\bar{x}$ son los promedios o medias de las variables analizadas respectivamente, S_{xy} es la covarianza y S_x^2 es la varianza de la variable *x*. En esta ecuación *y* es la variable dependiente y *x* la independiente. La otra ecuación de la recta será:

$$(x - \bar{x} = \frac{S_{xy}}{S_y^2}(y - \bar{y})$$

Como estas rectas son las que tenemos que obtener, es necesario calcular las variables marginales contenidas en ellas, las varianzas, las medias y la covarianza. Afortunadamente, todos estos cálculos son realizados por la función empleada de MlLib por lo que no fue necesario realizarlos de manera manual.

El objetivo de este análisis de regresión es pronosticar la demanda a partir de una (o posiblemente más causas), la cual puede es el tiempo. El análisis de regresión es pertinente cuando se evidencia una tendencia en los datos históricos del pronóstico. Este coeficiente, permitió entender qué tanta correlación existe entre la demanda y el tiempo. Teniendo a consideración los siguientes tipos de correlaciones:

- **Correlación perfecta**: Cuando el resultado de coeficiente es igual a 1 o -1. En este caso existe una relación directamente proporcional entre la demanda y el tiempo.
- **Correlación fuerte**: Cuando el resultado es mayor a 0.5 y menor que 1 (correlación positiva) o menor a -0.5 y mayor que -1(correlación negativa).
- **Correlación débil**: Valores que están entre -0.5 y 0.5.

Entre más cercano se encuentre el coeficiente de correlación a +1 o -1 más fuerte será la tendencia. Por ejemplo, si la correlación es igual a 1, es posible observar que la relación entre las variables es directamente proporcional, en el sentido que, si uno aumenta, la otra también lo hará.

Para poder elegir el coeficiente de correlación adecuado, se tuvo que analizar el tipo de variables y la distribución que presentan. En este caso, ambas variables son cuantitativas continuas y pueden transformarse en rangos para ordenarlas, por lo que a priori los tres coeficientes podrían aplicarse. La elección se hará en función de la distribución que presenten las observaciones.

Bien vale aclarar que este método es más útil cuando se enfoca en periodos de largo plazo. Esto aunado a su utilidad para estimar la demanda en función de variables independientes. Dado que, en este caso, el presente trabajo es un indicativo inicial, se considera que, en el futuro uso del mismo, se podrá contar información suficiente para llevar a cabo una validación completa.

5. RESULTADOS Y DISCUSIÓN

Resumen: *En este capítulo se presentan los resultados obtenidos del desarrollo de este trabajo y una discusión sobre la implementación del data lake para el análisis de movilidad dentro de la ZMG y la CDMX.*

5.1. Resultados

Una vez culminadas todas las fases del proyecto, sin duda alguna que, si algo queda claro, es que llevar a cabo la implementación de un data lake, no es una tarea sencilla. El hecho de que sea una tecnología nueva dentro del ámbito tecnológico ocasiona que la información existente muchas veces no sea necesaria o no esté completa para poder tener un entendimiento totalmente claro. Sin embargo, este conocimiento ya se tenía a priori, pues fue una de las motivantes que generó la planeación y ejecución del presente trabajo.

Con base en los resultados obtenidos, dichos resultados del proyecto se pueden dividir en dos etapas lógicas abstractas. La primera de ellas corresponde a la que engloba todo el trabajo realizado en AWS. Tal y como se mencionó al principio del proyecto, otro aspecto importante que ayudó a la planeación del proyecto es el correspondiente al tema económico. Es bien sabido que el tema de tecnología no es un aspecto del que muchas empresas y organizaciones estén dispuestos/as a hacer uso de, ya que conlleva una inversión que en ocasiones supera los presupuestos asignados y/o planeados. El hacer uso de los servicios en la nube ofrecidos por cualquier proveedor sin duda alguna que ha venido a ser una revolución para este ámbito, pues ofrece una solución fuerte, segura y confiable sin tener que realizar una inversión grande de dinero.

Otro aspecto importante de hacer uso de estos servicios es el referente a la Infraestructura el cual es un modelo de aprovisionamiento, en el cual una organización coloca 'fuera de ella' el equipo usado para soportar operaciones, esto incluye el almacenamiento de la información, el hardware, servidores y componentes de redes. El proveedor del servicio. En ocasiones la IaaS es referida también como *Hardware as a Service* o HaaS.

La ventaja más evidente, es la de transferir hacia el proveedor problemas relacionados con la administración de equipos de cómputo. Además, las Infraestructuras como Servicio permiten escalabilidad prácticamente automática y transparente para el consumidor, dejando la responsabilidad a los proveedores de los servicios.

De esta manera es que todo el desarrollo realizado en el presente trabajo no implicará ninguna problemática para un caso de uso futuro por parte de alguna institución que desee llevar a cabo investigaciones y análisis de la información, pues solo será necesario agregar los datos y realizar pequeñas configuraciones que no son para nada complejas y pueden ser realizadas por casi cualquier persona.

AWS ofreció una solución de data lake para configurar los servicios fundamentales de AWS necesarios para etiquetar, buscar, compartir, transformar, analizar y administrar fácilmente todos los subconjuntos específicos de datos con los cuales se trabajó. La propuesta implementa una solución a la cual los usuarios pueden acceder para buscar y encontrar conjuntos de datos disponibles para cualquier necesidad.

En la siguiente imagen se muestra la arquitectura que se planteó e implementó usando los servicios ofrecidos por AWS.

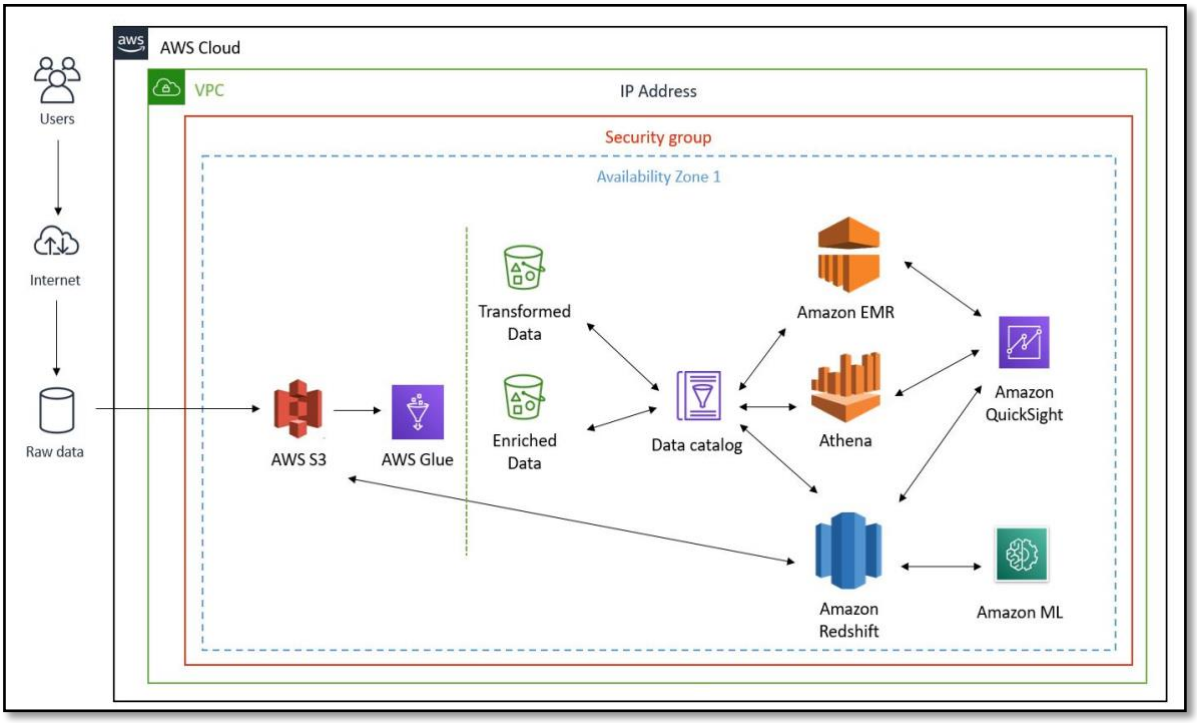


Figura 30. Arquitectura utilizada en AWS

La solución generada aprovecha la seguridad, la durabilidad y la escalabilidad de Amazon S3 para administrar un catálogo constante de conjuntos de datos de la organización, *Data Catalog* para administrar los metadatos correspondientes. Una vez catalogado un conjunto de datos, sus atributos y etiquetas descriptivas están disponibles para la búsqueda. Los usuarios pueden buscar y navegar por los conjuntos de datos disponibles, y crear una lista de datos a los que necesitan acceder para posteriormente generar análisis sin tener que preocuparse por nada más.

De igual manera, la solución hace un seguimiento de los conjuntos de datos que selecciona un usuario y genera un archivo de manifiesto con enlaces de acceso seguro al contenido deseado cuando el usuario finaliza la sesión.

Todo el desarrollo y la integración de los diferentes servicios se explicó en el capítulo anterior, por lo que en este solo se hablará sobre los resultados finales para el usuario obtenidos en AWS.

Dicho esto, se hace referencia principalmente a dos servicios; Amazon Athena y QuickSight. El primero de ellos, tal y como se habló previamente, es un servicio de consultas interactivo que facilita el análisis de datos con SQL estándar.

Una vez generado todo el catálogo de metadatos, se obtuvo la ventaja de poder utilizar este servicio, pues la lectura de los datos (almacenados físicamente en S3) fue mucho más sencilla y rápida, pues la integración de estos, le confirieron estas características. Es por eso por lo que en los casos donde se requirió hacer una consulta de información de los sub-conjuntos (revisar datos de los servicios por mes), Athena presentó una opción bastante fiable y segura para utilizar. Tal y como se puede ver en la Figura 31, el servicio ofreció una interfaz bastante amigable donde cualquier persona autorizada puede acceder a realizar las consultas de información utilizando SQL estándar y nada más.

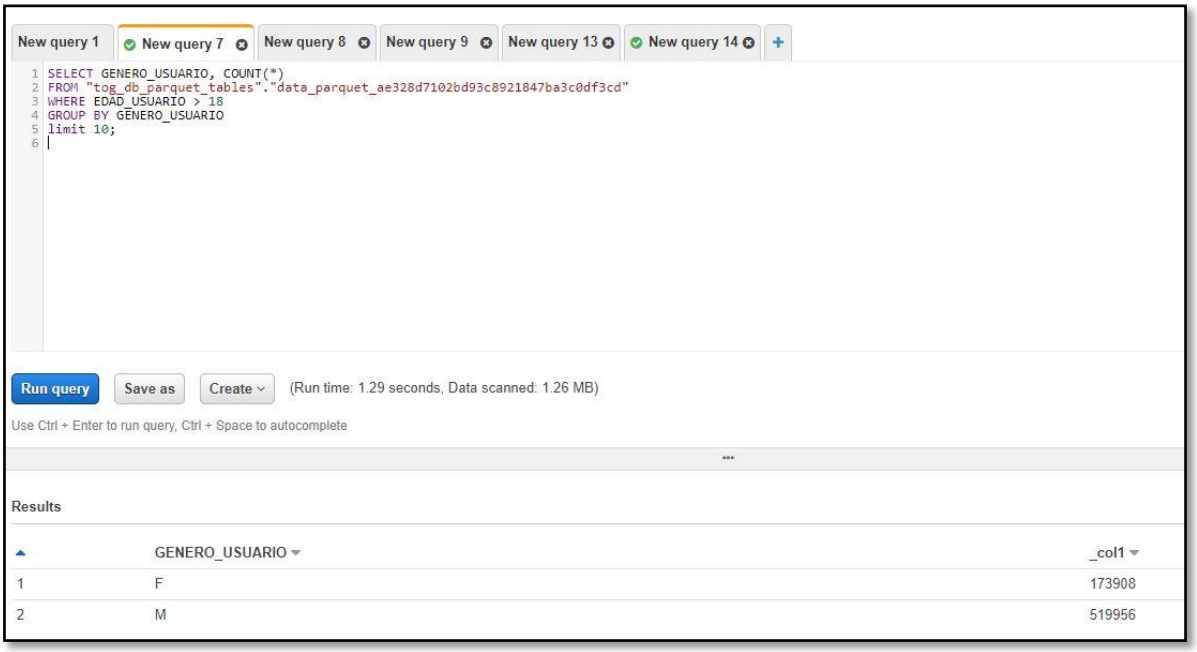


Figura 31. Ejemplo de query ejecutada en Amazon Athena

Habiendo utilizado esta aplicación se pudo observar como es un servicio sin servidor (*serverless*). Se pudo realizar consultas en los datos con rapidez sin tener que configurar ni administrar servidores ni almacenes de datos. Simplemente fue necesario apuntar al catálogo de datos, definir los esquemas y comenzar a realizar consultas con el editor de consultas integrado. Así mismo, permitió acceder a toda la información sin necesidad de configurar procesos complejos, además de que no fue necesario preocuparse de disponer de suficientes recursos informáticos para obtener un desempeño de consultas rápido e interactivo, ya que fue posible ejecutar consultas de manera simultánea automáticamente sin que se afectara el rendimiento, en donde la mayoría de los resultados se obtuvieron en cuestión de segundos.

En lo que respecta a Amazon QuickSight se obtuvo la, tan necesaria, aplicación gráfica en donde fue posible acceder a la información y desplegarla gráficamente mediante el uso de paneles y gráficas. Al igual que con Athena, este servicio presentó una arquitectura sin servidor, por lo que no fue necesario llevar a cabo administración de infraestructura, planificación de capacidad o scripting.

La mejora de las aplicaciones con los paneles incrustados (tal y como se puede observar en la Figura 32), sin duda alguna aceleró el tiempo de familiarización con los datos de una manera sencilla ya que se integró con los orígenes de datos locales incluida la integración nativa con los servicios RedShift, Data Catalog e IAM proporcionándole todo lo necesario para compilar una solución integral de BI.

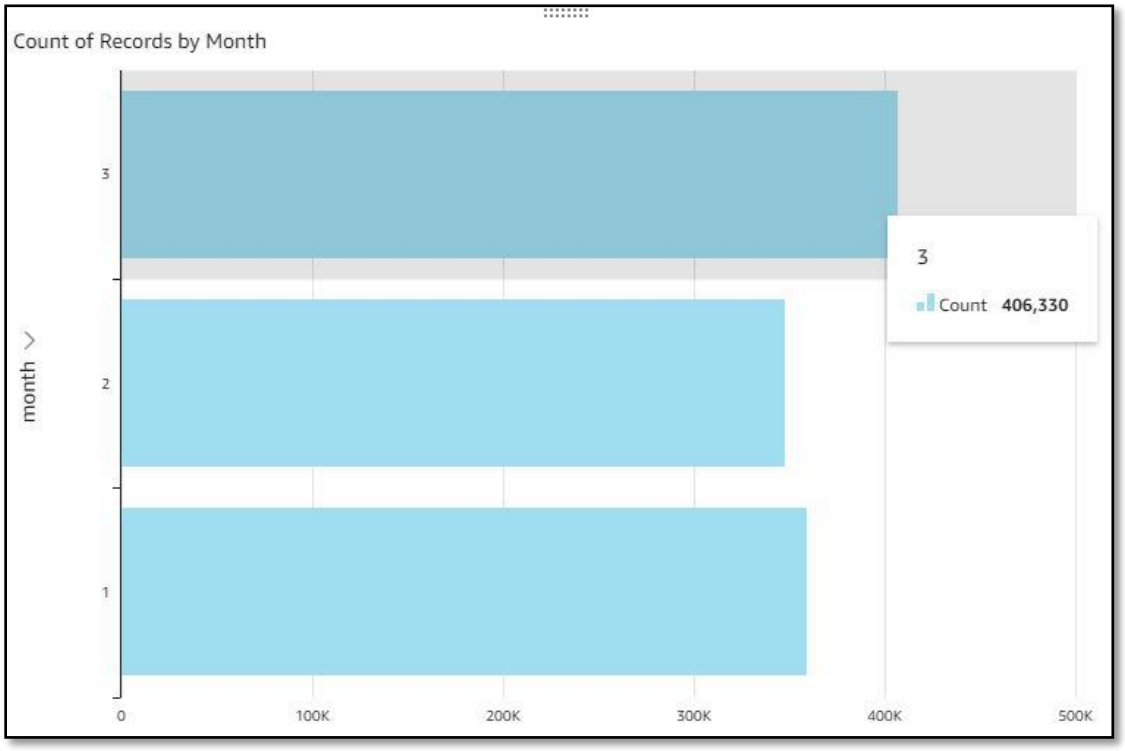


Figura 32. Número de usuarios de UBER en el primer trimestre del 2019 en la CDMX

El hecho de que las empresas y los organismos gubernamentales suelen depender de la escritura de consultas ad hoc SQL complejas o del envío manual de hojas de cálculo estáticas para compartir los datos y la información. Los resultados obtenidos por este servicio permitieron proporcionar información a todas las personas interesadas. Fue posible compartir paneles integrales e interactivos, que permitieron desglosar y explorar los datos para responder a preguntas hechas comúnmente por los usuarios y medios de comunicación además de que se obtuvo información gráfica la cuál muchas veces es utilizada en trabajos de investigación y reportajes.

En lo que respecta a la segunda etapa lógica de resultados, en esta se hablará sobre los que se obtuvieron por el uso de tecnologías "Big Data" (principalmente Spark). En esta etapa se deja de lado las tecnologías (AWS) donde fueron ejecutados todos los procesos realizados.

El primer enfoque por tratar es el referente al procesamiento ejecutado sobre el conjunto de datos en general. Tal y como se explicó en el capítulo de "*Análisis de la información con Machine Learning*", todos los procesos se ejecutaron sobre un solo conjunto de datos, es decir, se llevó a cabo una unificación de todos los subconjuntos procesados a fin de obtener uno solo con toda la información por servicio de movilidad de utilizado. El realizar esto, conllevó generar un procesamiento paralelo, pues fueron millones de registros utilizados. A fin de tener una mejor comparativa, se realizaron pruebas utilizando el mismo dataset, únicamente cambiando el motor utilizado para llevar a cabo todos los procesamientos programados, dichos resultados se pueden observar en la tabla número 5.

Motor de procesamiento	Características	Tiempo total requerido
Sistema centralizado (Computadora)	4 cores 16 GB de RAM 128 GB de SSD	32 minutos con 16 segundos
Sistema distribuido (Clúster de 3 instancias corriendo Spark)	4 cores 16 GB de RAM 64 GB de SSD (por instancia)	9 minutos con 23 segundos
Sistema distribuido (Clúster de 5 instancias corriendo Spark)	4 cores 16 GB de RAM 64 GB de SSD (por instancia)	4 minutos con 54 segundos

Tabla 5. Comparativa de motores de procesamiento utilizados

Tal y como se puede observar en la tabla comparativa, el tiempo que fue requerido por el sistema centralizado fue 3.48 veces más tardado que el sistema distribuido corriendo Spark en 3 instancias, y 7.08 veces más que el clúster con 5 instancias. De esta manera, fue posible reducir poco más de 27 minutos el tiempo de procesamiento requerido para poder llevar a cabo un análisis de ML. Es por esto por lo que se pudo comprobar como en el sistema distribuido se tiene un procesamiento mucho más rápido que un sistema centralizado, además de que tiene una mayor confiabilidad pues, cuando se realizaron las pruebas con el sistema centralizado, este presentó un calentamiento en el hardware y un deterioro en las demás funcionalidades a diferencia del clúster utilizado.

Al estar distribuida la carga de trabajo en muchas máquinas la falla de una de ellas no afecta a las demás, el sistema sobrevive como un todo. Además, en caso de que la carga de procesamiento llegue a aumentar se puede añadir procesadores al sistema incrementando su potencia en forma gradual según las necesidades.

Dejando de lado el tema referente al proceso de los sistemas utilizados para trabajar con la información y abordando los resultados obtenidos por el procesamiento de dicha información, es que se muestran a continuación.

Dado que el aumento en la problemática referente a la movilidad será un asunto que con el paso de tiempo tendrá un mayor impacto en la sociedad, es que se decidió realizar el análisis de regresión (previamente explicado) para poder tener un panorama hacía el futuro de la demanda que los servicios tratados en este proyecto tendrán en los próximos años. Dicho proceso permitió modelar una relación entre los conjuntos de variables. El resultado es una ecuación para definir la recta (véase en la Figura 33 la recta generada para el caso del programa MiBici) que se puede utilizar para hacer proyecciones o estimaciones sobre los datos.

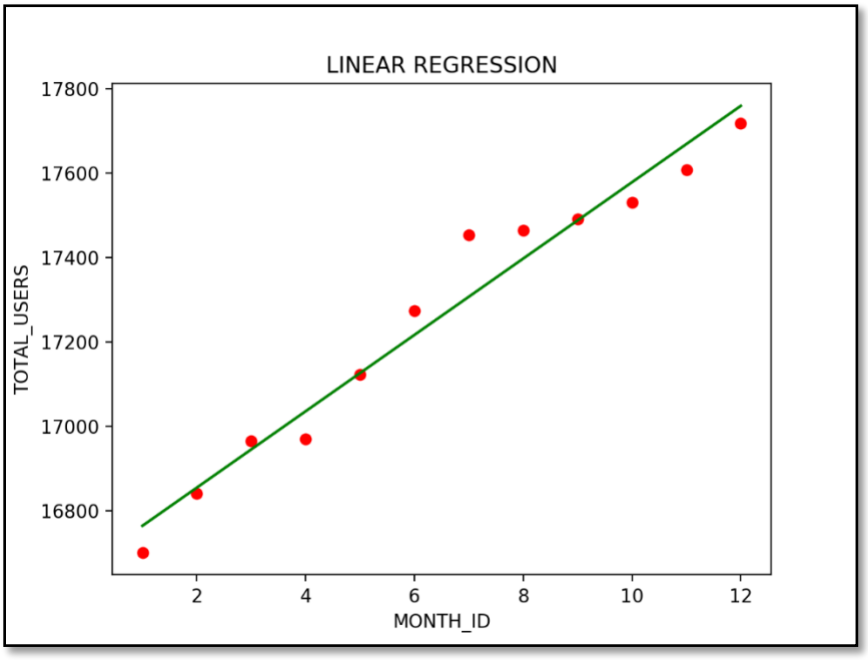


Figura 33. Regresión lineal aplicada para el número de usuarios del programa MiBici en el 2019

Dicho proceso permitió determinar con confianza cuáles son los factores más importantes, cuáles se pueden ignorar y cómo influyen entre sí para poder obtener una predicción sobre el uso de los servicios.

El modelo utilizado pudo ser validado con el dataset de prueba. Para dicha comparativa es que se utilizaron los valores reales y los valores calculados con la regresión (véase un subset de estos en la Figura 34).

features	label	prediction
[3.0]	5263006	4855284.8670212785
[4.0]	5121016	4908790.297872342
[7.0]	5144846	5069306.590425531
[12.0]	5486685	5336833.744680848

Figura 34. Comparativa de valores reales vs valores calculados

Una vez que el modelo fue generado y validado, finalmente se procedió a aplicar dicho modelo en los datos actuales a fin de poder generar una predicción de usuarios para el año 2020. Para este caso se consideró únicamente los meses del presente año como valores a predecir para tener una estimación de los usuarios que cada servicio tendrá. Dichos resultados se pueden observar en las Figuras 35 y 36 (considérese los valores de *MONTH_ID* como: **<u>1-12</u> → Meses del 2019**, **<u>13-24</u> → Meses del 2020**).

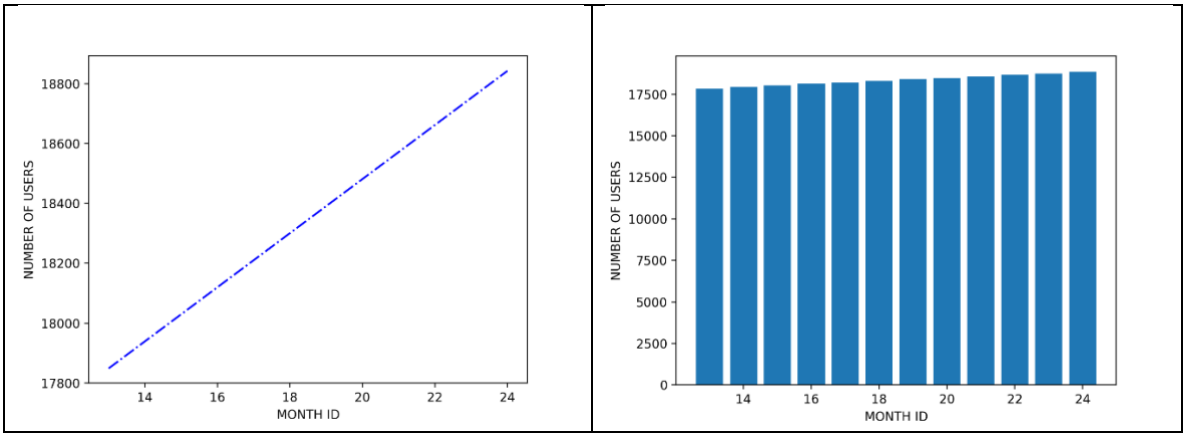


Figura 35. Estimación de usuarios activos del programa MiBici en la ZMG

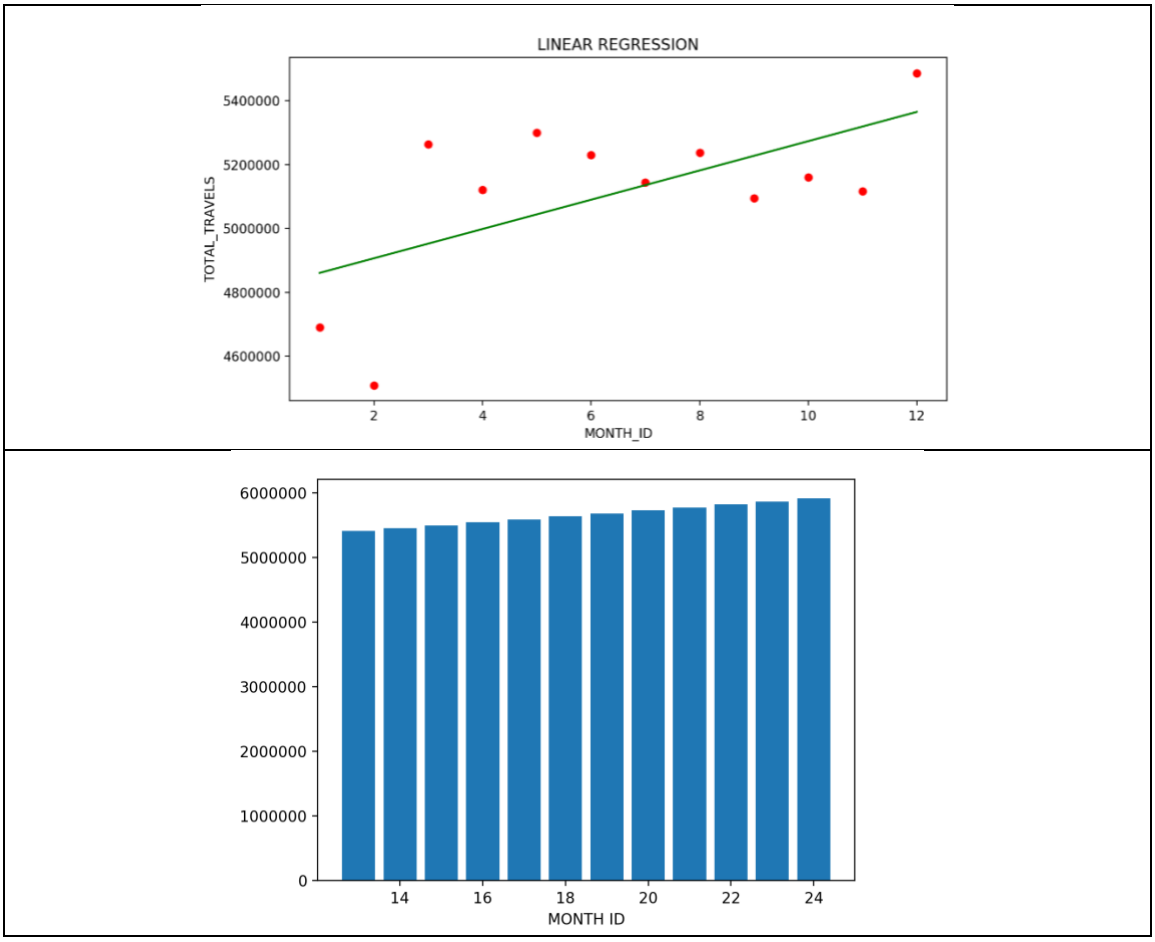


Figura 36. Regresión lineal y predicción de usuarios de Uber en la CDMX y la ZMG

Los datos y el análisis de estos son cada vez más importantes en la actualidad para detección de oportunidades y, mejor aún, para que los proveedores de servicios que utilizan tecnología para satisfacer necesidades del cliente puedan identificar las posibles áreas de oportunidad en las que se puede trabajar. Aún y que se tenía este conocimiento a priori, no fue suficiente, pues fue imprescindible sacarle partido a cada dato almacenado teniendo como sustento una estrategia y así contar con insights, patrones de comportamiento y sobre todo encontrar solución a distintas problemáticas que presentan y podrían presentar dichos servicios en un futuro.

Englobando en términos generales los resultados obtenidos en este trabajo, se pueden presentar de la siguiente manera:

1. **Indización de datos.** La implementación del data lake permitió almacenar datos relacionales (una colección de elementos de datos organizados como un conjunto de tablas descritas formalmente desde las cuales se puede acceder de muchas maneras diferentes sin tener que reorganizar las tablas de la base de datos). Bases de datos (datos con almacenamiento histórico), y, en un futuro no muy lejano, datos no relacionales como dispositivos conectados al servicio utilizado y redes sociales. También se brindó la capacidad de comprender qué datos se encuentran en el lago a través del rastreo, la catalogación y la indexación de datos.
2. **Analytics.** Se permitió el desarrollo y análisis de operaciones dentro de los servicios utilizados de la nube de AWS, los cuales accedieron a los datos con previa elección de marcos analíticos y herramientas. Esto también incluyó marcos de datos de fuente abierta como Apache Hadoop y Apache Spark. El data lake permitió ejecutar *Analytics* sin la necesidad de mover datos de un sistema a otro.
3. **Aprendizaje automático.** Tal y como se observó en el último capítulo, se logró generar diferentes tipos de información operativa. Donde se incluyó información sobre datos históricos y aprendizaje automático en los que los modelos generados fueron capaces de producir pronósticos y predicciones.
4. **Mejor interacción con el usuario.** Al final, todo el trabajo tiene un solo fin, y es el de poder generar mejoras que permitan ofrecer a los usuarios una movilidad mucho más efectiva dentro de las dos ciudades más congestionadas del país. Es por eso que dicha implementación logra tener una integración sencilla con nueva información, en donde se puede combinar los datos de los usuarios con el análisis de datos de diferentes servicios, aún y que sean de diferentes tipos y de distintas zonas geográficas (así como una plataforma que incluye el historial registrado por los proveedores de los servicios de movilidad analizados) a fin de generar un procesamiento, una integración y un análisis que conlleven a mejoras de los mismos.

5.2. Discusión

Con base en toda la bibliografía consultada y analizada durante la etapa de definición del estado del arte del presente trabajo, fue fácilmente identificable como hasta el día de hoy no existe una organización y/o plataforma que sea capaz de llevar a cabo una integración y análisis de todos los datos e información relacionada a la movilidad de las dos (o más) ciudades más importantes del país.

Si bien es cierto que hay instituciones – como lo es el caso del INEGI, el cual es un organismo público con autonomía técnica y de gestión, personalidad jurídica y patrimonio propios [30] el cuál se encarga de producir, integrar y dar a conocer la información estadística (de la población y la economía) y geográfica (donde se abarca todos los aspectos que caracterizan el territorio de México). – que se encargan de llevar a cabo tareas relacionadas a la recopilación y análisis de información sobre distintos temas y ámbitos sociales que engloban el día a día del país, la gran mayoría basan todos sus análisis únicamente en la información recabada por el personal afiliado a estas instituciones.

En un mundo globalizado donde la tecnología es un aspecto que evoluciona a pasos agigantados, así como la posibilidad de tener el acceso a los millones de datos generados por dicha tecnología. Hoy en día resulta sumamente sencillo identificar que los tiempos del análisis de información han cambiado de la misma manera, pues hoy ya no basta con realizar una encuesta anual en la cual se puedan basar todos los análisis generados por una institución sobre algún aspecto social del país – referente en este trabajo al tema relacionado a la movilidad – de una ciudad tan compleja, pues estos serían considerados sumamente "ineficientes" (por decirlo de alguna manera). Dichos análisis deben de formarse con los datos generados en el día a día para de esta manera generar información que sea mucho más exacta y verídica, que permita poder llevar a cabo una mejor toma de decisiones cuando se requiera hacer modificaciones a cualquier medio de transporte que sea utilizado dentro de la sociedad analizada.

El presente trabajo sin duda alguna que presenta una solución sólida con base en los resultados obtenidos, que permite atacar este problema de una manera confiable y sencilla de implementar, pues una de las principales características de un data lake, es que es una herramienta que almacena todos los tipos de datos, tanto si están estructurados, desestructurados o semi-estructurados. Estos, como fuimos capaces de observar, se acumulan de forma original, plana o en bruto, sin ningún tipo de procesamiento (*raw data*). La información almacenada procede de una gran variedad de orígenes, por lo que se tiene la oportunidad de almacenar datos de todo tipo: bases de datos, aplicaciones móviles, documentos ofimáticos, registros de servidores, recursos extraídos de Internet, redes sociales, textos, imágenes etc. con el objetivo de ser estudiados y analizados posteriormente como un solo conjunto en general.

Aún y que, en este caso, se tomaron en cuenta los datos de 3 instituciones (UBER, MiBici y EcoBici) para la formulación, estructuración y ejecución del proyecto, no se tiene ninguna limitante en cuanto a la posibilidad de llevar a cabo una integración con otros servicios, instituciones y/o organizaciones.

6. CONCLUSIONES

Resumen: *En este capítulo se presentan las conclusiones y trabajo futuro en relación con la implementación de un data lake para conducir un análisis de movilidad en las principales ciudades del país.*

6.1. Conclusiones

La idea de tener una infraestructura capaz de poder llevar a cabo una integración de datos sin importar la fuente, el formato y el origen de estos, a fin de conducir un análisis de movilidad en el país, hubiera resultado en una idea muy difícil de concebir hace unos años. El poco uso y conocimiento de la tecnología, así como el proceso de llevar a cabo una recolección de información con un intervalo mucho menor al hasta ahora realizado (1 vez por año), resultaba en un trabajo que involucraría una gran demanda de tiempo de planeación y desarrollo, así como un presupuesto mucho mayor al designado.

Los resultados presentados en el presente proyecto sin duda alguna ofrecen una opinión distinta a la ya mencionada, pues con el avance de la tecnología en conjunto con el uso de las herramientas *open source* creadas para el "Big Data" y el uso de la nube, se tiene la posibilidad de diseñar y generar infraestructuras capaces de llevar a cabo todo un proceso completo de procesamiento, integración y análisis de información. Todo esto, en conjunto con la posibilidad de hacer uso de los datos (cada vez más frecuentes) generados, y ofrecidos al público en general, por los proveedores de servicios de movilidad.

Si bien es cierto que el haber logrado la integración de todas las herramientas utilizadas requirió mucho más trabajo del esperado, también lo es que una vez que se logró generar dicha integración, los resultados fueron los esperados.

En lo que refiere a la fase de carga de datos, el "pipeline" generado fue capaz de llevar a cabo una lectura y escritura de estos sin presentar problemas por el formato original, pues el medio de almacenamiento utilizado (S3) no presentó dificultades ni problemas al momento de trabajar con estos.

Respecto al tema de la integración, fue en esta etapa donde se tuvo un poco más de problemas. Si bien es cierto que todos los servicios utilizados fueron de AWS, muchos de estos presentaron diferencias en cuanto a temas de configuración y protocolos de comunicación entre ellos, por lo que fue necesario llevar a cabo una investigación profunda de los servicios que no solo abarcó el conocer su funcionamiento de manera individual, sino que también un entendimiento de los procesos diseñados para la comunicación entre ellos. Una vez completado este proceso y lograda una integración total, no se tuvo ningún problema referente a su funcionamiento y/o comunicación.

Finalmente, hablando sobre la etapa del análisis, es aquí donde se pudo demostrar de una manera mucho más visual y didáctica los resultados esperados del proyecto, pues la generación de los reportes, la obtención de información específica por medio de consultas con SQL, la visualización de *dashboards*, así como los resultados del análisis de regresión en Spark, se realizaron de la misma manera, aún y que la información en su forma original no planteaba esa posibilidad, pues los tipos de datos y su estructuración variaba de un dataset a otro. El haber tenido la posibilidad de generar todos estos tipos de información siguiendo los mismos procedimientos, presentó en su forma más clara, el cumplimiento del objetivo principal del proyecto, el cuál es el de tener la posibilidad de utilizar una infraestructura integrada para llevar a cabo un análisis de información continua de movilidad en la ZMG y la CDMX.

Un aspecto importante a tener en cuenta es el tema referente a los resultados obtenidos por la regresión lineal aplicada a los 3 servicios con los que se trabajó en este proyecto, pues los valores predichos sobre el uso de estos servicios para el presente año (2020) no coincidieron con los valores reales registrados, en los meses que hasta el momento se tiene registro (a excepción de enero y febrero), debido a la problemática global que fue la aparición del COVID-19. Debido a la gravedad, y facilidad de contagio de este virus dentro de las sociedades, la reducción de la movilidad en las ciudades analizadas se vio reducida en gran medida (alrededor de un 40% para la ZMG [31] y un 27% para la CDMX [32]) por lo que la comparación de dichos valores no presenta resultados verídicos pues las condiciones fueros sumamente diferentes entre un año y otro. Sin embargo, a su vez se tiene una futura posibilidad que permita a los analistas y organismos conducir análisis tomando situaciones externas (como la vivida hasta ahora) para generar modelos mucho más complejos que sean capaces de integrar este tipo de variables poco comunes.

Finalmente, otro de los aspectos considerados más importantes, es el que corresponde a la capacidad de poder añadir nuevos datos y fuentes de información al data lake, a fin de que estos sean integrados con los hasta ahora utilizados para así poder ir mejorando el proceso de llevar a cabo un análisis de movilidad en las ciudades donde sea necesario hacerlo. Por tal motivo es que la infraestructura creada admite fácilmente esta posibilidad sin que sea necesario cambiar o generar una nueva configuración, únicamente es necesario cargar los datos en S3 y actualizar el catálogo de tablas con AWS Glue, para que de esta manera se tenga un mapeo de la nueva información para que finalmente esta pueda ser utilizada en cualquiera de las herramientas utilizadas de análisis.

Teniendo en cuenta todo lo presentado en este proyecto, y analizando las nuevas implementaciones utilizadas dentro de la industria, se puede fácilmente concluir como es cada vez más común ver que se esté experimentando con data lakes, a fin de capturar ventajas inherentes en los flujos de información que son fácilmente accesibles independientemente de la plataforma y el caso de uso, ya que sin duda alguna ofrecen una mejor opción para almacenar los datos que solo utilizando los almacenes tradicionales. Sin embargo, al igual que con cualquier implementación de una nueva tecnología, es necesario volver a diseñar algunos procesos y modelos hasta ahora utilizados, y mayormente basados, en únicamente datos estructurados que, sin duda alguna, merece la pena tener a consideración pues la velocidad y capacidad de procesamiento de un data lake no tiene comparación alguna con los sistemas tradicionales hasta ahora utilizados.

6.2. Trabajo Futuro

El día 11 de septiembre del año pasado (2019), Uber puso a disposición global los datos agregados y anónimos de todos viajes realizados con la aplicación de Uber en algunas ciudades del país, mismos que fueron utilizados en el desarrollo de este trabajo. Conforme pase el tiempo, es muy seguro que este tipo de información será cada vez más común y que los organismos proveedores de servicios de movilidad pongan a disposición del público en general datos generados por los usuarios de dichos servicios.

Si bien es cierto que el presente trabajo aborda una implementación con diferentes tipos de datos y formatos, también lo es que el data lake diseñado tiene potencial suficiente para trabajar con una mucha mayor cantidad de información, así como con más tipos de datos. De igual manera, sin duda alguna que los medios de transporte analizados son solo algunos de todos los que se utilizan como servicios de movilidad en la sociedad mexicana.

Es por esto por lo que el presente trabajo pretende forjar los cimientos de una nueva tecnología de análisis de movilidad que pueda ser utilizada y mejorada desde dos principales enfoques:

1. Añadiendo datos (una vez que estén disponibles) sobre otros medios de transporte utilizados por la ciudadanía, como lo sería:
    a. Automóvil: Posible obtención de información de desplazamiento por medio de APIs en aplicaciones como Google Maps o Waze que sean capaces de extraer y descargar los datos sobre los distintos trayectos recorridos por los usuarios de estas aplicaciones.
    b. Transporte público: Posibilidad de integrar un sistema tecnológico en autobuses y trenes que tenga la capacidad de contabilizar de manera digital el número de usuarios, así como todos los datos referentes a las rutas recorridas por cada unidad que se encuentre activa.
    c. Encuestas digitales sobre movilidad peatonal.
    d. Integración con aplicaciones como Facebook, Instagram y Twitter para extraer información anónima por medio del uso de APIs de las ubicaciones donde el usuario se encuentra cada vez que realiza un “check-in”
2. Incluyendo cada vez más ciudades con los datos de movilidad que se tenga disponibles de estas.

Toda esta información es importante, no solo en lo relativo a cada servicio que ofrece sus servicios de manera individual, sino que en el terreno del análisis de información se permite tener la posibilidad de conocer patrones y perfiles de los usuarios. Gracias a un buen análisis de los datos se puede llevar a cabo cualquier estrategia u objetivo que permita seguir creciendo y mejorando el sistema de movilidad de las ciudades que presentan mayores problemas.

De esta manera, es importante plantearse los beneficios que se pueden generar al aplicar estas mejoras sugeridas al Data Lake. Teniendo un fácil acceso a la información, siendo un almacenamiento económico y que a su vez permite múltiples procesos en paralelo de análisis, será posible obtener mejores y rápidos resultados cada vez que se requieran. No obstante, y aunque todos los servicios del Data Lake no son iguales y variarán en función de las necesidades de cada uno, el propósito es uno solo y este es hacer uso de las nuevas tecnologías que han surgido en los últimos años para conducir análisis más completos que a su vez permitan mejorar la movilidad dentro las ciudades del país.

BIBLIOGRAFÍA

[1] https://www.dineroenimagen.com/actualidad/mala-planeacion-el-principal-problema-de-movilidad-en-mexico/114915

[2] https://www.forbes.com.mx/las-ciudades-con-mejor-movilidad-en-mexico-segun-el-imco/

[3] https://www.lja.mx/2019/02/anatomia-de-la-movilidad-en-mexico-agenda-urbana/

[4] Secretaría de Desarrollo Agrario, Territorial y Urbano. Anatomía de la movilidad en México. Hacia dónde vamos. Pag 2 – 8.

[5] https://www.eluniversal.com.mx/ciencia-y-salud/plan-de-movilidad-en-bicicleta-para-una-cdmx-sustentable

[6] Lizbeth López Gómez, La bicicleta como medio de transporte en la Movilidad Sustentable. Mayo, 2018. Pag 2 – 3.

[7] https://www.elsoldemexico.com.mx/metropoli/cdmx/uber-abre-datos-de-sus-viajes-en-la-cdmx-4168506.html

[8] https://www.forbes.com.mx/uber-de-la-innovacion-a-la-realidad/

[9] http://bibliodigitalibd.senado.gob.mx/bitstream/handle/123456789/3971/CA_23.pdf?sequence=1&isAllowed=y

[10] https://www.educba.com/what-is-data-modeling/

[11] https://medium.com/@peerxp/the-6-stages-of-data-processing-cycle-3c2927c466ff

[12] Ma. Victoria Nevado Cabello, Introducción a las bases de datos relacionales. Editorial: Vision Libros, pp 17-20.

[13] W. H. Inmon, Building the Data Warehouse. Publisher: Wiley, Chapter 2. The Data Warehouse Environment.

[14] https://www.powerdata.es/data-warehouse

[15] https://docs.microsoft.com/en-us/azure/architecture/data-guide/relational-data/etl

[16] Bill Inmon, Data Lake Architecture: Designing the Data Lake and Avoiding the Garbage Dump. Published by Technics Publications, 2016. Chapter 1 Data Lakes.

[17] https://docs.microsoft.com/en-us/azure/architecture/data-guide/relational-data/etl

[18] Hrishikesh Vijay Karambelkar, Apache Hadoop 3 Quick Start Guide. Packt Publishing, 2018. Chapter 1 Hadoop 3.0 - Background and Introduction.

[19] https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html#Introduction

[20] Thilina Gunarathne, Hadoop MapReduce v2 Cookbook - Second Edition. Published by Packt Publishing, 2015. Chapter 1. Getting Started with Hadoop v2

[21] Shrey Mehrotra; Akhil Arora, Learning YARN. Packt Publishing, 2015. Chapter 1. Starting with YARN Basics

[22] Thilina Gunarathne, Hadoop MapReduce v2 Cookbook - Second Edition. Publisher: Packt Publishing. Chapter 1. Getting started with Hadoop v2.

[23] Bill Chambers; Matei Zaharia, Spark: The Definitive Guide. Published by O'Reilly Media, Inc., 2018. Chapter 1. What is Apache Spark?

[24] https://aws.amazon.com/es/about-aws/

[25] https://aws.amazon.com/es/s3/

[26] https://aws.amazon.com/es/glue/

[27] https://aws.amazon.com/es/emr/

[28] https://docs.aws.amazon.com/es_es/redshift/latest/dg/tutorial-loading-run-copy.html
[29] https://docs.aws.amazon.com/es_es/quicksight/latest/user/amazon-quicksight-user.pdf
[30] https://www.inegi.org.mx/
[31] https://www.jalisco.gob.mx/es/gobierno/comunicados
[32] https://www.semovi.cdmx.gob.mx/