

DESARROLLO DE APLICACIÓN PARA EL PROCESAMIENTO,
ALMACENAMIENTO Y CONSULTA DE DATOS DE LA MICRO-RED INTELIGENTE
UPB

ING. DANIEL BETANCUR TRUJILLO

UNIVERSIDAD PONTIFICIA BOLIVARIANA

ESCUELA DE INGENIERÍAS

MAESTRÍA EN INGENIERÍA

MEDELLÍN

2019

DESARROLLO DE APLICACIÓN PARA EL PROCESAMIENTO,
ALMACENAMIENTO Y CONSULTA DE DATOS DE LA MICRO-RED INTELIGENTE
UPB

ING. DANIEL BETANCUR TRUJILLO

TRABAJO DE GRADO PARA OPTAR AL TÍTULO DE MAGÍSTER EN INGENIERÍA

DIRECTOR

ANDRES FELIPE EUSSE GIRALDO

MAGÍSTER EN INGENIERÍA

UNIVERSIDAD PONTIFICIA BOLIVARIANA

ESCUELA DE INGENIERÍAS

MAESTRÍA EN INGENIERÍA

MEDELLÍN

2019

28 de junio de 2019

Yo Daniel Betancur Trujillo

“Declaro que este trabajo de grado no ha sido presentado con anterioridad para optar a un título, ya sea en igual forma o con variaciones, en ésta o en cualquiera otra universidad”. Art. 92, párrafo, Régimen Estudiantil de Formación Avanzada.

Firma:

Daniel Betancur T.

AGRADECIMEINTOS

Agradezco principalmente a mi director Andres Felipe Eusse Giraldo por el gran apoyo en el desarrollo en el presente trabajo de grado y por siempre ser un soporte y guía a la hora de enfrentar los diferentes inconvenientes que se presentaron durante su desarrollo. También agradezco a la Universidad Pontificia Bolivariana que a través del CIDI financio en su totalidad mis estudios de Maestría. A la Micro-Red Inteligente UPB por permitirme hacer parte de un excelente grupo de trabajo y brindarme las herramientas para desarrollar los diferentes objetivos del proyecto. Al grupo de investigación T&D y todos sus miembros por el apoyo.

CONTENIDO

GLOSARIO.....	11
SIGLAS.....	13
INTRODUCCIÓN.....	14
OBJETIVO GENERAL.....	15
OBJETIVOS ESPECÍFICOS.....	15
1. MARCO TEÓRICO.....	16
1.1. MICRO-RED INTELIGENTE.....	16
1.2. BASES DE DATOS.....	19
1.2.1. RELACIONALES.....	20
1.2.2. NO RELACIONALES (NO SQL).....	21
1.3. DINÁMICA DE PUBLICACIONES CIENTÍFICAS Y TENDENCIAS EN INVESTIGACIÓN.....	23
2. ANÁLISIS DE DATOS.....	28
2.1. ESTACIÓN METEOROLÓGICA.....	28
2.2. MEDIDORES DE RADIACIÓN.....	28
2.3. SFV [5kW _p].....	29
2.4. SFV [12.5kW _p] EV - EP.....	29
2.5. SFV [10kW _p] EP x2.....	30
2.6. SFV [28kW _p] 11B.....	30
2.7. SFV [26kW _p] 11C x2.....	31
2.8. CASA HÁBITAT.....	31
2.9. LUMINARIAS ILUPLUS.....	32
2.10. MEDIDORES INTELIGENTES.....	33
2.11. VARIABLE CALCULADAS.....	34

3.	BASE DE DATOS.....	36
3.1.	SELECCIÓN DE SISTEMA GESTOR DE BASE DE DATOS (DBMS)	36
3.1.1.	DBMS.....	36
3.1.2.	CRITERIOS.....	41
3.1.3.	EVALUACIÓN.....	43
3.2.	ESTRUCTURA.....	45
3.3.	VOLUMEN DE DATOS.....	47
3.4.	INSTALACIÓN <i>MONGODB</i>	49
3.4.1.	DESCARGA <i>MONGODB 4.0 COMMUNITY EDITION</i>	49
3.4.2.	EJECUCIÓN DEL INSTALADOR DESCARGADO	49
3.5.	MIGRACIÓN UBIDOTS	53
3.5.1.	CONEXIÓN <i>MONGODB</i>	54
3.5.2.	<i>GET VARIABLES</i>	54
3.5.3.	<i>GET VALORES</i>	55
3.5.4.	<i>INSERCIÓN MONGODB</i>	56
4.	INTERFAZ DE PROGRAMACIÓN DE APLICACIONES (API).....	57
4.1.	PETICIONES GET.....	58
4.1.1.	<i>GET SUBSISTEMAS</i>	58
4.1.2.	<i>GET VARIABLES</i>	59
4.1.3.	<i>GET COLECCIONES</i>	60
4.1.4.	<i>GET VALORES</i>	62
4.2.	PETICIONES POST.....	63
4.3.	PETICIONES PUT.....	63
4.3.1.	<i>PUT SUBSISTEMAS</i>	63
4.3.2.	<i>PUT VARIABLES</i>	65

4.3.3.	PUT COLECCIONES	65
4.3.4.	PUT VALORES	66
4.4.	PETICIONES DELETE	66
4.4.1.	DELETE SUBSISTEMAS.....	66
4.4.2.	DELETE VARIABLES.....	67
4.4.3.	DELETE COLECCIONES.....	67
4.4.4.	DELETE VALORES.....	68
5.	PROCESAMIENTO.....	69
5.1.	IMPUTACIÓN	69
5.2.	VARIABLES CALCULADAS	71
6.	CONCLUSIONES.....	73
7.	TRABAJO FUTURO	75
	BIBLIOGRAFIA.....	76
	ANEXO A. APLICACIONES	79
	ANEXO B. CONFIGURACIÓN APLICACIONES	80
	UBIDOTSToMONGO.PY	80
	UREDDBImpUTATION.PY.....	80
	UREDDBCALCULATEDVARIABLES.PY	80
	UREDDBAPI.PY.....	80
	ANEXO C. COLECCIONES Y DOCUMENTOS INSERTADOS	81

LISTA DE TABLAS

Tabla 1. Variables estación meteorológica	28
Tabla 2. Variables medidores de radiación	28
Tabla 3. Variables SFV [5kW _p]	29
Tabla 4. Variables SFV [12.5kW _p]	29
Tabla 5. Variables SFV [10kW _p]	30
Tabla 6. Variables SFV [28kW _p] 11B	30
Tabla 7. Variables SFV [10kW _p]	31
Tabla 8. Variables Casa Hábitat	32
Tabla 9. Variables Luminarias Iluplus	32
Tabla 10. Variables Medidores inteligentes	34
Tabla 11. Variables Medidores inteligentes	35
Tabla 12. Escala de evaluación DBMS	43
Tabla 13. Resultados evaluación DBMS	45
Tabla 14. Resumen de datos por subsistema	47
Tabla 15. Documentos por año por subsistema	48
Tabla 16. Almacenamiento requerido 2019-2030	49

LISTA DE FIGURAS

Figura 1. Componentes de una red inteligente.....	16
Figura 2. Flujo de energía e información en una micro-red.....	18
Figura 3. Dinámica de publicaciones relacionadas con bases de dato SQL y NoSQL.....	24
Figura 4. Principales países de publicación.	24
Figura 5. Principales instituciones de publicación bases de datos SQL	25
Figura 6. Principales instituciones de publicación bases de datos NoSQL.....	25
Figura 7. Principales autores de publicación bases de datos SQL.....	26
Figura 8. Principales autores de publicación bases de datos NoSQL	26
Figura 9. Principales temas de investigación relacionado con bases de datos SQL.....	27
Figura 10. Principales temas de investigación relacionado con bases de datos NoSQL.....	27
Figura 11. Evaluación DBMS necesidades Micro-Red Inteligente UPB	44
Figura 12. Evaluación DBMS comunidad existente.....	44
Figura 13. Evaluación DBMS experiencia autor	44
Figura 14. Ejemplo estructura embebida.....	46
Figura 15. Ejemplo estructura referencial.....	46
Figura 16. Ejemplo estructura clave-valor	46
Figura 17. Asistente de instalación MongoDB	50
Figura 18. Licencia MongoDB.....	50
Figura 19. Tipo de instalación MongoDB	51
Figura 20. Instalación MongoDB Compass.....	51
Figura 21. Servicio de Windows MongoDB	52
Figura 22. Comienzo instalación MogoDB	52
Figura 23. Finalización instalación instancia MongoDB	53
Figura 24. Funcionamiento script de migración.....	54
Figura 24. Estructura variables Ubidots	55

Figura 25. Estructura valor Ubidots.....	56
Figura 17. Asistente de instalación MongoDB	57
Figura 26. Datos SFV [10KWP] 1 Potencia	71
Figura 27. Imputación SFV [10KWP] 1 Potencia.....	71

GLOSARIO

Atomicidad: Propiedad de base de datos que garantiza que las operaciones se realizan completas o no se realizan, es decir, las operaciones se ejecutan al 100% o no son ejecutadas.

Aislamiento: Propiedad de base de datos que asegura que la ejecución de dos o más transacciones sobre la misma información sea independiente y no generen ningún tipo de error.

Base de datos: Conjunto de datos almacenados de forma estructurada.

Base de datos relacional: Conjunto de datos almacenados de forma estructurada en tablas que se relacionan entre sí, brindando una estructura lógica a los datos almacenados.

Base de datos no relacionales: Conjunto de datos almacenados de forma estructurada en la que prima la escalabilidad horizontal, la velocidad de consulta y libertad de en el esquema de datos.

Centro de control: Lugar donde se realiza la visualización, control y análisis de datos de una red eléctrica, ya sea autónoma o interconectada. Es el responsable de llevar un monitoreo constante y realizar disparos cuando las variables superan valores que pueden llegar a ser nocivos para el sistema en el que está conectado.

Consistencia: Propiedad de base de datos que asegura que cualquier transacción llevará a la base de datos desde un estado válido a otro también válido.

Durabilidad: Propiedad de base de datos que garantiza que, una vez realizada la operación, esta persistirá y no sea posible deshacerla.

Endpoint: Dispositivo remoto que permite comunicación bidireccional con la red a la que está conectado.

EPOCH: Formato para definir estampas de tiempo y corresponde al número de segundos que han transcurrido desde el primero de enero de 1970.

Hash: Corresponde a un valor o cadena de caracteres que representa una cadena de caracteres original, por lo general posee una longitud menor que el valor original. Se usa generalmente para indexación.

Micro-Red Inteligente: Sistema de distribución de energía eléctrica, que opera bien sea en esquema de islas o conectado a la red eléctrica principal, integrando de manera eficiente recursos de generación, esquemas de control, protecciones y gestión del usuario a través de plataformas basadas en TICs.

URL: Es la dirección específica que se asigna a cada uno de los recursos disponibles en la red con la finalidad de que estos puedan ser localizados o identificados.

SIGLAS

AC	Corriente alterna (<i>Alternating Current</i>)
ACID	Atomicidad, Consistencia, Aislamiento y Durabilidad (<i>Atomicity, Consistency, Isolation and Durability</i>)
API	Interfaz de programación de aplicaciones (<i>Application Programming Interface</i>)
BSON	JSON binario (<i>Binary JSON</i>)
CRUD	Crear, leer, actualizar y borrar (<i>Create, Read, Update and Delete</i>)
DB	Base de datos (<i>database</i>)
DBMS	Sistema gestor de base de datos (<i>Database Management System</i>)
DC	Corriente directa (<i>Direct Current</i>)
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet de las cosas (<i>Internet of Things</i>)
JSON	Notación de objeto de JavaScript (<i>JavaScript Object Notation</i>)
MVCC	Control de concurrencia multi-versión (<i>Multiversion Concurrency Control</i>)
SQL	Lenguaje de consulta estructurado (<i>Structured Query Language</i>)
TICs	Tecnologías de la información y la comunicación
UPB	Universidad Pontificia Bolivariana
URL	Localizador Uniforme de Recursos (<i>Uniform Resource Locator</i>)

INTRODUCCIÓN

La Universidad Pontificia Bolivariana con mira en la sostenibilidad energética del país y el desarrollo tecnológico en el ámbito de generación de energías renovables no convencionales, se encuentra en el desarrollo del proyecto Micro-Red Inteligente UPB, pretendiendo integrar subsistemas gestionables centralmente, para el uso racional y eficiente de los recursos energéticos. La ejecución del proyecto se desarrolla en tres fases: La primera fase corresponde a la implementación de la infraestructura de la Micro-Red; la segunda fase pretende investigar, probar y evaluar el desempeño de cada componente de la Micro-Red, se desarrollarán prototipos y se integrarán otros existentes; en la tercera fase se diseñarán Micro-Redes escaladas y soluciones a la medida para potenciales clientes, según sus necesidades [1], [2].

A día de hoy el proyecto Micro-Red Inteligente UPB se encuentra en la segunda fase y cuenta con un centro de control ubicado el bloque 6 del campus Medellín, desde donde se monitorean variables de diferentes subsistemas del proyecto: generación solar fotovoltaica, gestión de vehículos eléctricos y estación de carga modular de vehículos eléctricos - *carports*, sistema de aprovechamiento de desechos orgánicos (biodigestor) con acople a generador eléctrico, estación meteorológica, circuito de alumbrado público de alta eficiencia, vivienda sustentable *Habitat-Smart Living Lab* y monitoreo de variables eléctricas asociadas a las diferentes subestaciones del campus [3].

Con el crecimiento de la Micro-Red inteligente UPB se evidencia el aumento en el número de variables que se monitorean desde el centro de control de la Micro-Red. Actualmente se cuenta con un servicio en la nube de gestión y almacenamiento de datos (*Ubidots*); servicio que cuenta con limitaciones como, la conexión a internet y la respuesta de sus servidores. Debido a esto se evidencian dificultades con las peticiones que se realizan al servicio, generando inconvenientes en las aplicaciones de tiempo real del centro de control y en la operación de la Micro-red. Adicionalmente el servicio cuenta con una interfaz de usuario poco flexible, donde esta no se ajusta a las necesidades del usuario, sino que el usuario debe acoplarse a los servicios que permite *Ubidots*. Las dificultades y limitaciones presentadas hacen ver la necesidad de desarrollar una base de datos local, la cual albergue la información necesaria para monitorear la Micro- Red inteligente UPB desde su centro de control. Dicha base de datos debe ser escalable, permitir flexibilidad para interactuar con diferentes tipos de aplicaciones, tener una alta velocidad de lectura de datos y ser versátil en su estructura por la naturaleza de los datos que almacena.

El presente trabajo cuenta con siete capítulos. El primero capítulo (1) corresponde al marco teórico del proyecto, donde se describen los principales conceptos del proyecto y se realiza una investigación del estado del arte. En el segundo capítulo (2) se realiza una caracterización de los datos que se monitorean en la Micro-Red inteligente UPB. El tercer capítulo (3) es dedicado a la selección del DBMS, estructuración de la base de datos, cálculo del volumen de datos esperado y la migración de los datos existentes a la nueva base de datos. En el capítulo cuatro (4) se explica las diferentes funcionalidades de la interfaz de programación de aplicaciones (API). El capítulo cinco (5) expone el tratamiento que se llevó a cabo con la información existente y los resultados obtenidos. El capítulo seis (6) corresponde a las conclusiones del proyecto de acuerdo con los resultados obtenidos y los objetivos establecidos. Finalmente se tiene en el capítulo siete (7) los trabajos y desarrollos futuros que se pueden realizar con base en el presente trabajo de grado.

OBJETIVO GENERAL

Implementación de una aplicación para el procesamiento, almacenamiento y consulta de datos de las variables a monitorear de la Micro-Red inteligente ubicada en el campus de laureles de la Universidad Pontificia Bolivariana – Sede Medellín. Dicha base de datos se alojará en el servidor que se encuentra en el centro de control de la Micro-Red inteligente UPB.

OBJETIVOS ESPECÍFICOS

- Dimensionar la base de datos, estimando su tamaño, su estructura y crecimiento.
- Procesamiento de datos con el fin de calcular y almacenar variables de interés no medidas.
- Realizar un análisis temporal de las múltiples variables de la base de datos, mediante técnicas de *Big Data*, con el propósito encontrar “huecos” y de ser posible estimar los valores faltantes.
- Implementar una herramienta que permita la consulta de información a la base de datos.
- Elaborar memorias descriptivas de la instalación, configuración e implementación de las diferentes herramientas usadas para llevar a cabo el proyecto.
- Elaborar un documento en el que se expliquen las características de la herramienta finalizada.
- Elaborar manual de la aplicación, que permita conocer su funcionamiento y dar soporte a la herramienta.

1. MARCO TEÓRICO

1.1. MICRO-RED INTELIGENTE

Una red inteligente o *Smart Grid* es un sistema de distribución de energía eléctrica, que opera bien sea en esquema de islas o conectado a la red eléctrica principal, integrando de manera eficiente recursos de generación distribuida, esquemas de control, protecciones y gestión del usuario a través de plataformas basadas en TICs [4], [5]. En la Figura 1 se puede apreciar un esquema de red inteligente en el cual se puede apreciar elemento central de la red, el centro de control, encargado de integrar, monitorear y operar la red inteligente, a su alrededor se encuentran sistemas inteligentes de consumo y recursos de generación distribuidos.

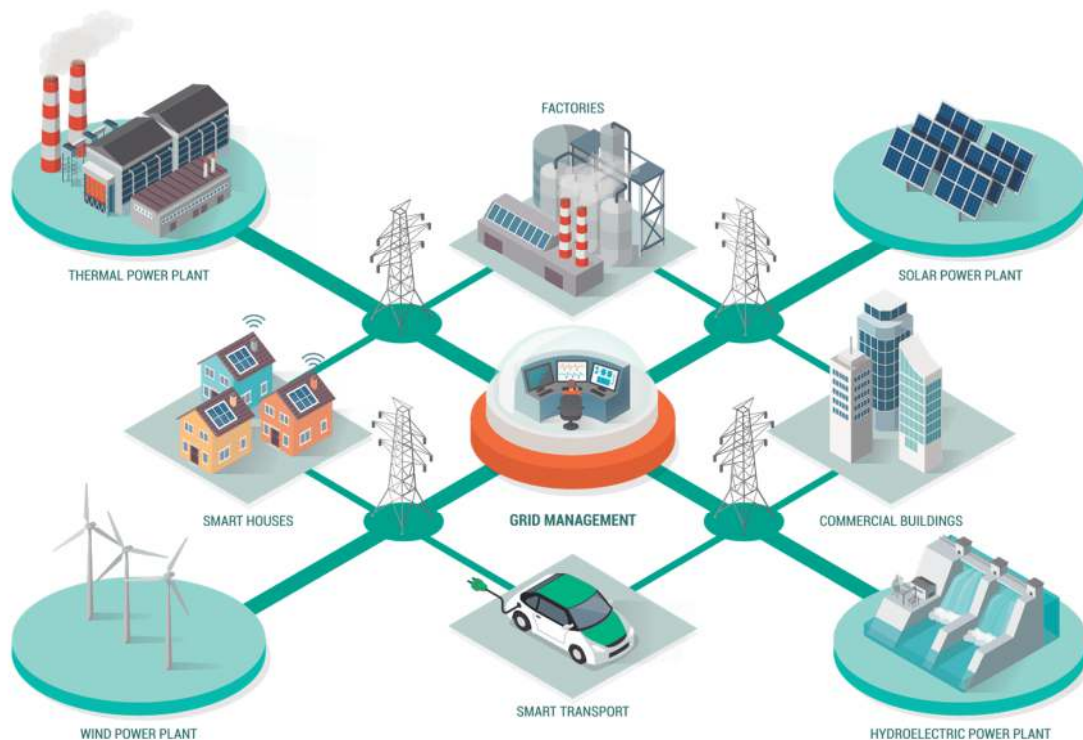


Figura 1. Componentes de una red inteligente

Fuente: Tomado de [6]

Una red eléctrica considerada inteligente, debe poseer al menos las siguientes dos características fundamentales [5]:

- Recursos de generación y demanda gestionables dentro de la red
- Capacidad de operar en isla o conectada a la red eléctrica principal

Las redes inteligentes también pueden incluir otras características que hacen uso de los diferentes avances tecnológicos para lograr sus principales objetivos, de acuerdo a la concepción propia de cada red inteligente. A continuación se presentan algunas cualidades adicionales que pueden incluir las redes inteligentes [4], [6]:

- Participación activa de los consumidores
- Demanda controlable: Puede limitarse o cambiarse
- Generación distribuida mediante uso fuentes de energías no convencionales de paneles, generalmente renovables, llamadas tecnologías limpias
- Almacenamiento de energía en momentos de baja demanda
- Nuevos productos, servicios y mercados
- Calidad de la energía para la economía digital
- Uso óptimo de los activos y operación eficiente
- Mediante dispositivos inteligentes anticipa y responde a las perturbaciones del sistema
- Tiene la capacidad de recuperarse contra ataques y desastres naturales

El funcionamiento de las micro-redes ofrece distintas ventajas para los consumidores y las empresas de servicios públicos, por ejemplo, mejora de la eficiencia energética, minimización del consumo energético, reducción del impacto ambiental, mejora de la fiabilidad del servicio, reducción de pérdidas en el sistema, control de voltaje, seguridad en el sistema y mayor rentabilidad en la reposición de infraestructura eléctrica [5].

Las micro-redes inteligentes pueden ser clasificadas en tres tipos de acuerdo a su fuente de alimentación [7]:

- Micro-red AC: son redes inteligentes que se asemejan a las redes de distribución convencionales, su alimentación es en corriente alterna.
- Micro-red DC: son redes inteligentes que alimentan un grupo de cargas en corriente directa. Este tipo de micro-redes proporciona una mayor eficiencia y una mejor respuesta ante cortocircuitos.
- Micro-red Híbrida: son redes inteligentes que combinan redes de distribución en corriente alterna y corriente directa. Los propósitos de implementar una micro-red híbrida son: minimizar las etapas de conversión, aumentar la confiabilidad y reducir los costos de energía, mejorando así la eficiencia general de la red.

Al integrar una gran variedad de tecnologías y servicios de computación y comunicación digital en la infraestructura del sistema eléctrico, las micro-redes van más allá de los medidores de

energía inteligentes para hogares y empresas, ya que permite contemplar una gran cantidad de funcionalidades en la operación del sistema eléctrico, gracias a los flujos bidireccionales de energía y de comunicación y control; en donde la gestión está dada en el control de equipos y cargas, permitiendo opciones de mayor productividad tanto en los operadores de servicios públicos como en los usuarios [8].

Desde el punto de vista normativo, en el año 2011, se desarrolla el estándar IEEE Std 2030, el cual tiene el objetivo de describir un conjunto de buenas prácticas que permitan la interoperabilidad de la red inteligente con las tecnologías de la información. Esta estándar proporciona definiciones y orientación en el diseño e implementación de componentes de redes inteligentes y aplicaciones de uso final; debido a que la interoperabilidad se logra a través de una amplia colaboración de todos los participantes. La guía permite identificar los desafíos y oportunidades de los diferentes roles de un proyecto de red inteligente, logrando definir tres perspectivas arquitectónicas integradas: sistemas de energía, tecnología de comunicaciones y tecnología de la información; las cuales indican las principales directrices para la interoperabilidad de una red inteligente [8]–[10].

La interoperabilidad se basa en la capacidad de dos o más redes, sistemas, dispositivos, aplicaciones o componentes para intercambiar información y hacer uso de esta de forma segura y efectiva [8]. En la Figura 2 se observan los flujos de energía y los flujos de información que se intercambian entre los diferentes componentes un esquema de micro-red. Inteligente.

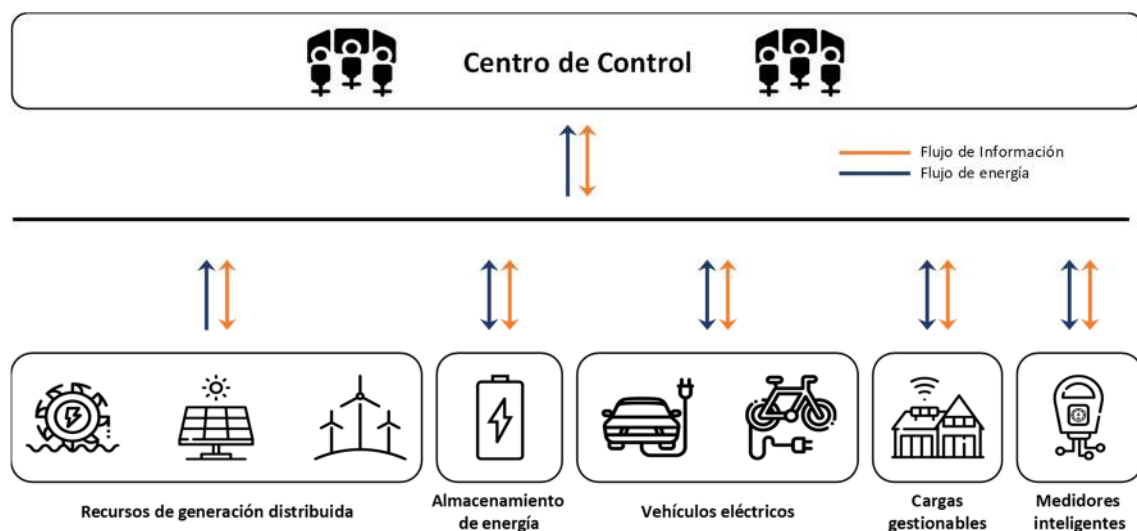


Figura 2. Flujo de energía e información en una micro-red

Fuente: Elaboración propia, iconos tomados de [11]

El concepto Internet de las Cosas (IoT) es el término que permite lograr la interoperabilidad de una red inteligente o micro-red, ya que les brinda la capacidad, a diferentes componentes o equipos de un sistema, de transferir y recibir datos a través de una red, donde cada elemento que interactuar en la red tiene un identificador único [9], [10], [12].

1.2. BASES DE DATOS

Las bases de datos corresponden a una colección de datos que se encuentran organizados de acuerdo con una estructura establecida, es decir, es un almacenamiento estructurado de datos. La finalidad de las bases de datos es operar grandes cantidades de información permitiendo el ingreso, el almacenamiento, la consulta y la administración de esa información [13], [14].

El sistema gestor de base de datos, DBMS por sus siglas en inglés, corresponden a una capa de software que permiten crear, leer, actualizar y borrar (CRUD) los datos que se albergan en las bases de datos. Es una herramienta de propósito general útil para estructurar, almacenar y controlar los datos ofreciendo interfaces de acceso a la base de datos. Tareas fundamentales que desempeñan estos sistemas hacen referencia a la seguridad de acceso a los datos, al mantenimiento de la integridad de los datos, a mecanismos de recuperación debidos a fallos físicos y lógicos, al control de concurrencia en el momento de acceder a los datos y a la eficiencia del sistema evaluada, generalmente, en términos del tiempo de respuesta a las consultas de los usuarios [14]–[16].

Los modelos de datos se definen como las características (descripción, relaciones, y restricciones) que permiten especificar una estructura lógica de datos particular, es decir, un conjunto de restricciones que delimitan las particularidades sobre esta estructura y un conjunto de mecanismos para manipular los datos. Para especificar la estructura de la descripción y las restricciones se hace uso de lenguajes de definición de datos (DDL) y para especificar la manipulación de los datos se hace uso de lenguajes de manipulación de datos (DML). Los DML permiten la consulta de datos del estado actual de la base de datos y la actualización de los datos, creando así un nuevo estado de la base de datos. Los modelos de datos integran los siguientes tres componentes [15], [16]:

- Componente estructural: describe los tipos de estructuras de datos
- Componente de manipulación de datos: reglas u operadores para manipular los datos
- Componente de integridad: reglas de integridad para asegurar estados consistentes de la base de datos

De acuerdo a la abstracción de la información se pueden categorizar los modelos de datos en las siguientes categorías: modelos lógicos basados en registros y modelos físicos [15], [16].

1.2.1. RELACIONALES

Una base de datos relacional consiste en un conjunto de datos, organizados en tablas que se relacionan entre sí, con el principal objetivo de brindar al usuario una percepción o visión de la base de datos como una estructura lógica que consiste en un conjunto de relaciones y no como una estructura física de implementación. Esto ayuda a conseguir un alto grado de independencia de los datos[14]–[16].

Las bases de datos relacionales usan un lenguaje de consulta estructurado (*SQL*), el cual permite realizar diferentes acciones de administración sobre la base de datos, la estructura básica de una expresión SQL consiste en tres sentencias: *select*, *from* y *where*, las cuales permiten crear, leer, actualizar y borrar los diferentes componentes de la base de datos [15].

Las bases de datos relacionales poseen las siguientes características [15], [16]:

- Utilizan el lenguaje de comunicación estándar SQL
- Esquemas rígidos definidos
- Garantiza las propiedades ACID
- Tienen un modelo de datos único

Las cinco base de datos relacionales más populares son [17]:

1. *Oracle*
2. *MySQL*
3. *Microsoft SQL Server*
4. *PostgreSQL*
5. *IBM DB2*

A continuación se presentan la ventajas y desventajas que presentan las bases de datos relacionales [18], [19]:

Ventajas

Madurez: Lleva más de 40 años de funcionamiento; es un lenguaje respetado, consolidado y aceptación por la comunidad de desarrolladores. Existe una gran variedad y cantidad de información para que permite facilitar y agilizar el desarrollo.

Atomicidad: Propiedad que garantiza que las operaciones se realizan completas o no se realizan, es decir, las operaciones se ejecutan al 100% o no son ejecutadas.

Consistencia: Propiedad que asegura que cualquier transacción llevará a la base de datos desde un estado válido a otro también válido.

Aislamiento: Propiedad que asegura que la ejecución de dos o más transacciones sobre la misma información sea independiente y no generen ningún tipo de error.

Durabilidad: Propiedad que garantiza que una vez realizada la operación, esta persistirá y no sea posible deshacerla.

Estándares bien definidos: Lenguaje que se rige por patrones, lo que garantiza que los cambios en su estructura no sean sustanciales y no varíen de fabricante a fabricante.

Sencillez en la escritura: se asemeja mucho al lenguaje humano, la comprensión de las operaciones que se programan puede ser escritas por personas que no tengan grandes conocimientos de informática.

Desventajas

Crecimiento: cuando estas bases de datos tienden a crecer demasiado, el mantenimiento de estas se hace verdaderamente difícil y costoso, y suelen presentar fallas en tiempo de respuesta.

Cambios en la estructura: al tener esquemas rígidos definidos, los cambios que impliquen un cambio en la estructura de datos pueden llevar a grandes problemas en el desarrollo de dichos cambios.

1.2.2. NO RELACIONALES (NOSQL)

NoSQL (acrónimo de *Not only Structured Query Language*) hace referencia al conjunto de tecnologías en bases de datos que buscan alternativas al sistema de bases de datos relacional, en un contexto donde priman la velocidad, el manejo de grandes volúmenes de datos y la posibilidad de tener un sistema distribuido [20].

Son un tipo de base de datos ágil que permite los cambios de estructura ante la evolución de las aplicaciones, proporcionando siempre la funcionalidad que los desarrolladores esperan de las bases de datos tradicionales, tales como índices secundarios, un lenguaje completo de búsquedas y consistencia estricta. Las bases de datos no relacionales poseen las siguientes características[20], [21]:

- Escalabilidad horizontal: facilidad de añadir, eliminar o realizar operaciones con elementos (hardware) del sistema, sin afectar el rendimiento.
- Uso eficiente de recursos: aprovecha las nuevas tecnologías, como los discos en estado sólido, la memoria RAM y los sistemas distribuidos en general.
- Libertad de esquema: al no tener una estructura rígida definida, permite mayor libertad para modelar los datos; además facilita la integración con los lenguajes de programación orientados a objetos, lo que evita el proceso de mapeado.
- Consultas simples: Estas requieren menos operaciones y son más naturales, por lo tanto, se gana en simplicidad y eficiencia.

Las cinco base de datos *NoSQL* más populares son [17]:

1. MongoDB
2. Redis
3. Elasticsearch
4. Cassandra
5. Splunk

A continuación se presentan la ventajas y desventajas que presentan las bases de datos *NoSQL* [18], [19]:

Ventajas

Versatilidad: la principal ventaja por la cual esta nueva tecnología difiere de las demás soluciones de bases de datos es la versatilidad que ofrece a crecimientos o cambios sobre la forma como almacena la información.

Escalabilidad Horizontal: soportan estructuras distribuidas.

Disponibilidad de recursos: no se requieren servidores con características de alto rendimiento disponibles para operar.

Optimización: los sistemas *NoSQL* tienen un algoritmo interno para reescribir las consultas escritas por los usuarios o las aplicaciones programadas, esto con el fin de no sobrecargar el rendimiento de los servidores y mantener un nivel óptimo en las operaciones.

Desventajas

Atomicidad: la información en ocasiones no es consistente, puede ser diferente en cada uno de los nodos replicas que se puedan configurar en la arquitectura de base de datos.

Documentación del Software: Dado que NoSQL, es relativamente nuevo, las operaciones pueden ser limitadas y se requiera de conocimientos avanzados con el uso de la herramienta y las personas que se encuentran realizando estos desarrollos en el software tengan que invertir más tiempo en los desarrollos.

Estándares en el lenguaje: No se tiene un estándar definido entre los diferentes motores de bases de datos NoSQL.

1.3. DINÁMICA DE PUBLICACIONES CIENTÍFICAS Y TENDENCIAS EN INVESTIGACIÓN

La vigilancia tecnológica es un proceso organizado, selectivo y sistemático, que permite la captación, selección y análisis de la información, para convertirla en conocimiento con el fin de tomar decisiones con menor riesgo y poder anticiparse a los cambios. El desarrollo de la vigilancia tecnológica en este caso será enfocado en la identificación de las principales tendencias en uso de bases de datos en micro-redes, la dinámica de publicaciones científicas, los principales autores, países e instituciones que publican sobre este tema y la obtención de las tendencias en investigación relacionadas.

Para el desarrollo de la vigilancia tecnológica se empleará la base de datos de artículos científicos *Scopus*, la cual es una base de datos bibliográficas que permite la obtención de citas y resúmenes de artículos de revistas científicas revisadas por pares en diferentes áreas de la ciencia como: la estadística, las matemáticas, la psicología, la ingeniería eléctrica y electrónica. La búsqueda se realizó en el periodo de tiempo comprendido entre el año 2000 y el 2019 (presente), con lo que se busca obtener los resultados más recientes relacionadas con este tema.

Se construyeron dos ecuaciones de búsqueda, empleando “*database*” como el termino clave principal y los términos “*NoSQL*” y “*SQL*” con el fin de acotar las búsquedas. A continuación en la ecuación (1) y en la ecuación (2), se presentan las ecuación de búsqueda empleadas.

$$(TITLE-ABS-KEY ("database") AND PUBYEAR > 2009) AND ("SQL") \quad (1)$$

$$(TITLE-ABS-KEY ("database") AND PUBYEAR > 2009) AND ("NoSQL") \quad (2)$$

Al momento de la búsqueda se identificaron, para el periodo de tiempo definido, un total de 9.625 documentos científicos relacionados con bases de datos SQL y 2.942 documentos científicos relacionados con bases de datos NoSQL.

En la Figura 3 se presenta la dinámica de publicaciones de artículos científicos relacionados con bases de datos SQL y bases de datos NoSQL. Allí se observa una tendencia estable para las bases

de datos SQL y una tendencia creciente en los últimos años para las bases de datos NoSQL, con un pico máximo de publicaciones en el 2017.

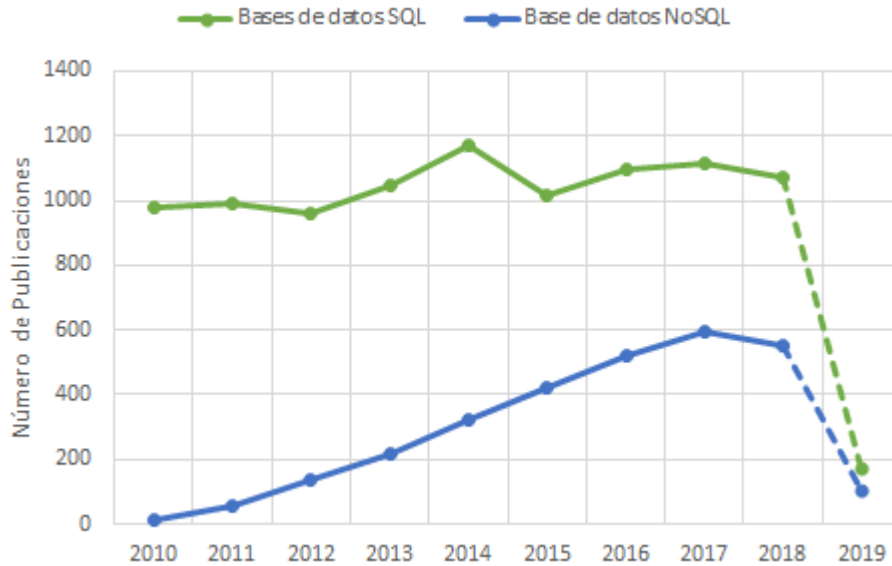


Figura 3. Dinámica de publicaciones relacionadas con bases de dato SQL y NoSQL

Fuente: Elaboración propia, información toma de [22]

En la Figura 4 se presentan los principales países de publicación de artículos científicos relacionados con bases de datos SQL y bases de datos NoSQL. En dicha figura se observa que Estados Unidos es el país con el mayor número de publicaciones, seguido por China e India.

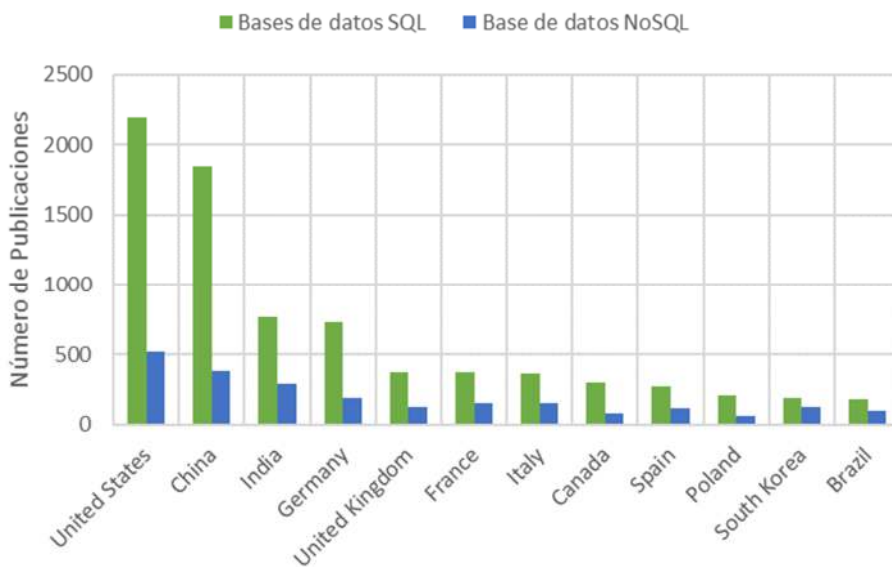


Figura 4. Principales países de publicación.

Fuente: Elaboración propia, información toma de [22]

En la Figura 5 se presentan las instituciones de mayor número de publicaciones científicas enfocadas en bases de datos no SQL. En esta lista de instituciones se destacan entidades académicas y compañías de tecnología.

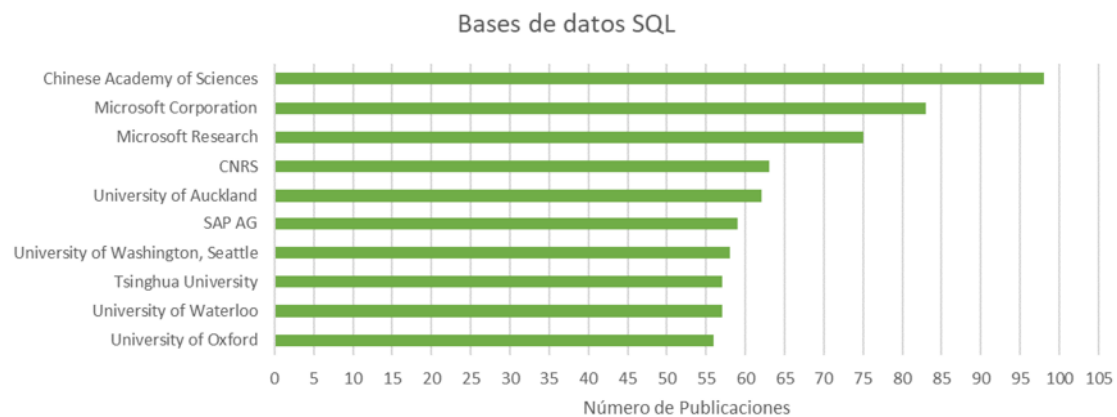


Figura 5. Principales instituciones de publicación bases de datos SQL

Fuente: Elaboración propia, información toma de [22]

En la Figura 6 se presentan las instituciones de mayor número de publicaciones científicas enfocadas en bases de datos no NoSQL. En esta lista de instituciones se destaca que únicamente aparecen instituciones académicas.

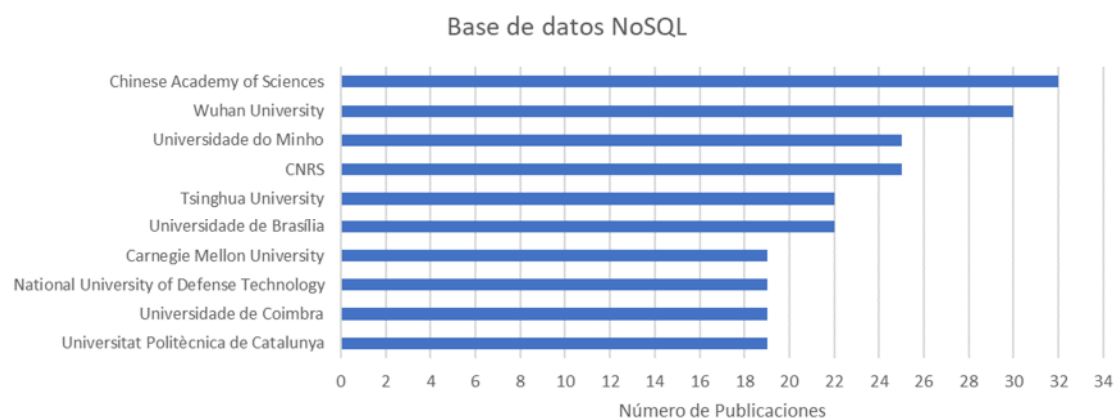


Figura 6. Principales instituciones de publicación bases de datos NoSQL

Fuente: Elaboración propia, información tomada de [22]

En el caso de los países, autores e instituciones de mayor publicación, la sumatoria de los documentos puede ser mayor a la descrita en los resultados de búsqueda, esto debido a que pueden existir trabajos colaborativos entre instituciones y autores de diferentes países.

En la Figura 7 se presentan los principales autores de publicación en en bases de datos SQL.

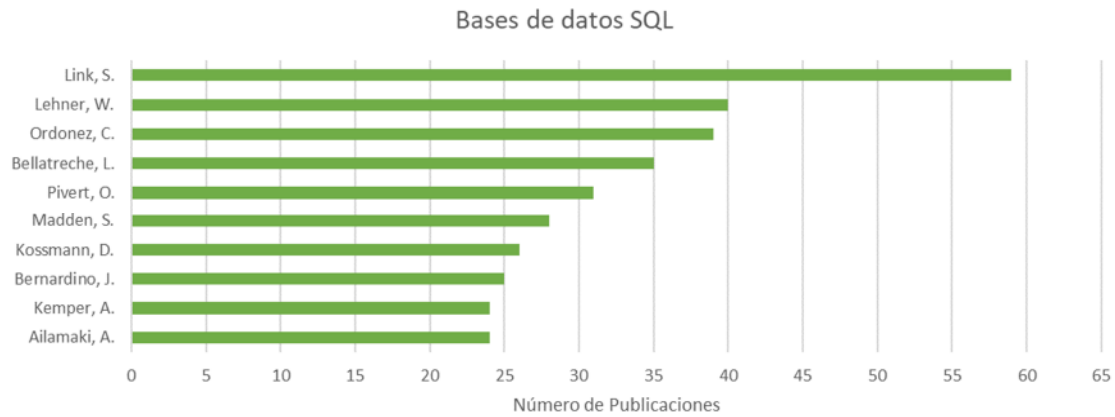


Figura 7. Principales autores de publicación bases de datos SQL

Fuente: Elaboración propia, información tomada de [22]

En la Figura 8 se presentan los principales autores de publicación en en bases de datos NoSQL.

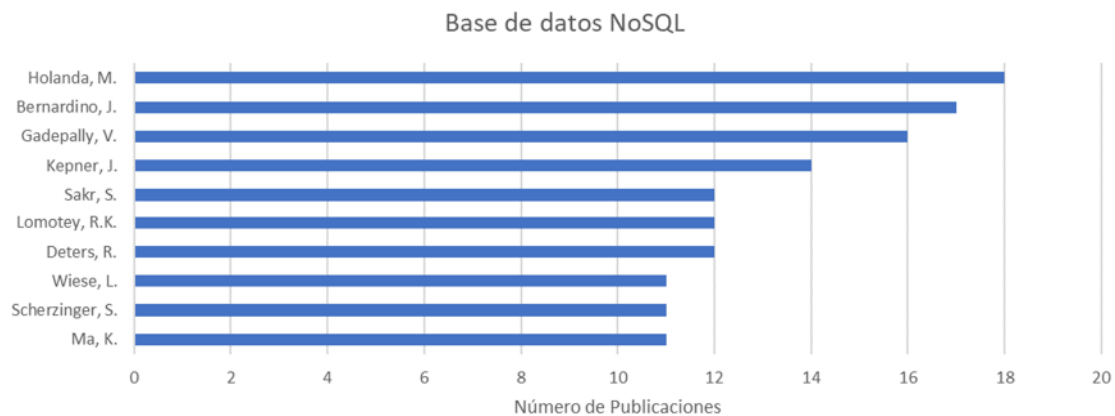


Figura 8. Principales autores de publicación bases de datos NoSQL

Fuente: Elaboración propia, información tomada de [22]

Obtenidos los artículos relacionados con bases de datos SQL y NoSQL, se extrajeron los temas de mayor mención que son tendencia de investigación en el mundo. En la Figura 9 y la Figura 10 se presentan una nube de palabras con los términos que son tendencia de investigación en los documentos identificados. Allí se observa una gran atención por sistemas de bases de datos, el termino *big data*, manejo de información, minería de datos y algunos motores de bases de como *MongoDB*.



Figura 9. Principales temas de investigación relacionado con bases de datos SQL

Fuente: Elaboración propia, información tomada de [22]



Figura 10. Principales temas de investigación relacionado con bases de datos NoSQL

Fuente: Elaboración propia, información tomada de [22]

2. ANÁLISIS DE DATOS

En este capítulo se realiza un análisis de cada una de las variables de cada una de las fuentes de datos de los subsistemas pertenecientes a la Micro-Red inteligente ubicada en el campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín.

2.1. ESTACIÓN METEOROLÓGICA

Subsistema tipo medición. En la Tabla 1 se presentan las diferentes variables reportadas por la estación meteorológica y sus características.

Tabla 1. Variables estación meteorológica

Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Radiación solar	Double	64-bit	24	1 min
Velocidad del viento	Double	64-bit	24	1 min
Dirección del viento	Double	64-bit	24	1 min
Material particulado 1	Double	64-bit	24	1 min
Material particulado 2.5	Double	64-bit	24	1 min
Material particulado 10	Double	64-bit	24	1 min
ICA 2.5	Double	64-bit	24	1 min
ICA 10	Double	64-bit	24	1 min

2.2. MEDIDORES DE RADIACIÓN

Medidores de radiación ubicados en el bloque 11, en el bloque 24 y en la entrada vehicular de la circular primera del campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín, subsistema tipo medición. En la Tabla 2 se presentan las diferentes variables reportadas por los medidores de radiación y sus características.

Tabla 2. Variables medidores de radiación

Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Radiación	Double	64-bit	24	1 min
Temperatura	Double	64-bit	24	1 min

2.3. SFV [5kW_p]

Sistema solar fotovoltaico con capacidad de 5kW_p ubicado en la terraza del bloque 10 del campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín, subsistema tipo generación. Este sistema solar cuenta con 20 paneles solares, cada uno con micro inversor, en la Tabla 3 se presentan las diferentes variables reportadas por cada uno de los micro inversores del sistema solar fotovoltaico de 5kW_p y sus características

Tabla 3. Variables SFV [5kW_p]

Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Potencia	Double	64-bit	24	5 min
Energía	Double	64-bit	24	5 min
Energía total día	Double	64-bit	24	5 min
Potencia μ -inversores x 20	Double	64-bit	24	5 min

2.4. SFV [12.5kW_p] EV - EP

Sistemas solares fotovoltaicos con capacidad de 12.5kW_p cada uno, se encuentran ubicados en los techos de la entrada vehicular y la entrada peatonal de la circular primera del campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín, subsistema tipo generación. En la Tabla 4 se presentan las diferentes variables reportadas por ambos sistemas solares fotovoltaicos.

Tabla 4. Variables SFV [12.5kW_p]

Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Potencia	Double	64-bit	14	1 min
Voltaje DC	Double	64-bit	14	1 min
Corriente DC	Double	64-bit	14	1 min
Voltaje AC	Double	64-bit	14	1 min
Corriente AC	Double	64-bit	14	1 min
Frecuencia	Double	64-bit	14	1 min
Energía total día	Double	64-bit	14	1 min
Energía total	Double	64-bit	14	1 min

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Energía total Año	Double	64-bit	14	1 min

2.5. SFV [10kW_p] EP x2

Sistemas solares fotovoltaicos con capacidad de 10kW_p cada uno, ubicados ambos en el techo de la entrada peatonal de la circular primera (bloque 24) del campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín, subsistema tipo generación. En la Tabla 5 se presentan las diferentes variables reportadas por ambos sistemas solares fotovoltaicos.

Tabla 5. Variables SFV [10kW_p]

Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Potencia	Double	64-bit	14	1 min
Voltaje DC	Double	64-bit	14	1 min
Corriente DC	Double	64-bit	14	1 min
Voltaje AC	Double	64-bit	14	1 min
Corriente AC	Double	64-bit	14	1 min
Frecuencia	Double	64-bit	14	1 min
Energía total día	Double	64-bit	14	1 min
Energía total	Double	64-bit	14	1 min
Energía total Año	Double	64-bit	14	1 min

2.6. SFV [28kW_p] 11B

Sistema solar fotovoltaico con capacidad de 28kW_p ubicado en el techo del bloque 11B del campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín, subsistema tipo generación. En la Tabla 6 se presentan las diferentes variables reportadas por el sistema solar fotovoltaico del bloque 11B y sus características.

Tabla 6. Variables SFV [28kW_p] 11B

Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Potencia generación x 2	Double	64-bit	24	1 min
Potencia AC	Double	64-bit	24	1 min

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Potencia DC	Double	64-bit	24	1 min
Voltaje AC x2	Double	64-bit	24	1 min
Voltaje DC	Double	64-bit	24	1 min
Energía día	Double	64-bit	24	1 min
Energía Total	Double	64-bit	24	1 min
Potencia consumo	Double	64-bit	24	1 min
Energía día consumo	Double	64-bit	24	1 min
Energía total consumo	Double	64-bit	24	1 min
Radiación	Double	64-bit	24	1 min
Temperatura	Double	64-bit	24	1 min

2.7. SFV [26kW_p] 11C x2

Sistemas solares fotovoltaicos con capacidad de 26 kW_p cada uno, ubicados ambos en el techo del bloque 11C del campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín, subsistema tipo generación. En la Tabla 7 se presentan las diferentes variables reportadas por el sistema solar fotovoltaico del bloque 11 y sus características.

Tabla 7. Variables SFV [10kW_p]

Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Potencia x2	Double	64-bit	14	1 min
Voltaje DC x2	Double	64-bit	14	1 min
Corriente DC x2	Double	64-bit	14	1 min
Voltaje AC x2	Double	64-bit	14	1 min
Corriente AC x2	Double	64-bit	14	1 min
Frecuencia x2	Double	64-bit	14	1 min
Energía total día x2	Double	64-bit	14	1 min
Energía total x2	Double	64-bit	14	1 min
Energía total Año x2	Double	64-bit	14	1 min

2.8. CASA HÁBITAT

Smart Living Lab que cuenta con sistema de generación solar fotovoltaico y sistema de almacenamiento de baterías, se encuentra ubicado en el costado sur de la cancha sintética

(Bloque 19) del campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín, subsistema tipo generación. En la Tabla 8 se presentan las diferentes variables reportadas por el controlador de la casa Hábitat.

Tabla 8. Variables Casa Hábitat

Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Potencia DC	Double	64-bit	24	1 min
Voltaje DC	Double	64-bit	24	1 min
Corriente DC	Double	64-bit	24	1 min
Potencia AC	Double	64-bit	24	1 min
Voltaje AC	Double	64-bit	24	1 min
Corriente AC	Double	64-bit	24	1 min
Frecuencia	Double	64-bit	24	1 min
Energía total día	Double	64-bit	24	1 min
Energía total	Double	64-bit	24	1 min
Voltaje SFV	Double	64-bit	24	1 min
Corriente SFV	Double	64-bit	24	1 min
Potencia Baterías	Double	64-bit	24	1 min
Voltaje Baterías	Double	64-bit	24	1 min
Corriente Baterías	Double	64-bit	24	1 min
Tiempo Baterías	Double	64-bit	24	1 min
Potencia Red	Double	64-bit	24	1 min
Potencia consumo x2	Double	64-bit	24	1 min

2.9. LUMINARIAS ILUPLUS

Sistema de luminarias ubicadas en el campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín, subsistema tipo iluminación. En la Tabla 9 se presentan las diferentes variables reportadas el sistema de iluminación Iluplus y sus características.

Tabla 9. Variables Luminarias Iluplus

Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Energía total x5	Double	64-bit	24	1 min

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Estado x5	Double	64-bit	24	1 min
Corriente x5	Double	64-bit	24	1 min
Voltaje x5	Double	64-bit	24	1 min

2.10. MEDIDORES INTELIGENTES

Este subsistema de medición está conformado por una red de 16 medidores inteligentes en diferentes subestaciones del campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín; cada subestación se encuentra asociada al consumo de un bloque o de una dependencia del bloque, a continuación se listan las ubicaciones de los diferentes medidores:

1. Bloque 3: Rectoral
2. Bloque 4: Colegio UPB - Primaria
3. Bloque 5: Colegio UPB - Bachillerato
4. Bloque 7: CTIC
5. Bloque 7: TAC
6. Bloque 8: Laboratorios
7. Bloque 8: Aire acondicionado
8. Bloque 8: CPA
9. Bloque 9: Formación avanzada
10. Bloque 9: Aire acondicionado
11. Bloque 10: Arquitectura y diseño
12. Bloque 12: Derecho
13. Bloque 15: Biblioteca
14. Bloque 17: Polideportivo y gimnasio
15. Bloque 18: Edificio de parqueaderos
16. Casa Hábitat

En la Tabla 10 se presentan las diferentes variables reportadas por cada uno de los medidores inteligentes y sus características.

Tabla 10. Variables Medidores inteligentes
Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Potencia activa monofásica - P1 ϕ x3	Double	64-bit	24	5 min
Potencia activa trifásica - P3 ϕ	Double	64-bit	24	5 min
Potencia reactiva monofásica - Q1 ϕ x3	Double	64-bit	24	5 min
Potencia reactiva trifásica - Q3 ϕ	Double	64-bit	24	5 min
Factor de potencia monofásico - cos(Θ) x3	Double	64-bit	24	5 min
Factor de potencia total - cos(Θ)	Double	64-bit	24	5 min
Armónicos de voltaje x3	Double	64-bit	24	5 min
Distorsión armónica total voltaje - THD _v	Double	64-bit	24	5 min
Armónicos de corriente x3	Double	64-bit	24	5 min
Distorsión armónica total corriente - THD _i	Double	64-bit	24	5 min
Distorsión armónica total Potencia - THD _p	Double	64-bit	24	5 min
Energía activa importación día	Double	64-bit	24	5 min
Energía activa exportación día	Double	64-bit	24	5 min
Energía reactiva importación día	Double	64-bit	24	5 min
Energía reactiva exportación día	Double	64-bit	24	5 min
Energía activa importación total	Double	64-bit	24	5 min
Energía activa exportación total	Double	64-bit	24	5 min
Energía reactiva importación total	Double	64-bit	24	5 min
Energía reactiva exportación total	Double	64-bit	24	5 min
Frecuencia	Double	64-bit	24	5 min
Corrientes de neutro - In	Double	64-bit	24	5 min
Magnitud tensión – V x6	Double	64-bit	24	5 min
Ángulo tensión – $\angle V$ x3	Double	64-bit	24	5 min
Corriente – I x3	Double	64-bit	24	5 min
Ángulo corriente – $\angle I$ x3	Double	64-bit	24	5 min

2.11. VARIABLE CALCULADAS

Las variables calculadas son un grupo de variable indirectas que pueden ser obtenidas a partir de las diferentes variables medidas dentro de todos los subsistemas de la Micro-Red Inteligente UPB ubicada en campus laureles de la Universidad Pontificia Bolivariana – Sede Medellín. En la

Tabla 11 se presentan las diferentes variables calculadas de la Micro-Red Inteligente UPB y sus características

Tabla 11. Variables Medidores inteligentes

Fuente: Elaboración propia

Variable	Tipo de dato	Memoria requerida	Horas Efectivas	Frecuencia de posteo
Consumo potencia activa x3	Double	64-bit	24	1 min
Energy share x3	Double	64-bit	24	1 min
Potencia generada x3	Double	64-bit	24	1 min
Power share x3	Double	64-bit	24	1 min
Consumo total diario de energía activa	Double	64-bit	24	1 min
Consumo total de potencia activa	Double	64-bit	24	1 min
Power share total	Double	64-bit	24	1 min
Potencia generada total	Double	64-bit	24	1 min
Energía generada total	Double	64-bit	24	1 min
Energía generada total día	Double	64-bit	24	1 min
Energy share total	Double	64-bit	24	1 min
Emisiones totales ahorradas	Double	64-bit	24	1 min
Equivalente en arboles	Double	64-bit	24	1 min

3. BASE DE DATOS

Este capítulo se presentan el diseño y los lineamientos establecidos para la concepción de la base de datos. Primero se realiza un análisis de los diferentes tipos de DBMS, con el fin de establecer cuál de ellos se adapta de mejor manera a las necesidades de la Micro-Red Inteligente UPB. En segundo lugar, se establece la estructura de la base datos, permitiendo constituir una representación visual de esta. Como tercer paso se realiza el cálculo del volumen de datos de la base datos con el propósito de evaluar los recursos del servidor requeridos y garantizar que se tiene el nivel de rendimiento y el espacio de almacenamiento necesarios. Posterior a esto se realiza un instructivo del DBMS seleccionado. Por última, se realiza la migración de datos de *Ubidots* a la nueva base datos. Por último, se determinan los cálculos a realizar de variables no medidas a partir de variables medidas.

3.1. SELECCIÓN DE SISTEMA GESTOR DE BASE DE DATOS (DBMS)

En esta sección se hace una introducción a diferentes DBMS, se presentan los criterios que toman mayor relevancia en el momento de selección y se construye una tabla comparativa de entre los diferentes DBMS, estos se evalúan de acuerdo a los criterios descritos y se determina cual es el DBMS con el que se trabajaría.

3.1.1. DBMS

A continuación, se presentan algunos de los DBMS más populares en el momento, los cuales serán evaluados más adelante de acuerdo a los criterios establecidos.

Oracle Database

Es el DBMS más usado alrededor del mundo, posee varios tipos de modelos de datos, entre los cuales destacan, modelo relacional, modelo documental (NoSQL) y modelo orientado a grafos (NoSQL); siendo el modelo relacional el de mayor distribución. Puede correr en casi cualquier sistema operativo, de manera local o en la nube. Posee una arquitectura totalmente escalable y a menudo es utilizada por empresas globales, que administran y procesan datos en redes de área amplia y local. La base de datos *Oracle* tiene su propio componente de red para permitir las comunicaciones a través de redes [23], [24].

A continuación se presentan algunas características del DBMS [25]:

- Totalmente escalable
- Inteligencia de negocios

- Agrupamiento
- Gestión de contenidos
- Servicios de localización
- Gestión del servidor
- Inteligencia de negocios
- Alto rendimiento, seguridad y análisis

MySQL

MySQL es el DBMS de código abierto más popular del mundo; se distribuye en varias versiones, una versión *Community*, distribuida bajo la Licencia GNU, y varias versiones *Enterprise*, las cuales incluyen productos o servicios adicionales tales como herramientas de monitoreo y asistencia técnica oficial. *MySQL* se ha convertido en la opción de base de datos líder para aplicaciones basadas en web. Su modelo más popular es el relacional pero también tiene la posibilidad de emplear un modelo documental. Cuenta con el apoyo de Oracle, empresa que impulsa la innovación de *MySQL* ofreciendo nuevas capacidades para impulsar aplicaciones web, en la nube, móviles y embebidas [26].

A continuación se presentan algunas características del DBMS [25]:

- Compatibilidad con SQL
- Arquitectura cliente/servidor
- Procedimientos almacenados
- Soporte multiplataforma
- Soporte de Unicode
- Consulta de caché
- Soporte SSL

Microsoft SQL Server

DBMS con compatibilidad únicamente para sistemas *Windows* y *Linux*. Sus nuevas versiones le permiten correr de manera local o en la nube, su integración con *Microsoft Azure* ha mejorado mucho su flexibilidad y rendimiento. El modelo de datos más usado para este sistema gestor es el relacional, aunque también permite la implementación de modelos orientados a grafos. Es desarrollado por *Microsoft*, brindándole al usuario un gran respaldo y facilidad a la hora de ofrecer obtener paquetes y soporte [27], [28].

A continuación se presentan algunas características del DBMS [25]:

- Admite una amplia variedad de aplicaciones de procesamiento de transacciones
- SQL está vinculado a Transact-SQL (T-SQL)
- Visualización de datos e informes en dispositivos móviles
- Compatibilidad con nube híbrida
- Escalabilidad y seguridad

PostgreSQL

PostgreSQL es un potente DBMS relacional orientado a objetos de código abierto que hace uso lenguaje SQL extendido, el cual permite el manejo de grandes volúmenes de datos. *PostgreSQL* se ha ganado una sólida reputación por su arquitectura, confiabilidad, integridad de datos, características robustas, extensibilidad y la dedicación de la comunidad de código abierto detrás del software para ofrecer constantemente soluciones, lo que lo hace el DBMS relacional de mayor crecimiento en la actualidad en cuanto a número de usuarios se refiere, durante los últimos tiempos. Puede correr en casi cualquier sistema operativo y puede trabajar con modelos de datos relacionales y documentales [29].

A continuación se presentan algunas características del DBMS [25]:

- Tipos definidos por el usuario
- Herencia de tablas
- Extensibilidad
- Mecanismo de bloqueo sofisticado
- Clave foránea de integridad referencial
- Integridad de datos
- Vistas, reglas, sub-consultas
- Control de concurrencia multi-versión (MVCC)

MongoDB

MongoDB es DBMS de tipo documental que ofrece una gran escalabilidad y flexibilidad, y un modelo de consultas e indexación avanzado. Probablemente sea el DBMS de mayor popularidad en la actualidad, poniéndola en el primer lugar en el ranking de bases de datos NoSQL y en el quinto lugar de las bases de datos más usadas. Tiene la posibilidad de trabajar con datos estructurados y no estructurados, los almacena en documentos BSON flexibles, es decir, cada documento puede contener diferentes campos y las estructuras de datos se pueden ir modificando. Destaca por su gran capacidad de escalado y rendimiento. Las consultas ad-hoc, la

indexación y la agregación en tiempo real permiten acceder a los datos y analizarlos con gran eficacia. Se distribuye bajo licencia de libre uso [30].

A continuación se presentan algunas características del DBMS [25]:

- Gratuito y de código abierto
- Alta disponibilidad
- Escalado horizontal
- Colecciones de tamaño fijo
- Duplicación de datos
- Distribución geográfica
- Balanceo de carga
- Indización y replicación

Redis

Acrónimo de *REmote DIctionary Server*, DBMS compatible con estructuras de datos como cadenas, hashes, listas, conjuntos, conjuntos ordenados con consultas de rango, mapas de bits, índices geoespaciales con consultas de radio y flujos. Tiene propiedades de atomicidad en casos particulares. Con el fin de mejorar su rendimiento trabaja con un conjunto de datos en memoria. Se distribuye bajo licencia de libre uso [31].

A continuación se presentan algunas características del DBMS [25]:

- Increíblemente rápido
- Simplicidad
- Operaciones atómicas
- Admite varios lenguajes de programación
- Replicación maestro/esclavo
- Persistencia instantánea
- Fácil de usar, instalar y mantener

Elasticsearch

Elasticsearch es un software de código abierto utilizado para el análisis de registros, búsqueda de texto completo, inteligencia de seguridad, análisis de negocios y casos de uso de inteligencia operacional. Accesible a través de una API extensa y elaborada, puede impulsar búsquedas extremadamente rápidas que admiten sus aplicaciones de descubrimiento de datos [25], [32].

A continuación se presentan algunas características del DBMS [25]:

- Operaciones de búsqueda casi en tiempo real
- Alto rendimiento
- Facilita la co-ubicación de datos
- Indización de documentos al repositorio
- Almacenamiento de documentos sin formato
- Altamente distribuible y escalable

Cassandra

Es un DBMS del tipo *wide column store*, altamente escalable, distribuida y de alto rendimiento, diseñada para manejar grandes cantidades de datos en muchos servidores de productos básicos, proporcionando una alta disponibilidad sin un punto único de falla. Es un tipo de base de datos NoSQL. Fue creada con el fin de brindar servicio a *Facebook*.

A continuación se presentan algunas características del DBMS [25]:

- Disponibilidad continua
- Rendimiento a escala lineal
- Simplicidad operativa
- Escalabilidad y alta disponibilidad
- Fácil distribución de datos
- Tolerancia a fallas

Neo4j

Es un DBMS de código abierto, con modelo de datos tipo grafo que proporciona un *backend* transaccional compatible con aplicaciones. Su código fuente, se encuentra escrito en *Java* y *Scala*, y está disponible de forma gratuita en *GitHub* o como una descarga de aplicaciones de escritorio fácil de usar. *Neo4j* se distribuye en dos versiones, la *Community Edition* y la versión *Enterprise Edition* que ofrece requisitos empresariales adicionales, como copias de seguridad, clústeres y capacidades de conmutación por error. Este DBMS guarda los datos sin restringirlos a un modelo predefinido, en su lugar, los nodos de un grafo representan las entidades, mientras que las relaciones representan la asociación de estos nodos [25], [33].

A continuación se presentan algunas características del DBMS [25]:

- Modelo de datos flexible
- Altamente escalable
- Información en tiempo real

- Datos conectados y semiestructurados
- Alta disponibilidad y fácil recuperación

3.1.2. CRITERIOS

A continuación se presentan los criterios usados para evaluar los DBMS de acuerdo a las necesidades de la Micro-Red Inteligente UPB, los aspectos generales dentro de la comunidad de desarrolladores y la experiencia y saberes del autor.

Micro-Red Inteligente UPB

Se identifican una serie de necesidades básicas con respecto a características que debe cumplir el DBMS, con el fin de garantizar un correcto acople del DBMS con la visión del proyecto Micro-Red Inteligente UPB y la sinergia con los subsistemas y los datos que estos postean. Los criterios identificados y su definición son:

- **Licenciamiento:** Es importante que el DBMS posea dentro de sus modelos de licenciamiento uno que permita el libremente el uso del DBMS para fines académicos y/o comerciales.
- **Sistema Operativo:** De acuerdo con los lineamientos de TI de la UPB y la experiencia del personal de la Micro-Red Inteligente UPB, es necesario que el DBMS se ejecute sobre sistema operativo Windows
- **Concurrencia:** Es necesario que el DBMS permita manipulación de datos bajo la simultaneidad de usuarios.
- **ACID:** Para garantizar un buen manejo de datos a través de transacciones y consultas es preciso que el DBMS garantice Atomicidad, Consistencia, Aislamiento y Durabilidad.
- **Consistencia:** Aparte de garantizar ACID se hace necesario que el DBMS garantice consistencia para sistemas de computación distribuidos.
- **Esquema libre:** Debido a la gran cantidad de subsistemas, los diferentes protocolos de comunicación, la gran cantidad de fabricantes, se hace necesario que la base de datos sea de libre esquema, ya que no es posible garantizar un esquema único de datos.
- **Consultas:** El DBMS debe garantizar el acceso a los datos mediante métodos o API de consulta.
- **Escalabilidad Horizontal:** Para garantizar la sostenibilidad de la base de datos es importante que el DBMS sea escalable horizontalmente.
- **Almacenamiento en memoria:** Debido a la necesidad de velocidad en las consultas se hace importante poder almacenar datos en memoria con el fin de mejorar el rendimiento del DBMS.

- Lenguajes soportados: Se identificó una serie de lenguajes más usados dentro del personal de la Micro-Red Inteligente UPB y se estableció que el DBMS debe soportar al menos los lenguajes C#, JavaScript, Python y R.
- Tipo de datos: Para garantizar la coherencia en los datos posteados es necesario que el DBMS posea tipos de datos predefinidos.

Comunidad existente

Para la implementación de una base de datos se debe tener en cuenta la experiencia de otras personas en este ámbito y la posibilidad de acceder a información y ayudas de forma fácil y ágil. Seguidamente se presenta una descripción de los criterios empleados para evaluar los DBMS en el ámbito de comunidad de desarrolladores:

- Popularidad: Criterio que establece que tanto renombre tiene el DBMS al rededor del mundo [17].
- Crecimiento: Criterio que determina la pendiente del crecimiento de la polaridad del DBMS en los últimos 6 años [17].
- Soporte y documentación: Es de gran importancia determinar y es fácil acceder a documentación oficial del DBMS y si prestan soporte oficial a los usuarios.
- Colaboración (foros): Este criterio evalúa la posibilidad de encontrar dentro de la comunidad, personas con conocimientos del DBMS específico, dispuestas a brindar apoyo en el desarrollo o implementación de un proyecto. Va muy ligado a la popularidad y crecimiento del DBMS.
- Librerías: Criterio que evalúa la existencia de librerías para interactuar con los DBMS.

Experiencia autor

La experiencia es un concepto que aplica para cada desarrollador en particular, basado en conocimientos previos y el uso de herramientas para determinadas actividades. Los criterios para evaluar la familiaridad del autor con los DBMS son:

- Experiencia: Describe la experiencia general que ha tenido el autor con los diferentes DBMS evaluados.
- Lenguaje de consulta: Evalúa si el autor tiene conocimiento en la estructura y sintaxis del lenguaje o los lenguajes de consulta del DBMS.
- Software de gestión: Valora si el autor conoce y ha trabajado con el software de gestión del DBMS.





- Librerías: Criterio que establece el conocimiento del autor de librerías que permitan la interacción con el DBMS.
- Modelo de datos: Este criterio pretende dar cuenta de la experiencia del autor con el modelo de datos principal del DBMS.
- Documentación: Determina si el autor se encuentra o no familiarizado con la documentación y ayudas del DBMS.
- Afinidad: Este criterio es el más personal de todos, el cual indica el deseo o no del autor por usar un DBMS en el desarrollo del trabajo de grado.

3.1.3. EVALUACIÓN

Para evaluar los DBMS descritos en la sección 3.1.1, se toman cada uno de los criterios explicados en la sección 3.1.2 y se les asigna un peso de acuerdo con lo establecido en la Tabla 12.

Tabla 12. Escala de evaluación DBMS

Fuente: Elaboración propia

Evaluación	Peso	Símbolo
Cumplimiento bueno	3	
Cumplimiento regular	2	
Cumplimiento deficiente	1	
No cumple	0	

En la Figura 11 se puede observar los resultados de la evaluación de acuerdo a los criterios establecidos por las necesidades de la Micro-Red Inteligente UPB. Se logra evidenciar que *MongoDB* sobresale sobre los demás DBMS, con muy buen desempeño para cumplir dichas necesidades.

En la evaluación de los criterios de comunidad se logra apreciar que los DBMS relacionales, basados en lenguaje de consulta SQL, sobresalen sobre los NoSQL debido a su gran trayectoria y madurez. *MongoDB* y *Redis*, siendo NoSQL logran un gran puntaje debido su gran crecimiento en los últimos años. Todo esto se puede apreciar en la Figura 12.

La Figura 13 nos presenta la evaluación de acuerdo a los criterios del autor, donde se puede observar que el autor presenta una gran inclinación hacia SQL Server y *MongoDB*, siendo estos los DBMS más conocidos por él.

Criterios	DBMS								
	Oracle	MySQL	SQL Server	PostgreSQL	MongoDB	Redis	Elasticsearch	Cassandra	Neo4j
¿Es el DBMS de licencia de libre uso?	!	✓	!	✓	✓	✓	✓	✓	✓
¿Puede correr en sistema operativo Windows?	✓	✓	✓	✓	✓	✓	✓	✓	✓
¿Permite simultaneidad de usuario (conurrencia)?	✓	✓	✓	✓	✓	✓	✓	✓	✓
¿El DBMS cumple con propiedades ACID?	✓	✓	✓	✓	✓	✗	●	●	✓
¿Garantiza consistencia en sistemas distribuidos?	✓	✓	✓	✓	✓	!	!	✓	!
¿Es el DBMS de esquema libre?	●	●	●	●	✓	✓	✓	✓	✓
¿Posee métodos de acceso o API de consulta?	✓	✓	✓	✓	✓	✓	✓	✓	✓
¿Soporta escalabilidad horizontal?	●	●	●	●	✓	●	●	✓	✓
¿Permite almacenar datos en memoria?	✓	✓	✓	●	✓	✓	✗	●	●
¿Soporta lenguajes de programación como C#, JavaScript, Python y R?	✓	!	✓	✗	✓	✓	✗	!	✗
¿Posee tipos de datos predefinidos?	✓	✓	✓	✓	✓	!	✓	✓	✓
Subtotal	26	26	26	22	33	26	22	26	27

Figura 11. Evaluación DBMS necesidades Micro-Red Inteligente UPB

Fuente: Elaboración propia

Criterios	DBMS								
	Oracle	MySQL	SQL Server	PostgreSQL	MongoDB	Redis	Elasticsearch	Cassandra	Neo4j
¿El DBMS goza de gran popularidad?	✓	✓	✓	!	!	✗	✗	✗	●
¿Ha crecido su popularidad en los últimos años?	●	✗	●	✓	✓	!	✓	!	!
¿Cuenta con soporte y documentación?	✓	✓	✓	✓	✓	✓	✓	✓	✓
¿Existen foros de ayuda para la solución de inquietudes?	✓	✓	✓	✓	✓	✓	!	✗	✗
¿Existen librerías bajo licencia de software libre?	✓	✓	✓	✓	✓	✓	✗	✗	✗
Subtotal	12	13	12	14	14	12	10	8	7

Figura 12. Evaluación DBMS comunidad existente

Fuente: Elaboración propia

Criterios	DBMS								
	Oracle	MySQL	SQL Server	PostgreSQL	MongoDB	Redis	Elasticsearch	Cassandra	Neo4j
¿Se encuentra familiarizado con el DBMS?	●	●	!	●	!	●	●	●	✗
¿Tiene conocimiento sobre el lenguaje de consulta?	!	!	!	!	●	●	●	●	●
¿Ha trabajado con los software de gestión?	●	●	✓	●	!	●	●	●	●
¿Conoce librerías para interactuar con el DBMS?	●	●	●	●	!	●	●	●	✗
¿Conoce el modelo de datos principal del DBMS?	✓	✓	✓	✓	✓	✗	●	✗	✗
¿Se encuentra familiarizado con la documentación?	●	●	!	●	!	●	●	●	●
¿Le gustaría trabajar con el DBMS?	●	✓	✓	✓	✓	✓	✓	●	●
¿Encuentra el DBMS interesante para el proyecto?	●	✓	!	✓	✓	!	✗	●	✗
Subtotal	5	11	17	11	17	6	4	1	4

Figura 13. Evaluación DBMS experiencia autor

Fuente: Elaboración propia

Finalmente se realiza la suma de los tres subtotales y se obtienen los puntajes para cada DBMS (ver Tabla 13). Como resultado final se obtiene que *MongoDB* es el DBMS más apropiado para llevar a cabo el proyecto, gracias a sus características y familiaridad con el autor.

Tabla 13. Resultados evaluación DBMS

Fuente: Elaboración propia

DBMS	Total
Oracle	43
MySQL	50
SQL Server	55
PostgreSQL	47
MongoDB	64
Redis	44
Elasticsearch	36
Cassandra	35
Neo4j	38

3.2. ESTRUCTURA

Esta sección se hace una introducción a la estructura de datos de *MongoDB* y el análisis del esquema de datos de la base de datos de la Micro-Red Inteligente UPB; para esto se tomó la información resultante del análisis de datos que se desarrolló en el capítulo 2.

MongoDB es un DBMS documental, es decir, permite almacenar estructuras de datos llamadas documentos en formato BSON (*Binary JSON*), conformando una colección a partir de un conjunto de documentos. Al realizar una analogía con DBMS relacionales se tiene que las colecciones son equivalentes a las tablas, los documentos corresponden a un registro o fila y los atributos de un documento coinciden con los campos de un registro. La gran ventaja de *MongoDB* es que permite un esquema libre, es decir, los documentos de una misma colección no necesariamente deben seguir una misma estructura, es posible tener en una misma colección (tabla) dos documentos (registros) con atributos (campos) completamente diferentes. Otra gran ventaja de *MongoDB* es el anidamiento o datos embebidos, permitiendo tener documentos dentro de otros documentos, logrando que dos campos relacionados que en un esquema relacional pertenecen a dos tablas, se puedan almacenar en un único documento [34].

En este orden de ideas se identifican tres estructuras de datos que se pueden implementar: estructura de datos embebida, referencial (análogo a un modelo relacional) y clave-valor. En la Figura 14 se presenta un ejemplo de estructura de datos embebidos, donde la información de contacto y acceso del usuario se encuentran en el mismo documento, en la Figura 15 se presenta

un ejemplo de estructura de datos referencial, donde la información de contacto y acceso del usuario se encuentran en documentos aparte donde se referencia al usuario a través de un identificador. Finalmente en la Figura 16 se presenta un ejemplo de un modelo de datos clave-valor donde únicamente se tienen dos atributos, un identificador (clave) y la información de un usuario (valor) que en este caso es un subdocumento.



Figura 14. Ejemplo estructura embebida

Fuente: Adaptado de [34]

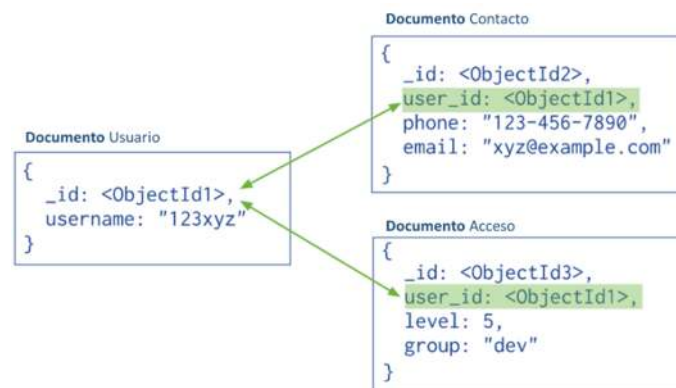


Figura 15. Ejemplo estructura referencial

Fuente: Adaptado de [34]



Figura 16. Ejemplo estructura clave-valor

Fuente: Adaptado de [34]

Para la base de datos de la Micro-Red Inteligente UPB se determina que la estructura que se ajusta mejor al tipo de datos es la estructura de datos clave-valor, donde cada colección corresponde a una variable de un subsistema y cada documento de dicha colección contiene la estampa de tiempo (clave) y el dato (valor) correspondiente a dicha estampa para esa variable determinada. Para nombrar cada colección se implementó la siguiente estructura, la cual corresponde al nombre del subsistema separado del nombre de la variable por un guion bajo:

<subsistema> | <variable>

Cada documento contiene un identificador creado por *MongoDB*, una estampa de tiempo y el valor correspondiente a dicha estampa de tiempo. La estructura de cada documento sigue el siguiente esquema de datos:

```
{
  "_id": <ObjectID>,
  "timestamp": <Int64>,
  "value": <Double>
}
```

3.3. VOLUMEN DE DATOS

En esta sección se calcula el volumen de datos que serán posteados en la base de datos de la Micro-Red Inteligente UPB, para esto se toma la información resultante del análisis de datos realizado en el capítulo 2. En la Tabla 14 se presenta un resumen de lo obtenido en dicho capítulo.

Tabla 14. Resumen de datos por subsistema

Fuente: Elaboración propia

Subsistema	Dispositivos	VARIABLES	Frecuencia [datos/min]	Horas efectivas [h/día]
Estación meteorológica	1	8	1	24
Medidores de radiación	3	2	1	24
SFV [5 kW _p]	1	24	0,2	24
SFV [12.5 kW _p] EV - EP	2	10	1	14
SFV [10 kW _p] EP x2	2	10	1	14
SFV [28 kW _p] 11B	1	15	1	24
SFV [26 kW _p] 11C x2	2	10	1	14
Casa Hábitat	1	22	1	24
Luminarias Iluplus	5	4	1	24
Medidores inteligentes	16	46	0,2	24
VARIABLES calculadas	1	21	1	24

Se puede determinar el número total de colecciones obteniendo el número total de variables, se multiplica el número de dispositivos por el número de variables para cada subsistema y se obtiene un total de **912** variables, equivalente al número de colecciones de la base de datos. Para obtener el número de documentos por año de cada subsistema se usa la ecuación (3).

$$\text{Documentos} = D * V * f_p * h_e * 60 \text{ min/h} * 365 \text{ días/año} \quad (3)$$

Donde:

D : Dispositivos

V : Variables

f_p : Frecuencia de posteo [datos/min]

h_e : Horas efectivas [h/día]

En la Tabla 15 se observan los documentos por año que serán posteados para cada subsistema y el espacio que estos ocupan en disco teniendo en cuenta que cada documento pesa 56 B (*ObjectId* pesa 12 B, *Int64* pesa 8 B y *Double* pesa 8 B, al estar en formato *BSON* estos valores se duplican). Como resultado se obtiene que en un año la base de datos tendrá **146.642.400** documentos nuevos que equivalen a **7.6 GB** de almacenamiento por año.

Tabla 15. Documentos por año por subsistema

Fuente: Elaboración propia

Subsistema	Documentos [datos/año]	Almacenamiento [MB/año]
Estación meteorológica	4.204.800	225
Medidores de radiación	3.153.600	168
SFV [5 kW _p]	2.522.880	135
SFV [12.5 kW _p] EV - EP	6.132.000	327
SFV [10 kW _p] EP x2	6.132.000	327
SFV [28 kW _p] 11B	7.884.000	421
SFV [26 kW _p] 11C x2	6.132.000	327
Casa Hábitat	11.563.200	618
Luminarias Iluplus	10.512.000	561
Medidores inteligentes	77.368.320	4.132
Variables calculadas	11.037.600	589

Contemplando dos escenarios de crecimiento anual en el número de documentos, crecimiento del 10% y crecimiento del 50%, se puede estimar que para el año 2030 la base de datos alcanzara un tamaño cercano a 160 GB y a 1.9 TB con un crecimiento del 10% y del 50% respectivamente; esto implica que no se requieren grandes cantidades de recursos de almacenamiento para albergar la base de datos y no se hace necesario en un futuro cercano la implementación de un

script que elimine de forma sistemática los datos más antiguos de la base de datos. En la Tabla 16 se presenta el almacenamiento requerido por año y acumulado de la base de datos para el periodo 2019-2030 para ambos escenarios contemplados.

Tabla 16. Almacenamiento requerido 2019-2030

Fuente: Elaboración propia

Año	Almacenamiento (10%) [GB]	Almacenamiento acumulado (10%) [GB]	Almacenamiento (50%) [GB]	Almacenamiento acumulado (50%) [GB]
2019	7,65	7,65	7,65	7,65
2020	8,41	16,06	11,47	19,12
2021	9,25	25,31	17,21	36,33
2022	10,18	35,49	25,81	62,14
2023	11,20	46,69	38,72	100,86
2024	12,32	59,01	58,08	158,93
2025	13,55	72,56	87,12	246,05
2026	14,90	87,46	130,67	376,72
2027	16,39	103,86	196,01	572,73
2028	18,03	121,89	294,01	866,75
2029	19,84	141,73	441,02	1.307,77
2030	21,82	163,55	661,53	1.969,30

3.4. INSTALACIÓN *MONGODB*

Para el presente trabajo se instaló una instancia de *MongoDB* versión 4.0, la cual corresponde a la versión vigente para junio de 2019, fecha en la que se desarrolló el presente trabajo de grado.

A continuación, se presentan los pasos que se deben seguir para la instalación de una instancia de *MongoDB* 4.0 como servicio de Windows:

3.4.1. DESCARGA *MONGODB* 4.0 COMMUNITY EDITION

Se realiza la descarga del instalador (.msi) de *MongoDB 4.0 Community Edition* a través de la URL <https://www.mongodb.com/download-center/community> donde se selecciona la versión que se desea descargar, el sistema operativo donde se va instalar y el paquete que se desea descargar y posterior a esto se da clic en el botón descargar; para el instalador de *MongoDB 4.0 Community Edition* para Windows .msi se puede descargar directamente a través de la URL https://fastdl.mongodb.org/win32/mongodb-win32-x86_64-2008plus-ssl-4.0.10-signed.msi.

3.4.2. EJECUCIÓN DEL INSTALADOR DESCARGADO

Se va a la ruta donde se guardó el instalador y se da doble clic, abriendo el asistente de instalación (ver Figura 17), en la primera ventana del asistente se hace clic en *Next*.

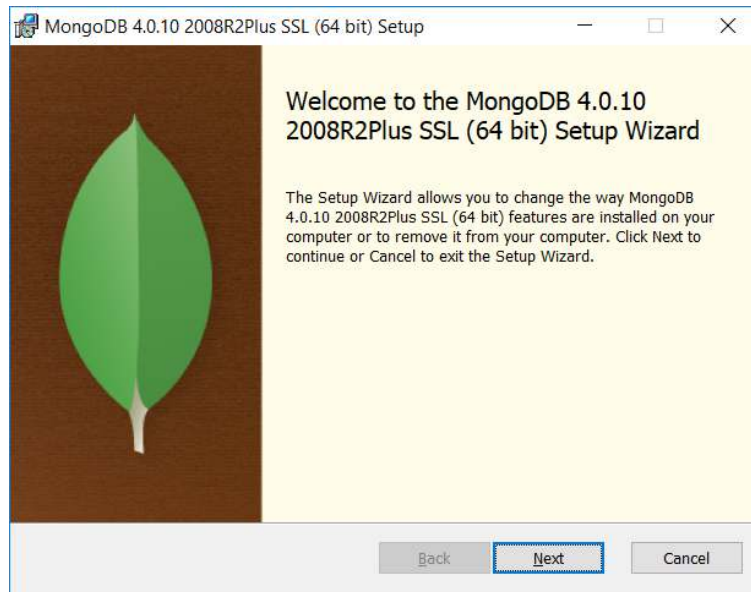


Figura 17. Asistente de instalación MongoDB

Fuente: Elaboración propia

En la siguiente ventana del asistente (ver Figura 18) se deben aceptar los términos de licencia de MongoDB y se da clic en *Next*.

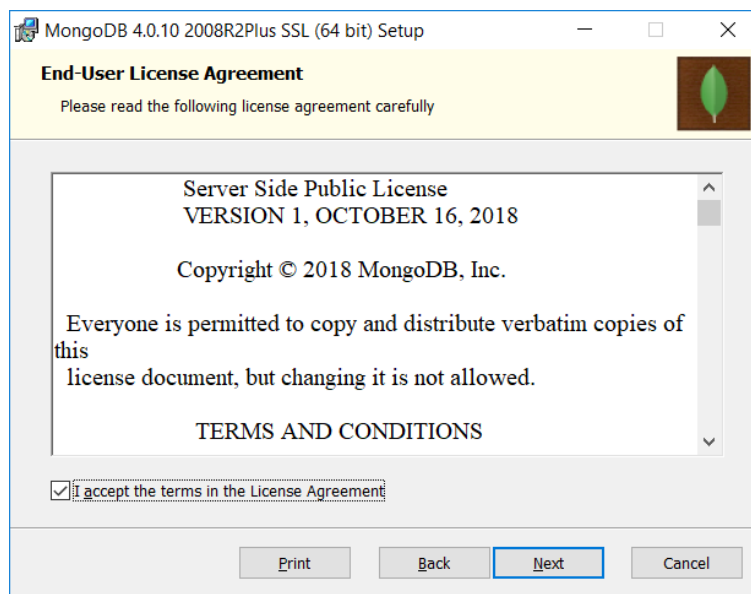


Figura 18. Licencia MongoDB

Fuente: Elaboración propia

Con el fin de instalar todos los paquetes que vienen con el instalador, se recomienda seleccionar versión completa de instalación en la ventana siguiente del asistente (ver Figura 19) y dar clic en *Next*.

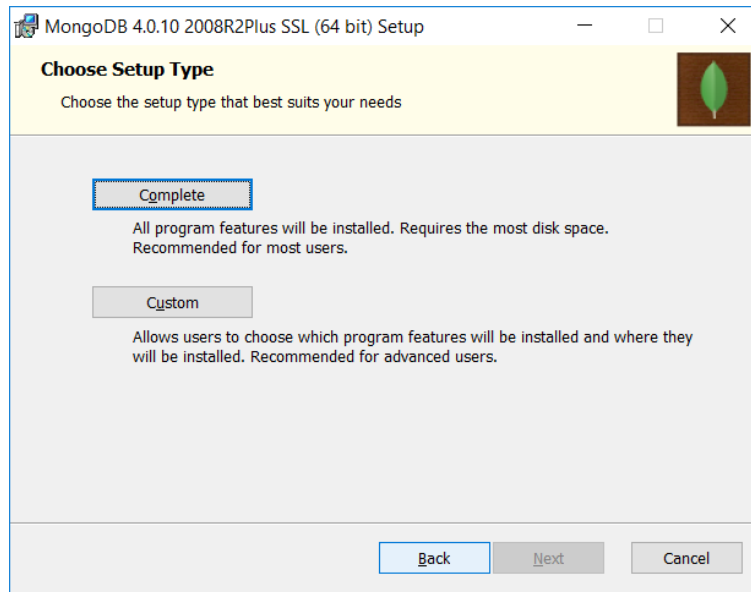


Figura 19. Tipo de instalación MongoDB

Fuente: Elaboración propia

En la siguiente ventana del asistente (ver Figura 20) se recomienda seleccionar la opción que permite instalar *MongoDB Compass*, software que permite la administración de la base de datos y hacer clic en *Next*.

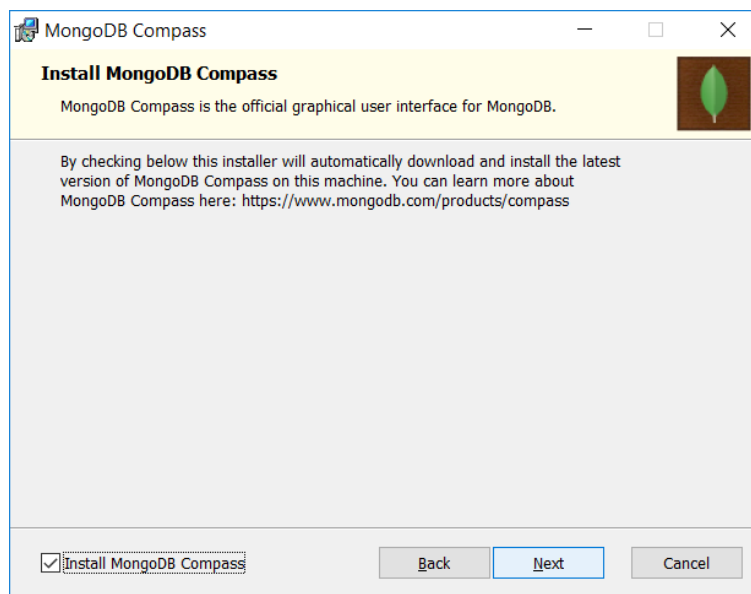


Figura 20. Instalación MongoDB Compass

Fuente: Elaboración propia

Para instalar *MongoDB* como servicio de *Windows* en la ventana siguiente del asistente se debe seleccionar la configuración que se muestra en la Figura 21, el campo de nombre puede ser modificado.

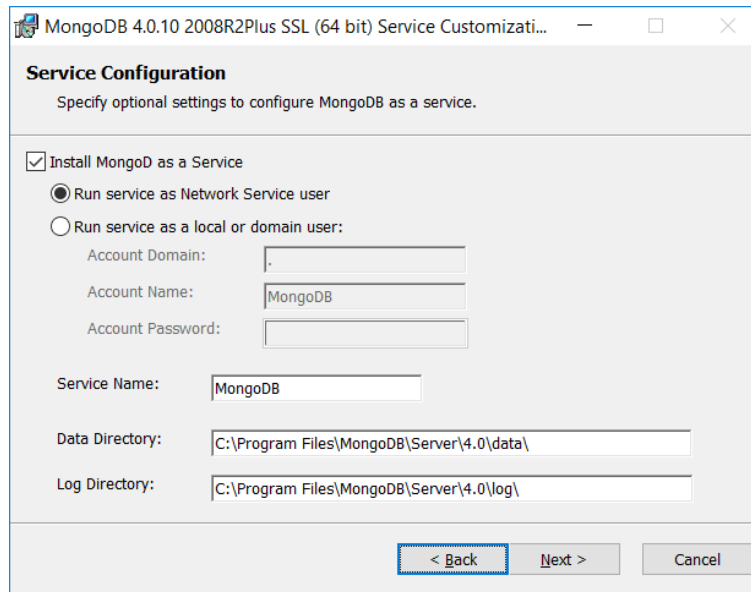


Figura 21. Servicio de Windows MongoDB

Fuente: Elaboración propia

Finalmente, en la ventana siguiente del asistente (ver Figura 22) se presiona clic en el botón *Install* para comenzar la instalación, es posible que el sistema operativo pida permisos de administrador para continuar con la instalación.

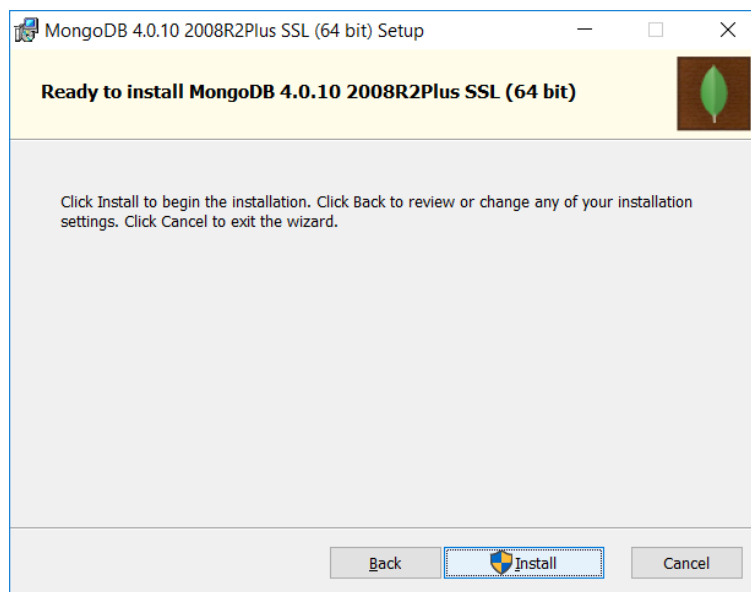


Figura 22. Comienzo instalación MogoDB

Fuente: Elaboración propia

Al finalizar el proceso de instalación se abre una ventana en el asistente (ver Figura 23) donde comunica la correcta instalación de *MongoDB*. Se hace clic en el botón *Finish* y se da por terminada la instalación

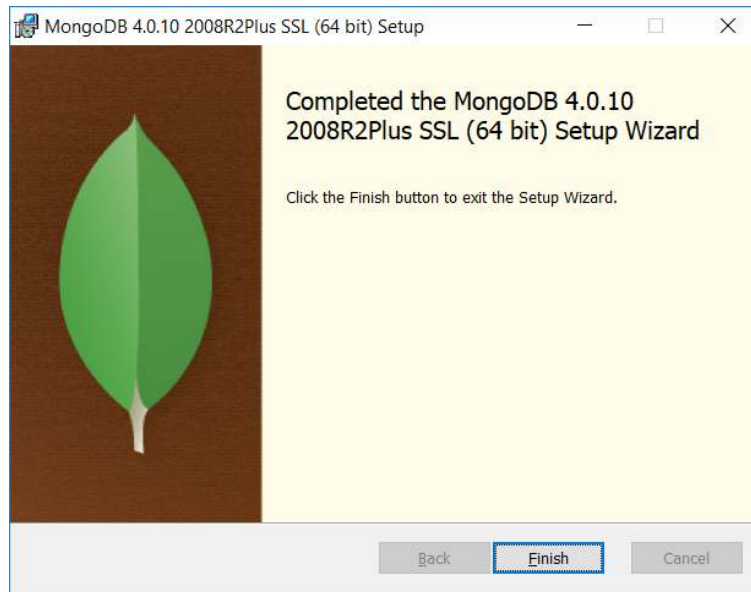


Figura 23. Finalización instalación instancia MongoDB

Fuente: Elaboración propia

3.5. MIGRACIÓN UBIDOTS

En esta sección se expone la forma en que se realizó la migración de los datos existentes de la Micro-Red Inteligente UPB desde *Ubidots* a la base de datos de *MongoDB*. Para esto se realizó un script en *Python*, lenguaje de programación con variables dinámicas, el cual cuenta con una sintaxis simplificada y rápida que permite una gran legibilidad, además cuenta con una gran comunidad de desarrolladores y una enorme cantidad de librerías que facilitan el desarrollo.

Para el desarrollo del script se usaron varias librerías, las cuales permitieron realizar ciertas tareas con mayor rapidez y facilidad, a continuación se pueden apreciar las librerías usadas:

- Time: Librería que permite el manejo de tareas relacionadas con el tiempo
- Requests: Librería que permite realizar peticiones HTTP.
- PyMongo: Es la librería recomendada por *MongoDB* para trabajar en Python conexiones y consultas a bases de datos *MongoDB*.

El script se divide en cuatro partes principales, la primera corresponde a la conexión con *MongoDB*, la segunda parte permite a través de una petición *GET* obtener las variables que se encuentran almacenadas en *Ubidots*, la tercera parte permite a través de una petición *GET* obtener los valores almacenados en cada variable de *Ubidots* y por último se encuentra el módulo que permite la inserción de los valores en la base de datos de *MongoDB*.

En la Figura 24 se presenta un diagrama del funcionamiento del script, donde se puede apreciar que este realiza consultas HTTP a la API de *Ubidots*, esta responde en formato JSON, el script procesa la información y la almacena en *MongoDB*; a continuación se explicara con mayor detalle cada uno de los módulos del script.

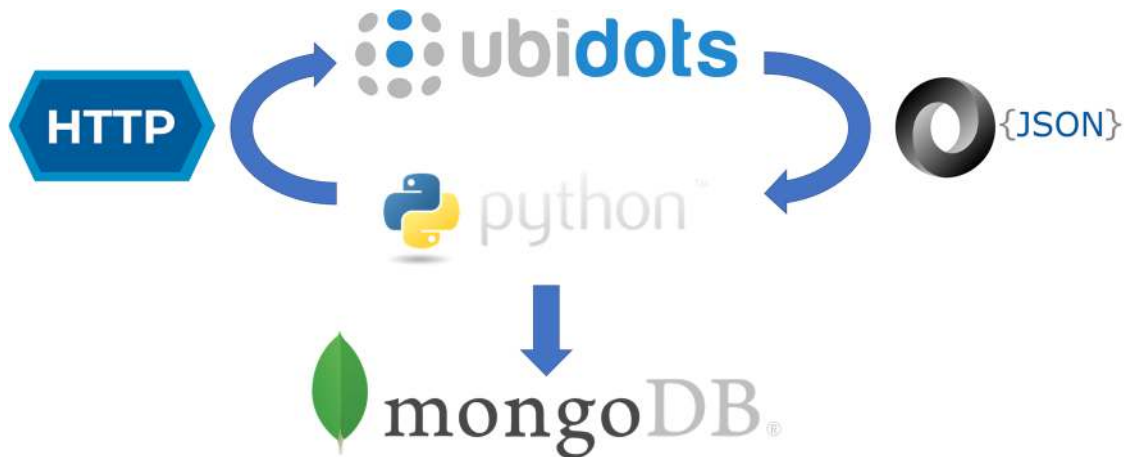


Figura 24. Funcionamiento script de migración

Fuente: Elaboración propia

3.5.1. CONEXIÓN MONGODB

Este módulo permite la conexión al servidor donde quedara albergada la base datos, como también permite asignarle un nombre a la nueva base de datos de la Micro-Red Inteligente UPB, a la cual se le da el nombre de *uRedDB*. Es necesario que al servidor al que se conecte tenga una instancia de *MongoDB* y esta se encuentre corriendo como servicio de *Windows* como se explicó en la sección 3.4.

3.5.2. GET VARIABLES

Para obtener las diferentes variables almacenadas en *Ubidots* se realiza una petición GET a través de la API de *Ubidots* por medio del siguiente *endpoint*:

<https://industrial.api.ubidots.com/api/v1.6/variables>

Este *endpoint* se acompañó con los siguientes parámetros para completar la consulta:

- *token*: Clave que se obtiene a través de la interfaz de *Ubidots* y permite el acceso a los datos
- *page_size*: Determina la cantidad máxima de variables que son recibidas. Se estableció en un valor de 600, ya que se estima que el número de variables no supera este valor

A partir de esta consulta se obtienen 521 variables, cada una con la información que se presenta en la Figura 25. Esta información se filtra y se obtiene para cada variable su identificador (id), su nombre (*name*) y el nombre del subsistema al que pertenece (*datasource/name*).

```
{
  id: "593855cd7625427d549414dc",
  name: "currentL2",
  icon: "",
  unit: null,
  label: "currentL2",
  - datasource: {
    id: "593851ce7625427d49d95159",
    name: "13_10ARQ",
    url: "https://industrial.api.ubidots.com/api/v1.6/datasources/593851ce7625427d49d95159",
    label: "13_10arq"
  },
  url: "https://industrial.api.ubidots.com/api/v1.6/variables/593855cd7625427d549414dc",
  description: null,
  properties: { },
  - tags: [
    "mongo"
  ],
  created_at: "2017-06-07T19:36:45.939000Z",
  - last_value: {
    value: 38.8,
    timestamp: 1559662481768,
    context: { },
    created_at: 1559662482668,
    id: "5cf68f9273efc315bb2fa71b"
  },
  last_activity: 1559662482000,
  type: 0,
  derived_expr: "",
  values_url: "https://industrial.api.ubidots.com/api/v1.6/variables/593855cd7625427d549414dc/values"
},
```

Figura 25. Estructura variables Ubidots

Fuente: Elaboración propia

3.5.3. GET VALORES

Posterior a la obtención de las diferentes variables, se realizan una petición GET para cada una de las variables obteniendo los valores que han sido guardados en *Ubidots*. Esta petición se realiza a través de la API de *Ubidots* por medio del siguiente *endpoint*:

<https://industrial.api.ubidots.com/api/v1.6/variables/<variableID>/values>

Donde *<variableID>* corresponde al identificador de cada una de las variables. Este *endpoint* se acompañó con los siguientes parámetros para completar la consulta:

- *token*: Clave que se obtiene a través de la interfaz de *Ubidots* y permite el acceso a los datos
- *page_size*: Determina la cantidad máxima de variables que son recibidas. Se estableció en un valor de 10.000.000, ya que se estima que el número de valores por variable no supera este valor
- *end*: Determina el *timestamp* final de consulta, es decir, la consulta traerá valores entre 0 y este valor. Se estableció en 1560704400000, correspondiente al 16 de junio del 2019 a las 12 del día

A partir de esta consulta se obtienen todos los valores de cada una de las 521 variables, cada uno con la información que se presenta en la Figura 26. Esta información se filtra y se obtiene para cada valor su estampa de tiempo (*timestamp*) y su valor (*value*).

```
{  
  timestamp: 1559662481768,  
  value: 38.8,  
  context: { },  
  created_at: 1559662482853  
},
```

Figura 26. Estructura valor Ubidots

Fuente: Elaboración propia

3.5.4. INSERCIÓN MONGODB

Se organiza la información obtenida, acoplándola a la estructura definida en el numeral 3.2 donde el nombre de cada colección sigue la siguiente estructura, modificando los nombres de acuerdo a una obtenidos en el numeral 3.5.2 por unos preestablecidos:

<datasource/name> | <name>

Cada documento se ajusta a la siguiente estructura de acuerdo con los datos obtenidos en el numeral 3.5.3:

```
{  
  "timestamp": <timestamp>,  
  "value": <value>  
}
```

Para agregar estos valores en la base de datos se hace uso de la función *insert_many*, la cual permite agregar varios documentos a una misma colección en la base de datos.

En el Anexo C se presentan las 521 colecciones que fueron creadas en la base datos y el número de documentos que se insertaron a cada colección.

4. INTERFAZ DE PROGRAMACIÓN DE APLICACIONES (API)

Una API es un conjunto de funciones, procesos o métodos de un software que permite la interacción con funcionalidades de otro programa. En el presente trabajo de grado la API corresponde a una aplicación web que permite la interacción con la base de datos a través de peticiones HTTP, permitiendo realizar 4 operaciones en la base de datos:

- Obtener datos (petición GET)
- Insertar datos (petición POST)
- Editar datos (petición PUT)
- Eliminar datos (petición DELETE)

La API se desarrolló de forma local en un equipo de la Micro-Red Inteligente UPB donde se encuentra albergada la base de datos migrada en el numeral 3.5. En las secciones siguientes se explica la estructura de las operaciones permitidas a través de la API con la base de datos.

Para acceder a la API es necesario determinar una URL o *endpoint* por medio de la cual se puede acceder a los diferentes métodos desarrollados:

<http://<host>/variables>

Donde <host> corresponde al servidor y a el puerto donde se encuentra ejecutando la API o su DNS correspondiente.

En la Figura 27 se presenta un diagrama del funcionamiento de la API, donde se evidencia el flujo de información al realizar las diferentes consultas posibles.

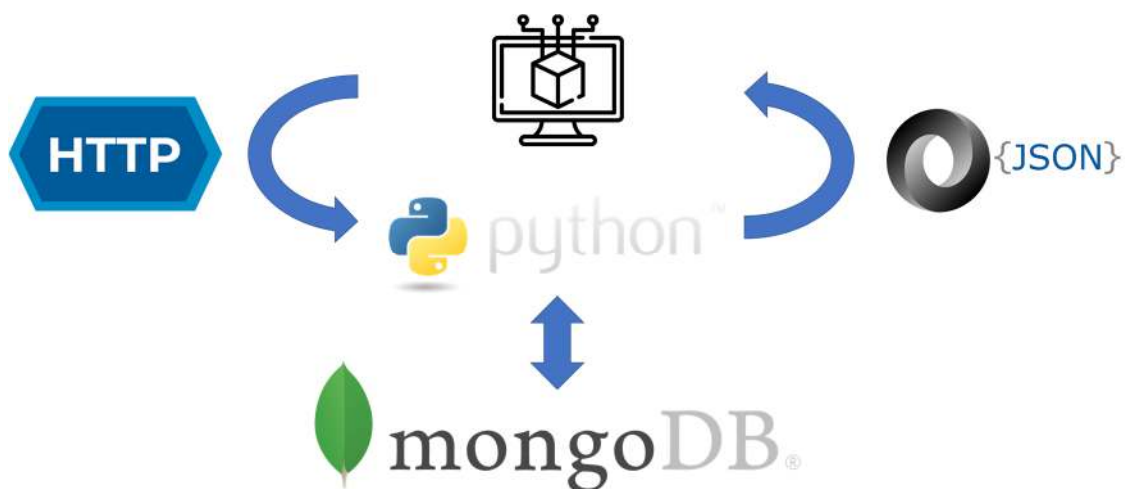


Figura 27. Asistente de instalación MongoDB

Fuente: Elaboración propia

4.1. PETICIONES GET

Las peticiones GET permiten obtener información de la base de datos, respondiendo en formato JSON. Fueron desarrollados cuatro métodos GET, los cuales de acuerdo al *endpoint* y los parámetros enviados en la petición responden con información determinada de la base de datos.

4.1.1. GET SUBSISTEMAS

Esta petición permite obtener la información de todos los subsistemas de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/variables/subsystems>

La respuesta de esta petición sigue la siguiente estructura:

```
{
  "count": <Int64>,
  "result": [
    {
      "subsystem": <string>,
      "subsystem_url": <string>
    }
  ]
}
```

Donde:

- *count*: Número total de subsistemas que se encuentran en la base de datos
- *results*: Vector donde cada posición es un objeto que representa un subsistema, los cuales cuentan con:
 - ✓ *subsystem*: Nombre del subsistema
 - ✓ *subsystem_url*: URL que permite acceder a la información individual de cada subsistema

Para acceder a la información individual de un subsistema específico es necesario agregar al *endpoint* anterior un parámetro *name* que corresponde al nombre del subsistema del que se desea información individual:

<http://<host>/variables/subsystemas?name=<nombreSubsistema>>

La respuesta de esta petición sigue la siguiente estructura:

```
{
  "subsystem": < string >,
  "subsystem_url": <string>,
  "variables": [
  ]
}
```

Donde:

- *subsystem*: Es el nombre del subsistema
- *subsystem_url*: URL usada
- *variables*: Vector donde cada posición es una variable del subsistema

4.1.2. GET VARIABLES

Esta petición permite obtener la información de todas las variables de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/variables>

La respuesta de esta petición sigue la siguiente estructura:

```
{
  "count": <Int64>,
  "result": [
    {
      "subsystem": <string>,
      "subsystem_url": <string>,
      "values_url": <string>,
      "variable": <string>,
      "variable_url": <string>
    }
  ]
}
```

Donde:

- *count*: Número total de variables que se encuentran en la base de datos,
- *results*: Vector donde cada posición es un objeto que representa una variable, las cuales cuentan con:
 - ✓ *subsystem*: Nombre del subsistema al que pertenece la variable
 - ✓ *subsystem_url*: URL que permite acceder a la información individual del subsistema al que pertenece la variable
 - ✓ *values_url*: URL por medio de la cual se pueden obtener los valores de la variable
 - ✓ *variable*: Nombre de la variable
 - ✓ *variable_url*: URL que permite acceder a la información individual de cada variable

Para acceder a la información individual de una variable específica es necesario agregar al *endpoint* anterior un parámetro *subsystem*, que corresponde al nombre del subsistema de la variable de la que se desea información, y un parámetro *variable* que indica el nombre de la variable de la cual se desea información individual:

<http://<host>/variables?subsystem=<nombreSubsistema>&variable=<nombreVariable>>

La respuesta de esta petición sigue la siguiente estructura:

```
{
  "last_value": {
    "timestamp": <Int64>,
    "value": <double>
  },
  "subsystem": <string>,
  "subsystem_url": <string>,
  "values_url": <string>,
  "variable": <string>,
  "variable_url": <string>
}
```

Donde:

- *last_value*: Objeto que representa el último valor posteo de la variable, el cual cuenta con:
 - ✓ *timestamp*: Estampa de tiempo del último valor
 - ✓ *value*: Valor correspondiente a dicha estampa de tiempo
- *subsystem*: Nombre del subsistema al que pertenece la variable
- *subsystem_url*: URL del subsistema al que pertenece la variable
- *values_url*: URL que permite acceder a los valores de la variable
- *variable*: Nombre de la variable
- *variable_url*: URL usada

4.1.3. GET COLECCIONES

Esta petición permite obtener la información de todas las colecciones de la base de datos de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/collections>

La respuesta de esta petición sigue la siguiente estructura:

```
{
  "count": <Int64>,
  "result": [
    {
      "collection": <string>,
      "collection_url": <string>,
      "documents": <string>
    }
  ]
}
```

Donde:

- *count*: Número total de colecciones que se encuentran en la base de datos
- *results*: Vector donde cada posición es un objeto que representa una colección, las cuales cuentan con:
 - ✓ *collection*: Nombre de la colección
 - ✓ *collection_url*: URL que permite acceder a la información individual de cada colección
 - ✓ *documents*: Número total de documentos que tiene la colección

Para acceder a la información individual de una colección específica es necesario agregar al *endpoint* anterior un parámetro *name* que corresponde al nombre de la colección de la que se desea información individual:

<http://<host>/collections?name=<nombreColeccion>>

La respuesta de esta petición sigue la siguiente estructura:

```
{
  "collection": <string>,
  "collection_url" : <string>,
  "documents": <string>
  "last_value": {
    "timestamp": <Int64>,
    "value": <double>,
  }
}
```

Donde:

- *collection*: Nombre de la colección
- *collection_url*: URL usada
- *documents*: Número total de documentos que tiene la colección
- *last_value*: Objeto que representa el último valor posteo de la colección, el cual cuenta con:
 - ✓ *timestamp*: Estampa de tiempo del último valor y

- ✓ *value*: Valor correspondiente a dicha estampa de tiempo

4.1.4. GET VALORES

Esta petición permite obtener los valores o documentos pertenecientes a una variable específica de un subsistema base de datos de la Micro-Red Inteligente UPB usando el siguiente *endpoint*

<http://<host>/values>

Es obligatorio incluir en la petición los parámetros *subsystem*, que corresponde al nombre del subsistema *variable* que indica el nombre la variable del subsistema del que se requieren los valores. Adicionalmente es opcional incluir el parámetro *size* que indica el número máximo de valores p documentos que devuelve la petición, por defecto se encuentra en 200, devolviendo los últimos 200 valores de la variable. Los parámetros *start* y *end*, también opcionales, deben ser estampas de tiempo en formato EPOCH en milisegundos; estos parámetros permiten consultar valores en un rango de tiempo, *start* es el valor más antiguo que es requiere, por defecto se encuentra en cero que corresponde al 1 de enero de 1970, y *end* es el valor más reciente que es requerido, por defecto se encuentra en 3187209600000 que corresponde al 31 de diciembre de 2070.

[http://<host>/values?subsystem=<nombreSubsitema>&variable=<nombreVariable>&size=<nu
meroDocumentos>&start=<timestampInicial>&end=<timestampFinal>](http://<host>/values?subsystem=<nombreSubsitema>&variable=<nombreVariable>&size=<numeroDocumentos>&start=<timestampInicial>&end=<timestampFinal>)

La respuesta de esta petición sigue la siguiente estructura:

```
{
  "count": <Int64>,
  "result": [
    {
      "timestamp": <Int64>,
      "value": <double>
    }
  ]
}
```

Donde:

- *count*: Número total de valores o documentos que devuelve la petición, puede coincidir con el parámetro *size*
- *results*: Vector donde cada posición es un objeto que representa un valor o documento, los cuales cuentan con:
 - ✓ *timestamp*: Estampa de tiempo correspondiente al valor
 - ✓ *value*: Dato que corresponde a la variable en dicha estampa de tiempo

4.2. PETICIONES POST

Las peticiones POST permiten agregar documentos (valores) a colecciones existentes o a nuevas colecciones (si la colección no existe, es creada). Se desarrolló un único método POST que permite crear o agregar valores de acuerdo al cuerpo enviado en la petición. El *endpoint* usado para las peticiones agregar documentos a la base de datos de la Micro-Red Inteligente UPB es:

<http://<host>/values>

El *body* o cuerpo de la petición es un vector de objetos que sigue la siguiente estructura:

```
[
  {
    "subsystem": <string>,
    "variable": <string>,
    "timestamp": <Int64>,
    "value": <double>
  }
]
```

Donde:

- *subsystem*: Nombre del subsistema al que pertenece el valor, si el subsistema no existe es creado
- *variable*: Variable a la que pertenece el valor, si la variable no existe es creada
- *timestamp*: Estampa de tiempo correspondiente al valor
- *value*: dato que corresponde a la variable en dicha estampa de tiempo

El vector del *body* puede tener el número de posiciones que se desee, cada posición debe tener la misma estructura.

4.3. PETICIONES PUT

Las peticiones PUT permiten editar elementos existentes de la base de datos de la Micro-Red Inteligente UPB. Fueron desarrollados cuatro métodos PUT, los cuales de acuerdo al *endpoint* y el *body* enviado en la petición, editan elementos particulares de la base de datos.

4.3.1. PUT SUBSISTEMAS

Esta petición permite editar el nombre de un subsistema de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/subsystems>

El *body* o cuerpo de la petición es un vector de objetos que sigue la siguiente estructura:

```
[  
  {  
    "oldName": <string>,  
    "newName": <string>  
  }  
]
```

Donde:

- *oldName*: Nombre del subsistema que desea editar
- *newName*: Nombre que le desea dar al subsistema

El vector del *body* puede tener el número de posiciones que se desee, cada posición debe tener la misma estructura.

4.3.2. PUT VARIABLES

Esta petición permite editar el nombre de una variable perteneciente a un subsistema de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/variables>

El *body* o cuerpo de la petición es un vector de objetos que sigue la siguiente estructura:

```
[
  {
    "subsystem": <string>,
    "oldName": <string>,
    "newName": <string>
  }
]
```

Donde:

- *subsystem*: Nombre del subsistema al que pertenece la variable que desea editar
- *oldName*: Nombre de la variable que desea editar
- *newName*: Nuevo nombre que le desea dar a la variable

El vector del *body* puede tener el número de posiciones que se desee, cada posición debe tener la misma estructura.

4.3.3. PUT COLECCIONES

Esta petición permite editar el nombre de una colección de la base de datos de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/collections>

El *body* o cuerpo de la petición es un vector de objetos que sigue la siguiente estructura:

```
[
  {
    "oldName": <string>,
    "newName": <string>
  }
]
```

Donde:

- *oldName*: Nombre de la colección que desea editar
- *newName*: Nuevo nombre que le desea dar a la colección

El vector del *body* puede tener el número de posiciones que se desee, cada posición debe tener la misma estructura.

4.3.4. PUT VALORES

Esta petición permite editar los valores de un documento de la base de datos de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/values>

El *body* o cuerpo de la petición es un vector de objetos que sigue la siguiente estructura:

```
[
  {
    "collection": <string>,
    "timestamp": <Int64>,
    "newValue": <double>
  }
]
```

Donde:

- *collection*: Nombre de la colección al que pertenece el valor que desea editar
- *timestamp*: Estampa de tiempo del valor que desea editar
- *newValue*: Valor que desea dar al documento

El vector del *body* puede tener el número de posiciones que se desee, cada posición debe tener la misma estructura.

4.4. PETICIONES DELETE

Las peticiones DELETE permiten eliminar elementos existentes de la base de datos de la Micro-Red Inteligente UPB. Fueron desarrollados cuatro métodos DELETE, los cuales de acuerdo al *endpoint* y el *body* enviado en la petición, elimina elementos particulares de la base de datos.

4.4.1. DELETE SUBSISTEMAS

Esta petición permite eliminar subsistemas de la base de datos de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/subsystems>

El *body* o cuerpo de la petición es un vector de objetos que sigue la siguiente estructura:

```
[
  {
    "subsystem": <string>
  }
]
```

Donde:

- *Subsystem*: Nombre del subsistema que desea eliminar

El vector del *body* puede tener el número de posiciones que se desee, cada posición debe tener la misma estructura.

4.4.2. DELETE VARIABLES

Esta petición permite eliminar variables pertenecientes a un subsistema de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/variables>

El *body* o cuerpo de la petición es un vector de objetos que sigue la siguiente estructura:

```
[
  {
    "subsystem": <string>,
    "variable": <string>
  }
]
```

Donde:

- *Subsystem*: Nombre del subsistema al que pertenece la variable que desea eliminar
- *Variable*: Nombre de la variable que desea eliminar

El vector del *body* puede tener el número de posiciones que se desee, cada posición debe tener la misma estructura.

4.4.3. DELETE COLECCIONES

Esta petición permite eliminar colecciones de la base de datos de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/collections>

El *body* o cuerpo de la petición es un vector de objetos que sigue la siguiente estructura:

```
[
  {
    "collection": <string>
  }
]
```

Donde:

- *collection*: Nombre de la colección que desea eliminar

El vector del *body* puede tener el número de posiciones que se desee, cada posición debe tener la misma estructura.

4.4.4. DELETE VALORES

Esta petición permite eliminar valores los valores de un documento de la base de datos de la Micro-Red Inteligente UPB usando el siguiente *endpoint*:

<http://<host>/values>

El *body* o cuerpo de la petición es un vector de objetos que sigue la siguiente estructura:

```
[
  {
    "allCollections": <boolean>,
    "collection": <string>,
    "start": <Int64>,
    "end": <Int64>,
  }
]
```

Donde:

- *allCollection*: Booleano que indica si se desea eliminar valores de todas las colecciones o no (1 elimina valores de todas colecciones, 0 elimina valores de una única colección)
- *collection*: Nombre de la colección de la cual se desea eliminar valores, este atributo únicamente es válido si *allCollections* es 0
- *start* y *end*: Son estampas de tiempo en formato EPOCH en milisegundos; *start* es el valor más antiguo que se desea eliminar y *end* es el valor más reciente que se desea eliminar

El vector del *body* puede tener el número de posiciones que se desee, cada posición debe tener la misma estructura.

5. PROCESAMIENTO

Este capítulo presenta los desarrollos realizados para procesar los datos obtenidos desde *Ubidots* y almacenados en la base de datos *uRedDB* de *MongoDB*. En primer lugar, se desarrolló un script que permite rellenar (imputar) los datos perdidos o vacíos, usando un método general para todas las variables de la base de datos. En segundo lugar, se desarrolló una script que calcular variables de interés a partir de variables medidas. Ambos desarrollos son códigos que se ejecutan en servidor y no deben ser manipulados por usuarios ajenos a la Micro-Red Inteligente UPB.

5.1. IMPUTACIÓN

La ausencia de datos corresponde a un problema de calidad de datos donde campos obligatorios de la base de datos no fueron registrados, en el caso de la Micro-Red Inteligente UPB, se han presentado diferentes situaciones que han impedido el registro de medidas y posterior almacenamiento en *Ubidots*, llevando a grandes vacíos en las series de tiempo de cada variable.

Los datos faltantes pueden derivar en problemas a la hora de obtener estadísticos y cuando se desea realizar una revisión de una variable en particular. Existen 3 opciones la hora de trabajar con bases de datos donde se registran valores faltantes, la primera es trabajar únicamente con los datos completos, la segunda es trabajar con los datos disponibles y la tercera es la reconstrucción de datos a partir de técnicas de imputación.

La imputación implica estimar el valor faltante, y para esto se puede usar tres metodologías:

- Uso de un estadístico de la variable
- Uso de modelo matemático que describe la variable
- Interpolación entre valores más cercanos

Para la base de datos de *uRedDB* se usa un método de imputación basado en la mediana de los valores existentes en el mismo tipo de día y a la misma hora, esta agrupación se realiza debido a los grandes vacíos que se encuentran en la base de datos y que impiden realizar agrupaciones más granulares. Para esto se realizó un script en *Python* que hace uso de las siguientes librerías:

- *DateTime*: Librería que permite el manejo de tareas relacionadas con el tiempo y *timestamp*
- *PyMongo*: Es la librería recomendada por *MongoDB* para trabajar en Python conexiones y consultas a bases de datos *MongoDB*.

- NumPy: Librería que permite el manejo de datos como matrices, incluyendo funciones matemáticas y estadísticas
- Pandas: Librería que permite el manejo de datos como matrices (*DataFrame*) es una extensión de NumPy

El script se divide en cinco partes principales. La primera parte corresponde a la conexión con *MongoDB*, la cual se realiza de acuerdo a lo expuesto en el numeral 3.5.1, también se obtienen los documentos de cada variable, usando peticiones GET explicadas en el numeral 4.1.4. La segunda parte convierte los documentos obtenidos en matrices (*DataFrame*) y se re-muestra con el fin de obtener datos cada minuto, todo esto mediante la librería Pandas, permitiendo agregar las columnas de tipo de día (1-31) y hora (0-23). En la tercera se hayan las medianas para cada combinación de tipos de día y hora, es decir, se obtiene una matriz de 31x24. La cuarta parte es el módulo que permite imputar los valores faltantes, en el cual se buscan los datos faltantes y se reemplazan con los valores correspondientes obtenidos. Por último, se realiza la inserción de los valores calculados en la base de datos *uRedDB* siguiendo lo descrito en la sección 3.5.4.

Se realiza un caso de prueba de la variable de potencia del subsistema SFV [10KWP] 1 para los datos del año 2017, donde únicamente se tienen datos del mes de octubre y del mes de noviembre con algunos datos faltantes como se observa en la Figura 1, al aplicar la imputación con la mediana por tipo de día y hora se logran generar los datos faltantes y se complementa el mes de noviembre y el mes de diciembre como se observa en la Figura 29. Al comparar ambas gráficas, se aprecia que el método empleado se comporta muy bien cuando los datos faltantes no superan un día (octubre y primero días de noviembre), sin embargo, cuando es necesario imputar días o meses completos el método presenta un comportamiento aceptable, pudiendo mejor al incluir más dimensiones a la clasificación, es decir, adicionar al tipo de día y hora variables como mes o incluso variables medidas que influyan en la medida de interés.

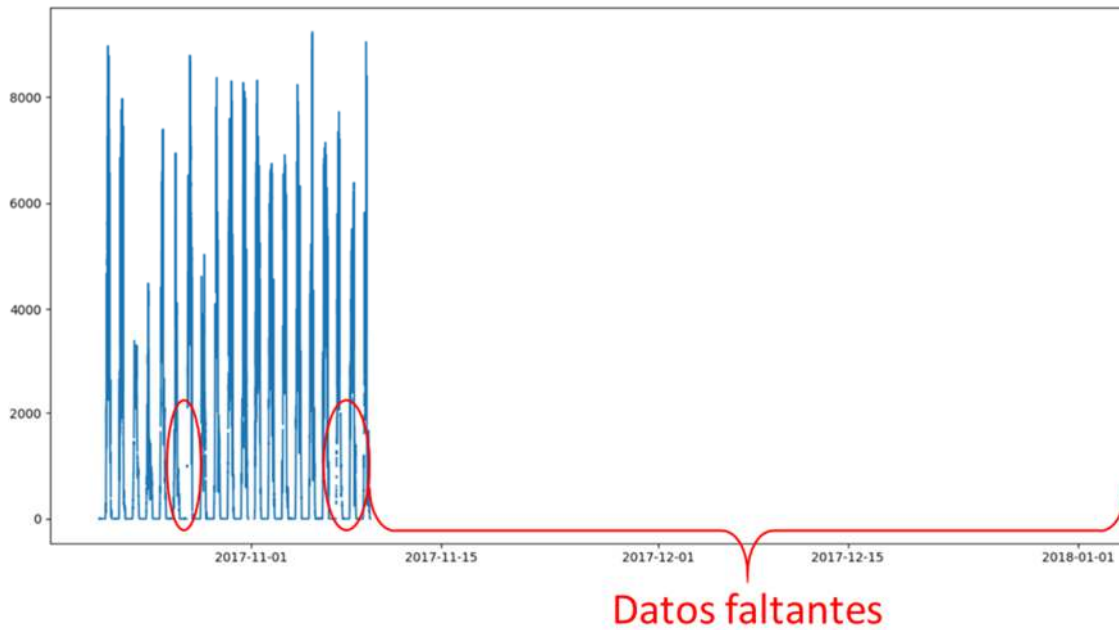


Figura 28. Datos SFV [10KWP] 1 | Potencia

Fuente: Elaboración propia

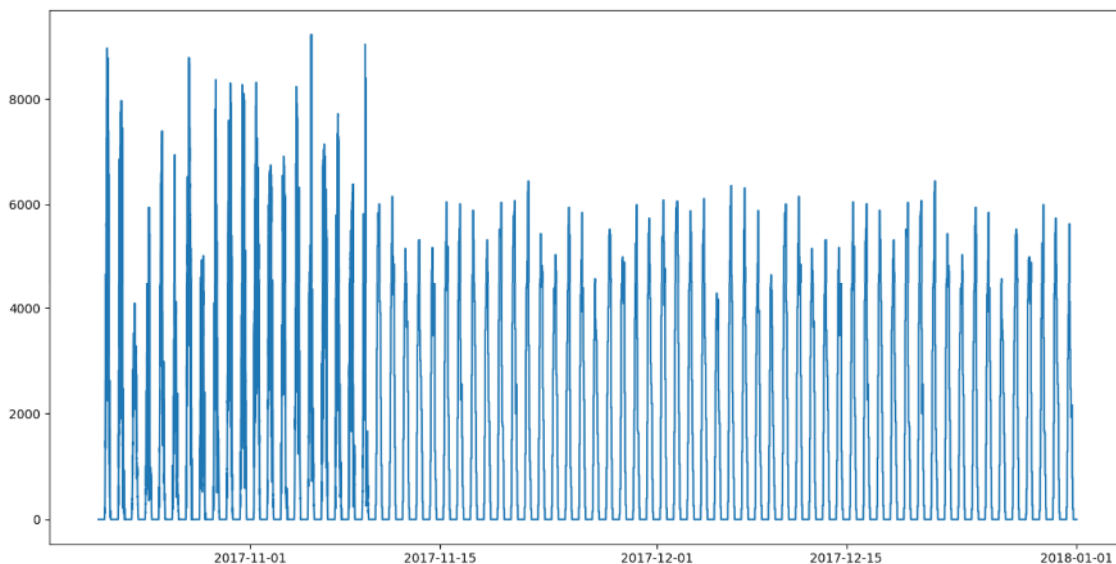


Figura 29. Imputación SFV [10KWP] 1 | Potencia

Fuente: Elaboración propia

5.2. VARIABLES CALCULADAS

Una vez son imputadas las variables se procede al cálculo de variables de interés a partir de variables medidas. Gracias al re-muestreo realizado previamente a la imputación todas las

variables se ajustan a una hora exacta y poseen la misma frecuencia de muestreo durante todo el tiempo de existencia de la variable, lo que permite realizar cálculos con mayor facilidad, únicamente es necesario identificar las variables medidas necesarias para realizar el cálculo y realizar la operación de acuerdo a cada *timestamp*.

A continuación se listan las variables de interés calculadas, a medida que van entrando subsistemas es posible incluir nuevas variables:

- Eficiencia subsistema SFV [5 kW_p]
- Eficiencia subsistema SFV SFV [12.5kW_p] EV - EP
- Eficiencia subsistema SFV [10 kW_p] EP x2
- Eficiencia subsistema SFV [28 kW_p] 11B
- Eficiencia subsistema SFV [26 kW_p] 11C x2

Para obtener estas variables se desarrolló un script en *Python* que hace uso de las siguientes librerías:

- *DateTime*: Librería que permite el manejo de tareas relacionadas con el tiempo y *timestamp*
- *PyMongo*: Es la librería recomendada por *MongoDB* para trabajar en Python conexiones y consultas a bases de datos *MongoDB*
- *Pandas*: Librería que permite el manejo de datos como matrices (*DataFrame*) es una extensión de *NumPy*

El script se divide en cuatro partes principales. La primera parte corresponde a la conexión con *MongoDB*, la cual se realiza de acuerdo a lo expuesto en el numeral 3.5.1, también se obtienen los documentos de cada variable necesaria para realizar el cálculo, usando peticiones GET explicadas en el numeral 4.1.4. La segunda realiza los cálculos de la variable de interés. Finalmente se realiza la inserción de los valores calculados en la base de datos *uRedDB* siguiendo lo descrito en la sección 3.5.4.

6. CONCLUSIONES

Se logra desarrollar un conjunto de cuatro aplicaciones que permiten el procesamiento, almacenamiento y consulta de datos de las variables a monitorear de la Micro-Red inteligente ubicada en el campus de laureles de la Universidad Pontificia Bolivariana – Sede Medellín.

La evaluación de los diferentes DBMS se hace vital ya que permite enfocar los diferentes desarrollos en un motor de base de datos que se ajusta a las necesidades de Micro-Red inteligente UPB y al desarrollador, se selecciona un DBMS NoSQL, lo que le da una mayor versatilidad, disponibilidad y un rendimiento más óptimo a la base de datos.

Se logra establecer que de acuerdo con el tamaño estimado de la base de datos y al contemplar dos escenarios de crecimiento, no se requieren grandes recursos de almacenamiento y no es necesario pensar en la eliminación de información en un futuro cercano, información que puede ser importante para analizar el crecimiento del proyecto Micro-Red inteligente UPB. Es importante resaltar que los cálculos se realizan en base a las variables y dispositivos que se tienen a octubre de 2019 y aunque se tienen dos escenarios de crecimiento, se recomienda rehacer este cálculo si se prevé un crecimiento de la base de datos mayor al 50%.

Debido a la versatilidad que presenta *MongoDB* a la hora de establecer una estructura de datos, se evaluaron varias posibilidades y finalmente se optó por una estructura que se acomoda a los datos existentes y que garantiza una agilidad y velocidad a la hora de consulta y almacenamiento de datos.

El desarrollo de la API permite a los diferentes programas o aplicaciones de la Micro-Red inteligente UPB interactuar con la base de datos, permitiendo a los usuarios visualizar esta información en las aplicaciones diseñadas para tal fin.

Debido a las grandes cantidades de datos perdidos en las diferentes variables de los subsistemas de la Micro-Red inteligente UPB, se hace difícil implementar métodos de imputación óptimos, ya que en ciertos casos son más los datos faltantes que los datos completos, sin embargo, se logra desarrollar una técnica que permite la imputación de datos con comportamiento aceptable.

La aplicación que permite obtener variables calculadas a partir de variables medidas es una aplicación en construcción, la cual debe ir siendo modificada a medida que vayan entrando nuevos subsistemas y que vayan apareciendo nuevas variables de interés.

Las variables calculadas permiten tener una vista más amplia a los diferentes subsistemas de la Micro-Red Inteligente UPB, como también permite mayor agilidad en la consulta de datos y elaboración de informes de gestión de la Micro-Red.

El presente trabajo de grado es una guía, con la cual cualquier desarrollador de la Micro-Red Inteligente UPB puede interactuar con el fin de configurar y ejecutar las diferentes aplicaciones desarrolladas.

La implementación de la base de datos de la Micro-Red Inteligente UPB brinda a esta la posibilidad de gestionar su propia base de datos local, la cual le permite una infinidad de desarrollos nuevos y oportunidades de mejora. Con esto, también se mitigan los inconvenientes de disponibilidad que tenía el anterior repositorio y se potencia las aplicaciones existentes al brindar una mayor velocidad de respuesta.

7. TRABAJO FUTURO

A partir del trabajo desarrollado se logran identificar puntos en los cuales se puede dar continuidad a las aplicaciones, con el fin de obtener mejores productos que finalmente transmiten mayor seguridad y confiabilidad a los usuarios de la Micro-Red Inteligente UPB. A continuación se exponen los puntos identificados

- Asignación de usuarios y roles dentro de la base de datos
- Implementación de autenticación para uso de la API
- Realizar análisis específico para cada variable de la Micro-Red Inteligente UPB y evaluar la posibilidad de mejorar la imputación de datos faltantes usando métodos multi-variables o incluyendo más dimensiones al método de clasificación
- Implementar métodos de predicción meteorológicos que permitan guardar pronósticos en la base de datos con el fin de estimar generación dentro del campus
- Realizar un estudio de las variables indirectas que se pueden calcular a partir de las variables medidas con el fin de complementar la aplicación de variables calculadas
- Implementación de esquema de alta disponibilidad, con el fin de garantizar la disponibilidad de la base de datos durante el mayor tiempo posible

BIBLIOGRAFIA

- [1] J. W. González Sanchez, G. J. López Jiménez, I. A. Isaac, and H. Cardona Agudelo, "Anteproyecto Micro-Red UPB." p. 71, 2013.
- [2] Grupo de Investigación en Transmisión y Distribución de Energía Eléctrica (TyD) and Universidad Pontificia Bolivariana, "Micro-Red Inteligente UPB," 2018. [Online]. Available: <https://microred.upb.edu.co/>. [Accessed: 07-Mar-2019].
- [3] G. J. López Jiménez, I. A. Isaac, J. W. González Sanchez, and H. Cardona Agudelo, "Integración de energías renovables (solar fotovoltaica) en campus upb laureles-micro red inteligente," *Investig. Apl.*, vol. 8, no. 2, pp. 152–159, 2014.
- [4] U.S. Department of Energy (DOE), "SmartGrid.gov: What is the Smart Grid?," 2012. [Online]. Available: https://www.smartgrid.gov/the_smart_grid/. [Accessed: 09-Mar-2019].
- [5] Cigré, *Microgrids 1 Engineering, Economics, & Experience*. 2015.
- [6] Soluciones Energía, "Soluciones Energía | Micro redes eléctricas | Guadalajara." [Online]. Available: <http://solucionesenergia.com/>. [Accessed: 09-Mar-2019].
- [7] A. Hossain *et al.*, "Evolution of Microgrids with Converter-Interfaced Generations: Challenges and Opportunities," *Int. J. Electr. Power Energy Syst.*, vol. 109, no. October, pp. 1–49, Jul. 2018.
- [8] V. A. Gómez, C. Hernández, and E. Rivas, "Visión General, Características y Funcionalidades de la Red Eléctrica Inteligente (Smart Grid)," *Inf. tecnológica*, vol. 29, no. 2, pp. 89–102, 2018.
- [9] IEEE Standards Coordinating Committee 21, *IEEE Std 2030™-2011, IEEE Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications, and Loads*. New York, 2011.
- [10] F. Leccese, "An overview on IEEE Std 2030," *2012 11th Int. Conf. Environ. Electr. Eng. IEEEIC 2012 - Conf. Proc.*, pp. 340–345, 2012.
- [11] Flaticon, "Free vector icons - SVG, PSD, PNG, EPS & Icon Font - Thousands of free icons." [Online]. Available: <https://www.flaticon.com/>. [Accessed: 18-Mar-2019].
- [12] IEEE, "Towards Definition Internet of Things Revision1," *Toward. a Defin. Internet Things*, pp. 1–86, 2015.

- [13] A. Kumar, "No SQL DB," 2015.
- [14] R. Camps Paré, L. A. Casillas Santillán, D. Costal Costa, M. Gibert Ginestà, C. Martín Escofet, and O. Pérez Mora, "Bases de datos," *Fund. per a la Univ. Oberta Catalunya*, 2005.
- [15] A. Silberschatz, H. F. Korth, and S. Sudarshan, *Fundamentos de bases de datos*. 2002.
- [16] M. E. Millán, *Fundamentos de Bases de Datos - Notas de referencia*. 2017.
- [17] db-engines, "DB-Engines Ranking - popularity ranking of database management systems." [Online]. Available: <https://db-engines.com/en/ranking>. [Accessed: 20-Mar-2019].
- [18] PANDORAFMS, "Bases de datos NoSQL: Guía definitiva con las diferencias, ventajas y desventajas de cada una." [Online]. Available: <https://blog.pandorafms.org/es/bases-de-datos-nosql/>. [Accessed: 15-Mar-2019].
- [19] Medium, "SQL vs NoSQL; Ventajas y Desventajas – Marlon Manzo – Medium." [Online]. Available: <https://medium.com/@marlonmanzo/sql-vs-nosql-ventajas-y-desventajas-849ccc9db3d4>. [Accessed: 20-Mar-2019].
- [20] A. Castro Romero, M. Callejas Cuervo, and J. S. Gonzales Sanabria, "Utilidad y funcionamiento de las bases de datos NoSQL," *Rev. Fac. Ing. UPTC*, vol. 21, no. 33, pp. 21–32, 2012.
- [21] C. Carvajal, "Reinventando la gestión de datos | MongoDB – Desarrollo y Automatización," 2017. [Online]. Available: <https://automatizacion977.wordpress.com/2017/09/15/reinventando-la-gestion-de-datos-mongodb/>. [Accessed: 09-Mar-2019].
- [22] "Scopus." [Online]. Available: <https://www.scopus.com>.
- [23] "Database - Oracle." [Online]. Available: <https://www.oracle.com/database/>. [Accessed: 26-May-2019].
- [24] "What is Oracle Database (Oracle DB)? - Definition from Techopedia." [Online]. Available: <https://www.techopedia.com/definition/8711/oracle-database>. [Accessed: 26-May-2019].
- [25] "21 Gestores de Bases de Datos más Usados 2019." [Online]. Available: <https://www.diarlu.com/gestores-bases-datos/>. [Accessed: 30-May-2019].

- [26] “MySQL :: About MySQL.” [Online]. Available: <https://www.mysql.com/about/>. [Accessed: 26-May-2019].
- [27] “Plataforma de datos de Microsoft | Microsoft.” [Online]. Available: <https://www.microsoft.com/es-mx/sql-server/>. [Accessed: 26-May-2019].
- [28] “Tipos de bases de datos y las mejores bases de datos del 2016.” [Online]. Available: <https://blog.pandorafms.org/es/tipos-de-bases-de-datos-y-las-mejores-bases-de-datos-del-2016/>. [Accessed: 26-May-2019].
- [29] “PostgreSQL: About.” [Online]. Available: <https://www.postgresql.org/about/>. [Accessed: 26-May-2019].
- [30] “¿Qué es MongoDB? | MongoDB.” [Online]. Available: <https://www.mongodb.com/what-is-mongodb?lang=es-es>. [Accessed: 30-May-2019].
- [31] “Introduction to Redis – Redis.” [Online]. Available: <https://redis.io/topics/introduction>. [Accessed: 30-May-2019].
- [32] “Elasticsearch: RESTful, Distributed Search & Analytics | Elastic.” [Online]. Available: <https://www.elastic.co/es/products/elasticsearch>. [Accessed: 30-May-2019].
- [33] “What Is a Graph Database and Property Graph | Neo4j.” [Online]. Available: <https://neo4j.com/developer/graph-database/>. [Accessed: 30-May-2019].
- [34] “Data Modeling Introduction — MongoDB Manual.” [Online]. Available: <https://docs.mongodb.com/manual/core/data-modeling-introduction/>. [Accessed: 09-Jun-2019].

ANEXO A. APLICACIONES

Característica	Aplicación			
	UbidotsToMongo.py	uRedDBImputation.py	uRedDBCalculatedVariables.py	uRedDBAPI.py
Tipo	Script	Aplicación	Aplicación	Aplicación
Lenguaje de programación	Python 3.7.2	Python 3.7.2	Python 3.7.2	Python 3.7.2
Librerías necesarias	import time import requests import pymongo	import datetime import pandas import pymongo import numpy	import datetime import pandas import pymongo	from flask import Flask from flask import jsonify from flask import request from flask_pymongo import PyMongo
Ejecución	Doble clic	Doble clic	Doble clic	Doble clic
Orden de ejecución	1	2	3	4
Frecuencia de ejecución	Una única ejecución	Cada 24 horas	Continua	Continua
Plataforma de ejecución	Servidor	Servidor	Servidor	Servidor
Configurable	Si	Si	Si	Si
Variables configurables	instance db ubidotsURL parameters	instance db	instance db	app.config url
Versión	1.0	1.0	1.0	1.0
Manual de usuario	Sección 3.5	Sección 5.1	Sección 5.2	Capítulo 4

ANEXO B. CONFIGURACIÓN APLICACIONES

UBIDOTSTOMONGO.PY

- *instance*: Instancia de *MongoDB* donde serán migrados los datos *Ubidots*
- *db*: Nombre de la base de datos de *MongoDB* que será creada
- *ubidotsURL*: Es el *endpoint* de *Ubidots* (ver documentación de *Ubidots*)
- *parameters*: son los parámetros que acompañan las diferentes consultas a *Ubidots* (ver documentación de *Ubidots*)

UREDDBIMPUTATION.PY

- *instance*: Instancia de *MongoDB* a la que se concretará para obtener los datos
- *db*: Nombre de la base de datos de *MongoDB* a la que se desea conectar

UREDDBCALCULATEDVARIABLES.PY

- *instance*: Instancia de *MongoDB* a la que se concretará para obtener los datos
- *db*: Nombre de la base de datos de *MongoDB* a la que se desea conectar

UREDDBAPI.PY

- *app.config*: Es el *endpoint* de la base de datos de *MongoDB* a la cual accederá la API
- *url*: Es el *endpoint* a través del cual los usuarios podrán realizar peticiones a la API

ANEXO C. COLECCIONES Y DOCUMENTOS INSERTADOS

Colección	Documentos
FR1_B18_10 EnergyDay	243.831
FR1_B18_10 EnergyTotal	268.491
FR1_B18_10 Frequency	253.324
FR1_B18_10 IAC	275.201
FR1_B18_10 IDC	278.816
FR1_B18_10 PAC	294.082
FR1_B18_10 VAC	260.839
FR1_B18_10 VDC	346.457
FR1_B18_12.5 EnergyDay	131.972
FR1_B18_12.5 EnergyTotal	131.971
FR1_B18_12.5 Frequency	131.971
FR1_B18_12.5 IAC	131.971
FR1_B18_12.5 IDC	131.974
FR1_B18_12.5 PAC	131.972
FR1_B18_12.5 VAC	131.972
FR1_B18_12.5 VDC	131.972
FR2_B18_10 EnergyDay	252.271
FR2_B18_10 EnergyTotal	264.894
FR2_B18_10 Frequency	278.503
FR2_B18_10 IAC	249.138
FR2_B18_10 IDC	250.884
FR2_B18_10 PAC	254.908
FR2_B18_10 VAC	237.019
FR2_B18_10 VDC	261.180
FR2_B18_12.5 EnergyDay	132.073

Colección	Documentos
FR2_B18_12.5 EnergyTotal	132.074
FR2_B18_12.5 Frequency	132.074
FR2_B18_12.5 IAC	132.074
FR2_B18_12.5 IDC	132.074
FR2_B18_12.5 PAC	132.074
FR2_B18_12.5 VAC	132.074
FR2_B18_12.5 VDC	132.074
MI_B10_5 ET_I1	50.891
MI_B10_5 ET_I10	38.817
MI_B10_5 ET_I11	37.339
MI_B10_5 ET_I12	35.871
MI_B10_5 ET_I13	35.546
MI_B10_5 ET_I14	31.484
MI_B10_5 ET_I15	57.096
MI_B10_5 ET_I16	33.425
MI_B10_5 ET_I17	29.722
MI_B10_5 ET_I18	30.738
MI_B10_5 ET_I19	40.639
MI_B10_5 ET_I2	37.847
MI_B10_5 ET_I20	37.420
MI_B10_5 ET_I3	52.631
MI_B10_5 ET_I4	32.107
MI_B10_5 ET_I5	39.867
MI_B10_5 ET_I6	30.775
MI_B10_5 ET_I7	32.349

Colección	Documentos
MI_B10_5 ET_I8	31.420
MI_B10_5 ET_I9	28.869
MI_B10_5 EnergyDay	18.092
MI_B10_5 EnergyTotal	17.949
MI_B10_5 InstantaneousEnergy	18.092
MI_B10_5 PAC	18.092
MI_B10_5 P_I1	38.728
MI_B10_5 P_I10	31.657
MI_B10_5 P_I11	52.058
MI_B10_5 P_I12	40.056
MI_B10_5 P_I13	35.319
MI_B10_5 P_I14	35.918
MI_B10_5 P_I15	31.400
MI_B10_5 P_I16	79.179
MI_B10_5 P_I17	30.603
MI_B10_5 P_I18	39.258
MI_B10_5 P_I19	54.389
MI_B10_5 P_I2	35.387
MI_B10_5 P_I20	38.367
MI_B10_5 P_I3	36.762
MI_B10_5 P_I4	32.455
MI_B10_5 P_I5	42.708
MI_B10_5 P_I6	29.825
MI_B10_5 P_I7	41.336
MI_B10_5 P_I8	32.355
MI_B10_5 P_I9	47.298
SL_B11_28 ConsumptionEnergyDay	210.274

Colección	Documentos
SL_B11_28 ConsumptionEnergyTotal	241.001
SL_B11_28 ConsumptionPower	208.723
SL_B11_28 EnergyDay	235.320
SL_B11_28 EnergyTotal	225.875
SL_B11_28 PAC	213.559
SL_B11_28 PDC	225.057
SL_B11_28 Pocc	226.889
SL_B11_28 Psur	221.870
SL_B11_28 VAC	204.993
SL_B11_28 VDC	232.220
SL_B6 E1_EnergyTotal	12.594
SL_B6 E1_I	12.594
SL_B6 E1_Status	12.595
SL_B6 E1_V	12.594
SL_B6 E2_EnergyTotal	12.866
SL_B6 E2_I	12.867
SL_B6 E2_Status	12.867
SL_B6 E2_V	12.867
SL_B6 E3_EnergyTotal	12.721
SL_B6 E3_I	12.721
SL_B6 E3_Status	12.722
SL_B6 E3_V	12.722
SL_B6 E4_EnergyTotal	11.300
SL_B6 E4_I	11.300
SL_B6 E4_Status	11.300
SL_B6 E4_V	11.300
SL_B6 M0_EnergyTotal	11.209

Colección	Documentos
SL_B6 MO_I	11.210
SL_B6 MO_Status	11.210
SL_B6 MO_V	11.210
SM_B10_ARQ ActiveEnergyExport	182.228
SM_B10_ARQ ActiveEnergyExportDay	234.217
SM_B10_ARQ ActiveEnergyImport	193.411
SM_B10_ARQ ActiveEnergyImportDay	219.271
SM_B10_ARQ ActivePower	226.596
SM_B10_ARQ ActivePower1	216.451
SM_B10_ARQ Frequency	198.094
SM_B10_ARQ I1	211.852
SM_B10_ARQ I2	194.500
SM_B10_ARQ I3	220.526
SM_B10_ARQ In	215.443
SM_B10_ARQ ReactiveEnergyExport	203.190
SM_B10_ARQ ReactiveEnergyExportDay	212.904
SM_B10_ARQ ReactiveEnergyImport	252.328
SM_B10_ARQ ReactiveEnergyImportDay	238.831
SM_B10_ARQ ReactivePower	212.050
SM_B10_ARQ ReactivePower1	191.837
SM_B10_ARQ RelativeTHDVoltage	206.069
SM_B10_ARQ TotalPowerFactor	237.267
SM_B10_ARQ V1	257.308
SM_B10_ARQ V1Angle	221.592
SM_B10_ARQ V2	203.907
SM_B10_ARQ V2Angle	259.061
SM_B10_ARQ V3	211.702

Colección	Documentos
SM_B10_ARQ V3Angle	238.727
SM_B12_DERE ActiveEnergyExport	219.108
SM_B12_DERE ActiveEnergyExportDay	223.506
SM_B12_DERE ActiveEnergyImport	241.032
SM_B12_DERE ActiveEnergyImportDay	197.504
SM_B12_DERE ActivePower	238.974
SM_B12_DERE ActivePower1	200.667
SM_B12_DERE Frequency	202.120
SM_B12_DERE I1	189.124
SM_B12_DERE I2	223.237
SM_B12_DERE I3	247.140
SM_B12_DERE In	264.365
SM_B12_DERE ReactiveEnergyExport	261.705
SM_B12_DERE ReactiveEnergyExportDay	264.570
SM_B12_DERE ReactiveEnergyImport	243.479
SM_B12_DERE ReactiveEnergyImportDay	239.247
SM_B12_DERE ReactivePower	220.390
SM_B12_DERE ReactivePower1	217.643
SM_B12_DERE RelativeTHDVoltage	221.429
SM_B12_DERE TotalPowerFactor	276.885
SM_B12_DERE V1	217.850
SM_B12_DERE V1Angle	261.271
SM_B12_DERE V2	213.234
SM_B12_DERE V2Angle	191.875
SM_B12_DERE V3	205.952
SM_B12_DERE V3Angle	210.533
SM_B15_BIBL ActiveEnergyExport	223.358

Colección	Documentos
SM_B15_BIBL ActiveEnergyExportDay	236.712
SM_B15_BIBL ActiveEnergyImport	193.538
SM_B15_BIBL ActiveEnergyImportDay	218.016
SM_B15_BIBL ActivePower	205.799
SM_B15_BIBL ActivePower1	191.265
SM_B15_BIBL Frequency	202.236
SM_B15_BIBL I1	222.230
SM_B15_BIBL I2	209.538
SM_B15_BIBL I3	262.839
SM_B15_BIBL In	215.727
SM_B15_BIBL ReactiveEnergyExport	204.179
SM_B15_BIBL ReactiveEnergyExportDay	181.563
SM_B15_BIBL ReactiveEnergyImport	262.001
SM_B15_BIBL ReactiveEnergyImportDay	199.635
SM_B15_BIBL ReactivePower	186.663
SM_B15_BIBL ReactivePower1	199.080
SM_B15_BIBL RelativeTHDVoltage	267.146
SM_B15_BIBL TotalPowerFactor	213.257
SM_B15_BIBL V1	203.619
SM_B15_BIBL V1Angle	207.999
SM_B15_BIBL V2	189.822
SM_B15_BIBL V2Angle	214.095
SM_B15_BIBL V3	221.488
SM_B15_BIBL V3Angle	223.516
SM_B17_POLI ActiveEnergyExport	217.864
SM_B17_POLI ActiveEnergyExportDay	222.594
SM_B17_POLI ActiveEnergyImport	214.188

Colección	Documentos
SM_B17_POLI ActiveEnergyImportDay	206.582
SM_B17_POLI ActivePower	216.809
SM_B17_POLI ActivePower1	232.946
SM_B17_POLI Frequency	195.276
SM_B17_POLI I1	192.048
SM_B17_POLI I2	252.848
SM_B17_POLI I3	195.059
SM_B17_POLI In	216.629
SM_B17_POLI ReactiveEnergyExport	190.575
SM_B17_POLI ReactiveEnergyExportDay	204.614
SM_B17_POLI ReactiveEnergyImport	191.666
SM_B17_POLI ReactiveEnergyImportDay	194.188
SM_B17_POLI ReactivePower	192.163
SM_B17_POLI ReactivePower1	220.945
SM_B17_POLI RelativeTHDVoltage	215.558
SM_B17_POLI TotalPowerFactor	188.139
SM_B17_POLI V1	199.379
SM_B17_POLI V1Angle	217.462
SM_B17_POLI V2	191.692
SM_B17_POLI V2Angle	189.945
SM_B17_POLI V3	220.658
SM_B17_POLI V3Angle	221.103
SM_B18_PARQ ActiveEnergyExport	181.512
SM_B18_PARQ ActiveEnergyExportDay	180.218
SM_B18_PARQ ActiveEnergyImport	201.014
SM_B18_PARQ ActiveEnergyImportDay	177.112
SM_B18_PARQ ActivePower	193.163

Colección	Documentos
SM_B18_PARQ ActivePower1	193.506
SM_B18_PARQ Frequency	224.041
SM_B18_PARQ I1	201.065
SM_B18_PARQ I2	195.275
SM_B18_PARQ I3	180.670
SM_B18_PARQ In	171.128
SM_B18_PARQ ReactiveEnergyExport	198.767
SM_B18_PARQ ReactiveEnergyExportDay	189.400
SM_B18_PARQ ReactiveEnergyImport	195.624
SM_B18_PARQ ReactiveEnergyImportDay	201.698
SM_B18_PARQ ReactivePower	207.333
SM_B18_PARQ ReactivePower1	210.981
SM_B18_PARQ RelativeTHDVoltage	187.795
SM_B18_PARQ TotalPowerFactor	171.967
SM_B18_PARQ V1	195.422
SM_B18_PARQ V1Angle	171.728
SM_B18_PARQ V2	206.370
SM_B18_PARQ V2Angle	195.273
SM_B18_PARQ V3	175.938
SM_B18_PARQ V3Angle	201.032
SM_B3_RECT ActiveEnergyExport	252.203
SM_B3_RECT ActiveEnergyExportDay	192.344
SM_B3_RECT ActiveEnergyImport	222.512
SM_B3_RECT ActiveEnergyImportDay	259.206
SM_B3_RECT ActivePower	198.466
SM_B3_RECT ActivePower1	229.896
SM_B3_RECT Frequency	192.867

Colección	Documentos
SM_B3_RECT I1	202.538
SM_B3_RECT I2	217.610
SM_B3_RECT I3	216.505
SM_B3_RECT In	200.970
SM_B3_RECT ReactiveEnergyExport	223.724
SM_B3_RECT ReactiveEnergyExportDay	213.347
SM_B3_RECT ReactiveEnergyImport	197.790
SM_B3_RECT ReactiveEnergyImportDay	262.835
SM_B3_RECT ReactivePower	234.860
SM_B3_RECT ReactivePower1	197.229
SM_B3_RECT RelativeTHDVoltage	239.468
SM_B3_RECT TotalPowerFactor	208.175
SM_B3_RECT V1	218.679
SM_B3_RECT V1Angle	238.995
SM_B3_RECT V2	190.448
SM_B3_RECT V2Angle	203.333
SM_B3_RECT V3	263.582
SM_B3_RECT V3Angle	200.895
SM_B4_PRIM ActiveEnergyExport	197.024
SM_B4_PRIM ActiveEnergyExportDay	197.611
SM_B4_PRIM ActiveEnergyImport	195.509
SM_B4_PRIM ActiveEnergyImportDay	236.420
SM_B4_PRIM ActivePower	238.506
SM_B4_PRIM ActivePower1	237.120
SM_B4_PRIM Frequency	216.007
SM_B4_PRIM I1	189.161
SM_B4_PRIM I2	202.984

Colección	Documentos
SM_B4_PRIM I3	201.465
SM_B4_PRIM In	200.681
SM_B4_PRIM ReactiveEnergyExport	237.135
SM_B4_PRIM ReactiveEnergyExportDay	214.150
SM_B4_PRIM ReactiveEnergyImport	214.127
SM_B4_PRIM ReactiveEnergyImportDay	181.647
SM_B4_PRIM ReactivePower	219.657
SM_B4_PRIM ReactivePower1	221.214
SM_B4_PRIM RelativeTHDVoltage	190.645
SM_B4_PRIM TotalPowerFactor	230.359
SM_B4_PRIM V1	213.913
SM_B4_PRIM V1Angle	191.552
SM_B4_PRIM V2	192.635
SM_B4_PRIM V2Angle	219.122
SM_B4_PRIM V3	208.282
SM_B4_PRIM V3Angle	185.114
SM_B5_BACH ActiveEnergyExport	188.680
SM_B5_BACH ActiveEnergyExportDay	189.428
SM_B5_BACH ActiveEnergyImport	216.014
SM_B5_BACH ActiveEnergyImportDay	201.348
SM_B5_BACH ActivePower	236.975
SM_B5_BACH ActivePower1	217.213
SM_B5_BACH Frequency	205.976
SM_B5_BACH I1	193.558
SM_B5_BACH I2	218.942
SM_B5_BACH I3	222.502
SM_B5_BACH In	197.614

Colección	Documentos
SM_B5_BACH ReactiveEnergyExport	246.117
SM_B5_BACH ReactiveEnergyExportDay	237.929
SM_B5_BACH ReactiveEnergyImport	238.645
SM_B5_BACH ReactiveEnergyImportDay	211.395
SM_B5_BACH ReactivePower	210.166
SM_B5_BACH ReactivePower1	214.283
SM_B5_BACH RelativeTHDVoltage	223.531
SM_B5_BACH TotalPowerFactor	210.359
SM_B5_BACH V1	212.772
SM_B5_BACH V1Angle	218.432
SM_B5_BACH V2	218.258
SM_B5_BACH V2Angle	205.872
SM_B5_BACH V3	232.655
SM_B5_BACH V3Angle	203.127
SM_B7_CTIC ActiveEnergyExport	210.911
SM_B7_CTIC ActiveEnergyExportDay	188.228
SM_B7_CTIC ActiveEnergyImport	248.570
SM_B7_CTIC ActiveEnergyImportDay	185.016
SM_B7_CTIC ActivePower	192.663
SM_B7_CTIC ActivePower1	219.618
SM_B7_CTIC Frequency	213.919
SM_B7_CTIC I1	212.548
SM_B7_CTIC I2	215.754
SM_B7_CTIC I3	209.073
SM_B7_CTIC In	247.619
SM_B7_CTIC ReactiveEnergyExport	216.355
SM_B7_CTIC ReactiveEnergyExportDay	195.170

Colección	Documentos
SM_B7_CTIC ReactiveEnergyImport	200.183
SM_B7_CTIC ReactiveEnergyImportDay	194.033
SM_B7_CTIC ReactivePower	187.212
SM_B7_CTIC ReactivePower1	226.587
SM_B7_CTIC RelativeTHDVoltage	219.180
SM_B7_CTIC TotalPowerFactor	183.582
SM_B7_CTIC V1	233.184
SM_B7_CTIC V1Angle	235.125
SM_B7_CTIC V2	221.548
SM_B7_CTIC V2Angle	217.485
SM_B7_CTIC V3	233.091
SM_B7_CTIC V3Angle	229.410
SM_B7_TAC ActiveEnergyExport	199.387
SM_B7_TAC ActiveEnergyExportDay	188.721
SM_B7_TAC ActiveEnergyImport	192.683
SM_B7_TAC ActiveEnergyImportDay	228.465
SM_B7_TAC ActivePower	248.481
SM_B7_TAC ActivePower1	222.404
SM_B7_TAC Frequency	216.904
SM_B7_TAC I1	204.145
SM_B7_TAC I2	217.809
SM_B7_TAC I3	226.458
SM_B7_TAC In	242.163
SM_B7_TAC ReactiveEnergyExport	229.961
SM_B7_TAC ReactiveEnergyExportDay	201.921
SM_B7_TAC ReactiveEnergyImport	199.974
SM_B7_TAC ReactiveEnergyImportDay	188.703

Colección	Documentos
SM_B7_TAC ReactivePower	222.482
SM_B7_TAC ReactivePower1	179.192
SM_B7_TAC RelativeTHDVoltage	213.645
SM_B7_TAC TotalPowerFactor	254.751
SM_B7_TAC V1	199.879
SM_B7_TAC V1Angle	203.818
SM_B7_TAC V2	202.160
SM_B7_TAC V2Angle	211.911
SM_B7_TAC V3	197.870
SM_B7_TAC V3Angle	250.097
SM_B8_AA ActiveEnergyExport	218.239
SM_B8_AA ActiveEnergyExportDay	200.658
SM_B8_AA ActiveEnergyImport	211.058
SM_B8_AA ActiveEnergyImportDay	203.511
SM_B8_AA ActivePower	214.639
SM_B8_AA ActivePower1	204.353
SM_B8_AA Frequency	214.193
SM_B8_AA I1	226.570
SM_B8_AA I2	243.234
SM_B8_AA I3	215.440
SM_B8_AA In	231.485
SM_B8_AA ReactiveEnergyExport	220.020
SM_B8_AA ReactiveEnergyExportDay	221.331
SM_B8_AA ReactiveEnergyImport	220.302
SM_B8_AA ReactiveEnergyImportDay	210.308
SM_B8_AA ReactivePower	239.364
SM_B8_AA ReactivePower1	227.361

Colección	Documentos
SM_B8_AA RelativeTHDVoltage	217.474
SM_B8_AA TotalPowerFactor	234.078
SM_B8_AA V1	223.474
SM_B8_AA V1Angle	203.836
SM_B8_AA V2	220.860
SM_B8_AA V2Angle	197.995
SM_B8_AA V3	219.128
SM_B8_AA V3Angle	216.397
SM_B8_CPA ActiveEnergyExport	220.929
SM_B8_CPA ActiveEnergyExportDay	193.128
SM_B8_CPA ActiveEnergyImport	219.875
SM_B8_CPA ActiveEnergyImportDay	232.943
SM_B8_CPA ActivePower	210.538
SM_B8_CPA ActivePower1	219.346
SM_B8_CPA Frequency	213.769
SM_B8_CPA I1	197.173
SM_B8_CPA I2	193.977
SM_B8_CPA I3	251.161
SM_B8_CPA In	224.104
SM_B8_CPA ReactiveEnergyExport	201.032
SM_B8_CPA ReactiveEnergyExportDay	236.729
SM_B8_CPA ReactiveEnergyImport	268.318
SM_B8_CPA ReactiveEnergyImportDay	213.391
SM_B8_CPA ReactivePower	191.463
SM_B8_CPA ReactivePower1	219.451
SM_B8_CPA RelativeTHDVoltage	193.797
SM_B8_CPA TotalPowerFactor	191.130

Colección	Documentos
SM_B8_CPA V1	226.394
SM_B8_CPA V1Angle	220.486
SM_B8_CPA V2	237.434
SM_B8_CPA V2Angle	227.662
SM_B8_CPA V3	191.472
SM_B8_CPA V3Angle	202.299
SM_B8_LABS ActiveEnergyExport	204.157
SM_B8_LABS ActiveEnergyExportDay	210.804
SM_B8_LABS ActiveEnergyImport	228.446
SM_B8_LABS ActiveEnergyImportDay	205.011
SM_B8_LABS ActivePower	200.235
SM_B8_LABS ActivePower1	219.907
SM_B8_LABS Frequency	189.436
SM_B8_LABS I1	210.140
SM_B8_LABS I2	191.810
SM_B8_LABS I3	189.198
SM_B8_LABS In	215.394
SM_B8_LABS ReactiveEnergyExport	248.226
SM_B8_LABS ReactiveEnergyExportDay	191.454
SM_B8_LABS ReactiveEnergyImport	209.018
SM_B8_LABS ReactiveEnergyImportDay	191.255
SM_B8_LABS ReactivePower	192.296
SM_B8_LABS ReactivePower1	261.999
SM_B8_LABS RelativeTHDVoltage	213.835
SM_B8_LABS TotalPowerFactor	204.319
SM_B8_LABS V1	238.164
SM_B8_LABS V1Angle	189.182

Colección	Documentos
SM_B8_LABS V2	190.625
SM_B8_LABS V2Angle	193.660
SM_B8_LABS V3	198.960
SM_B8_LABS V3Angle	234.881
SM_B9_SFA1 ActiveEnergyExport	197.607
SM_B9_SFA1 ActiveEnergyExportDay	193.618
SM_B9_SFA1 ActiveEnergyImport	192.435
SM_B9_SFA1 ActiveEnergyImportDay	222.341
SM_B9_SFA1 ActivePower	246.394
SM_B9_SFA1 ActivePower1	227.430
SM_B9_SFA1 Frequency	239.732
SM_B9_SFA1 I1	220.878
SM_B9_SFA1 I2	202.690
SM_B9_SFA1 I3	197.261
SM_B9_SFA1 In	220.427
SM_B9_SFA1 ReactiveEnergyExport	207.767
SM_B9_SFA1 ReactiveEnergyExportDay	201.219
SM_B9_SFA1 ReactiveEnergyImport	228.729
SM_B9_SFA1 ReactiveEnergyImportDay	194.857
SM_B9_SFA1 ReactivePower	235.915
SM_B9_SFA1 ReactivePower1	226.152
SM_B9_SFA1 RelativeTHDVoltage	233.356
SM_B9_SFA1 TotalPowerFactor	209.331
SM_B9_SFA1 V1	197.654
SM_B9_SFA1 V1Angle	181.514
SM_B9_SFA1 V2	177.941
SM_B9_SFA1 V2Angle	213.435

Colección	Documentos
SM_B9_SFA1 V3	196.347
SM_B9_SFA1 V3Angle	187.355
SM_B9_SFA2 ActiveEnergyExport	203.616
SM_B9_SFA2 ActiveEnergyExportDay	192.484
SM_B9_SFA2 ActiveEnergyImport	213.516
SM_B9_SFA2 ActiveEnergyImportDay	240.555
SM_B9_SFA2 ActivePower	191.652
SM_B9_SFA2 ActivePower1	189.717
SM_B9_SFA2 Frequency	224.180
SM_B9_SFA2 I1	192.020
SM_B9_SFA2 I2	262.465
SM_B9_SFA2 I3	258.944
SM_B9_SFA2 In	270.850
SM_B9_SFA2 ReactiveEnergyExport	190.654
SM_B9_SFA2 ReactiveEnergyExportDay	203.737
SM_B9_SFA2 ReactiveEnergyImport	206.651
SM_B9_SFA2 ReactiveEnergyImportDay	209.040
SM_B9_SFA2 ReactivePower	189.029
SM_B9_SFA2 ReactivePower1	216.392
SM_B9_SFA2 RelativeTHDVoltage	199.419
SM_B9_SFA2 TotalPowerFactor	231.297
SM_B9_SFA2 V1	208.189
SM_B9_SFA2 V1Angle	217.999
SM_B9_SFA2 V2	199.940
SM_B9_SFA2 V2Angle	202.460
SM_B9_SFA2 V3	216.259
SM_B9_SFA2 V3Angle	262.036

Colección	Documentos
SM_HABITAT ActiveEnergyExport	197.169
SM_HABITAT ActiveEnergyExportDay	182.749
SM_HABITAT ActiveEnergyImport	376.630
SM_HABITAT ActiveEnergyImportDay	199.072
SM_HABITAT ActivePower	235.727
SM_HABITAT ActivePower1	192.315
SM_HABITAT Frequency	246.564
SM_HABITAT I1	203.573
SM_HABITAT I2	199.451
SM_HABITAT I3	191.470
SM_HABITAT In	200.000
SM_HABITAT ReactiveEnergyExport	191.466
SM_HABITAT ReactiveEnergyExportDay	225.423
SM_HABITAT ReactiveEnergyImport	190.175
SM_HABITAT ReactiveEnergyImportDay	196.379
SM_HABITAT ReactivePower	211.469
SM_HABITAT ReactivePower1	197.839
SM_HABITAT RelativeTHDVoltage	227.621
SM_HABITAT TotalPowerFactor	224.986
SM_HABITAT V1	191.479

Colección	Documentos
SM_HABITAT V1Angle	221.589
SM_HABITAT V2	207.854
SM_HABITAT V2Angle	199.933
SM_HABITAT V3	224.192
SM_HABITAT V3Angle	206.117
SR B11_Radiation	267.678
SR B11_Temperature	236.102
SR PedestrianRadiation	292.108
SR PedestrianTemperature	290.469
SR VehicularRadiation	286.091
SR VehicularTemperature	286.622
WS ICA_PM10	235.546
WS ICA_PM2.5	235.553
WS WS_PM1	235.797
WS WS_PM10	236.061
WS WS_PM2.5	235.792
WS WS_Radiation	235.962
WS WS_WindDirection	235.993
WS WS_WindSpeed	236.323