UNIVERSIDAD TECNOLÓGICA DE CHIHUAHUA

TECNOLOGÍAS DE LA INFORMACIÓN

EXTRACCIÓN DE CONOCIMIENTO EN BASES DE DATOS

REPORTE DE INVESTIGACIÓN DE LOS LENGUAJES Y BIBLIOTECAS PARA ANÁLISIS Y PROCESAMIENTO DE DATOS

IDGS81N

PRESENTA:

REGINA CHÁVEZ TAMAYO

DOCENTE:

LUIS ENRIQUE MASCOTE CANO

Chihuahua, Chih., 21 de septiembre de 2025

Contents

Introducción

El campo de la ciencia de datos y la inteligencia artificial se ha convertido en un eje central para la innovación tecnológica, donde la elección del lenguaje de programación y sus bibliotecas asociadas puede marcar la diferencia en el éxito de un proyecto. Cada lenguaje presenta características particulares que impactan en la facilidad de aprendizaje, el rendimiento computacional, la disponibilidad de librerías especializadas y la capacidad de escalar soluciones desde un análisis exploratorio hasta sistemas de producción a gran escala. Conocer las fortalezas y limitaciones de los principales lenguajes empleados en proyectos de análisis de datos, minería de datos, aprendizaje automático y Big Data resulta esencial tanto para estudiantes como para profesionales.

El presente reporte tiene como objetivo describir y comparar ocho lenguajes de programación ampliamente utilizados en este campo: Python, R, Scala, SQL, Julia, Java, SAS y MATLAB. De cada uno se revisan sus características generales, el paradigma de programación que lo sustenta, los ámbitos de aplicación más comunes, las bibliotecas o frameworks clave que lo acompañan y un breve ejemplo práctico de manipulación de datos. Finalmente, se ofrece una reflexión comparativa que resalta las semejanzas, diferencias y posibles recomendaciones para distintos escenarios de aplicación.

Lenguajes y bibliotecas para análisis y procesamiento de datos

Python

Descripción general:

Python es un lenguaje interpretado, de tipado dinámico y multiparadigma. Es ampliamente utilizado por su sintaxis sencilla y por la gran comunidad que lo respalda. Su ecosistema de librerías lo hace ideal para proyectos de ciencia de datos, aprendizaje automático, desarrollo web y prototipado rápido.

Paradigma: Interpretado; tipado dinámico; multiparadigma (imperativo, orientado a objetos y funcional).

Ámbito de uso principal: Ciencia de datos, aprendizaje automático y profundo (ML/DL), desarrollo de APIs y automatización.

Bibliotecas y frameworks clave:

- **pandas.** Estructuras de datos tipo DataFrame para manipulación tabular.
  Caso de uso: carga y limpieza de archivos CSV para análisis inicial.
- **scikit-learn.** Conjunto de algoritmos de ML clásico.
  Caso de uso: entrenamiento de un modelo de regresión con validación cruzada.
- **PyTorch.** Framework para redes neuronales profundas y tensores en GPU.
  Caso de uso: entrenamiento de un modelo de clasificación de imágenes.

Ejemplo:

R

Descripción general:

R es un lenguaje interpretado y de tipado dinámico, diseñado específicamente para estadística y visualización. Su sintaxis vectorizada facilita operaciones estadísticas y lo convierte en una herramienta esencial en investigación científica y académica.

Paradigma: Interpretado; tipado dinámico; orientación estadística.

Ámbito de uso principal: Análisis estadístico, visualización de datos y generación de reportes reproducibles.

Bibliotecas y frameworks clave:

- tidyverse. Colección de paquetes para importar, transformar y visualizar datos.
  Caso de uso: visualización de tendencias con ggplot2.
- data.table. Herramienta eficiente para manejar grandes volúmenes de datos tabulares.
  Caso de uso: agregaciones rápidas sobre millones de registros.
- tidymodels. Framework unificado para flujos de trabajo de ML.
  Caso de uso: entrenamiento de un modelo de clasificación con métricas reproducibles.

Ejemplo:

Scala

Descripción general:

Scala es un lenguaje compilado a bytecode de la JVM, de tipado estático, que combina programación funcional y orientada a objetos. Se distingue en entornos de Big Data por ser el lenguaje nativo de Apache Spark.

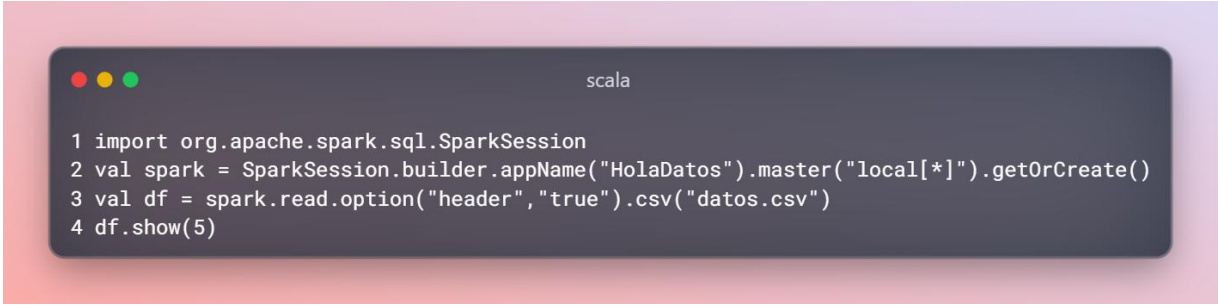
Paradigma: Compilado; tipado estático; orientado a objetos y funcional.

Ámbito de uso principal: Procesamiento de grandes volúmenes de datos en clústeres y aplicaciones distribuidas.

Bibliotecas y frameworks clave:

- Apache Spark (Scala API). Procesamiento distribuido con SQL, streaming y MLlib.
  Caso de uso: análisis de grandes volúmenes de datos en DataFrames.
- Breeze. Librería matemática para álgebra lineal y optimización.
  Caso de uso: operaciones vectoriales en modelos de ML.

Ejemplo:



SQL

Descripción general:

SQL es un lenguaje declarativo para interactuar con bases de datos relacionales. Aunque no es de propósito general, constituye el estándar en almacenamiento estructurado y manipulación de datos tabulares.

Paradigma: Declarativo; ejecución optimizada por el motor de base de datos.

Ámbito de uso principal: Modelado de datos, consultas analíticas y procesos ETL/ELT.

Bibliotecas y frameworks clave:

- COPY (PostgreSQL). Comando para importar grandes archivos CSV.
  Caso de uso: carga masiva de datos a una tabla de staging.

- BULK INSERT (SQL Server). Funcionalidad para ingestión de archivos externos.
  Caso de uso: integración de datos desde archivos locales a una base de datos.

Ejemplo:

Julia

Descripción general:

Julia es un lenguaje moderno, compilado Just-In-Time (JIT), que logra un rendimiento cercano a C con una sintaxis clara. Combina tipado dinámico con especialización de funciones.

Paradigma: Compilado JIT; tipado dinámico con optimización; multiparadigma.

Ámbito de uso principal: Computación científica, análisis numérico de alto rendimiento y aprendizaje automático.

Bibliotecas y frameworks clave:

- DataFrames.jl. Manipulación tabular al estilo pandas o dplyr.
  Caso de uso: agrupación de datos experimentales.
- CSV.jl. Lectura y escritura rápida de archivos delimitados.
  Caso de uso: carga de un dataset extenso en paralelo.
- MLJ.jl. Framework unificado para aprendizaje automático en Julia.
  Caso de uso: comparar y ajustar algoritmos de ML.

Ejemplo:

Java

Descripción general:

Java es un lenguaje compilado, de tipado estático y orientado a objetos, con gran relevancia en sistemas empresariales. Aunque no nació para la ciencia de datos, se integra eficazmente con Spark y frameworks de aprendizaje profundo en la JVM.

Paradigma: Compilado; tipado estático; orientado a objetos.

Ámbito de uso principal: Procesamiento distribuido, aplicaciones empresariales y producción a gran escala.

Bibliotecas y frameworks clave:

- Apache Spark (Java API). Framework para procesamiento de datos en clústeres.
  Caso de uso: construcción de pipelines ETL distribuidos.
- Deeplearning4j. Framework de deep learning en la JVM.
  Caso de uso: entrenamiento de redes neuronales en sistemas Java.

Ejemplo:

SAS

Descripción general:

SAS es un entorno propietario orientado a estadística y análisis de datos. Ofrece estabilidad, soporte técnico y certificaciones, lo que lo hace relevante en sectores regulados como salud y finanzas.

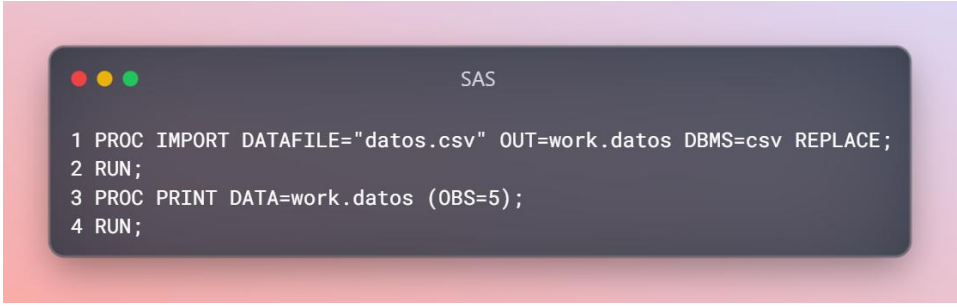
Paradigma: Interpretado; propietario; orientado a estadística.

Ámbito de uso principal: Estadística avanzada, minería de datos y generación de reportes en industrias reguladas.

Bibliotecas y frameworks clave:

- SAS/STAT. Conjunto de procedimientos estadísticos avanzados.
  Caso de uso: análisis de regresión logística en investigación clínica.
- SAS Enterprise Miner. Herramienta para minería de datos y modelado predictivo.
  Caso de uso: construcción de árboles de decisión para análisis de riesgo.

Ejemplo:



MATLAB

Descripción general:

MATLAB es un lenguaje interpretado y de tipado dinámico enfocado en cómputo numérico y simulaciones. Es muy utilizado en ingeniería, procesamiento de señales y matemáticas aplicadas.

Paradigma: Interpretado; tipado dinámico; orientado a cómputo numérico.

Ámbito de uso principal: Simulación, procesamiento de señales, análisis estadístico y machine learning.

Bibliotecas y frameworks clave:

- Statistics and Machine Learning Toolbox. Algoritmos de regresión, clasificación y clustering.

  Caso de uso: análisis de regresión lineal en un dataset experimental.
- Deep Learning Toolbox. Framework para el desarrollo de redes neuronales.

  Caso de uso: entrenamiento de una CNN para clasificación de imágenes.

Ejemplo:

Conclusión

Los lenguajes de programación revisados en este reporte evidencian la diversidad de opciones disponibles para trabajar con datos en distintos contextos. Python y R destacan por su sencillez, expresividad y vasto ecosistema de bibliotecas, lo que los convierte en la opción preferida para análisis exploratorios, proyectos académicos y prototipos rápidos. Julia, con su capacidad de combinar sintaxis sencilla y velocidad cercana a C, emerge como una alternativa poderosa para cómputo científico de alto rendimiento. SQL, aunque no es un lenguaje de propósito general, sigue siendo indispensable para la organización y extracción de datos estructurados en prácticamente cualquier pipeline de datos.

Por otro lado, lenguajes como Scala y Java muestran su fortaleza en entornos empresariales y de Big Data gracias a su integración con Apache Spark y a su robustez en sistemas distribuidos. Mientras que SAS y MATLAB, a pesar de ser soluciones comerciales, mantienen un papel importante en sectores regulados, investigación aplicada y contextos donde la estabilidad y el soporte empresarial son prioritarios.

En términos generales, para proyectos de análisis ligero y exploratorio es recomendable optar por Python o R debido a su productividad y facilidad de aprendizaje. Para proyectos de producción a gran escala que requieren procesamiento distribuido, Scala y Java resultan más convenientes. Julia ofrece un equilibrio interesante para quienes buscan rendimiento sin sacrificar legibilidad. SAS y MATLAB, aunque menos flexibles, son especialmente útiles en industrias específicas donde el soporte, las certificaciones y las herramientas integradas son factores determinantes.

Referencias

Apache Spark: Documentation. (s. f.). *Apache Software Foundation*. Recuperado el 21 de septiembre de 2025 de https://spark.apache.org/docs/latest/

ChatGPT (GPT-5). (2025). Asistente para consulta de información sobre lenguajes y bibliotecas de programación.

DataFrames.jl. (s. f.). *JuliaData*. Recuperado el 21 de septiembre de 2025 de https://juliadata.github.io/DataFrames.jl/stable/

Deeplearning4j overview. (2023). *Konduit AI*. Recuperado el 21 de septiembre de 2025 de https://deeplearning4j.konduit.ai/

Dowle, M., & Srinivasan, A. (2025). *data.table: Introduction vignette*. CRAN. Recuperado el 21 de septiembre de 2025 de https://cran.r-project.org/package=data.table/vignettes/datatable-intro.html

MathWorks. (s. f.). *MATLAB documentation*. Recuperado el 21 de septiembre de 2025 de https://www.mathworks.com/help/

Microsoft. (2025). *BULK INSERT (Transact-SQL)*. Microsoft Learn. Recuperado el 21 de septiembre de 2025 de https://learn.microsoft.com/sql/t-sql/statements/bulk-insert-transact-sql

pandas documentation. (s. f.). *PyData*. Recuperado el 21 de septiembre de 2025 de https://pandas.pydata.org/docs/

PostgreSQL Global Development Group. (s. f.). *COPY*. Recuperado el 21 de septiembre de 2025 de https://www.postgresql.org/docs/current/sql-copy.html

PyTorch documentation. (s. f.). *PyTorch Foundation*. Recuperado el 21 de septiembre de 2025 de https://docs.pytorch.org/

scikit-learn documentation. (s. f.). *scikit-learn developers*. Recuperado el 21 de septiembre de 2025 de https://scikit-learn.org/

SAS Institute. (s. f.). *SAS/STAT User's Guide*. Recuperado el 21 de septiembre de 2025 de https://documentation.sas.com/

Tidyverse. (s. f.). *RStudio PBC*. Recuperado el 21 de septiembre de 2025 de https://www.tidyverse.org/

Tidymodels. (s. f.). *RStudio PBC*. Recuperado el 21 de septiembre de 2025 de https://www.tidymodels.org/