

UNIVERSIDAD TECNOLÓGICA DE CHIHUAHUA

Ingeniería en Desarrollo y Gestión de Software



Extracción de Conocimiento en Bases de Datos

**I.4. Reporte de investigación de los lenguajes y bibliotecas
para análisis y procesamiento de datos**

IDGS91N

PRESENTA:

Giselle Cantú Chávez

NOMBRE DEL DOCENTE:

Ing. Luis Enrique Mascote Cano

Chihuahua, Chih., 23 de septiembre de 2025

ÍNDICE

INTRODUCCIÓN	3
IMPORTANCIA DE CONOCER LOS LENGUAJES Y BIBLIOTECAS DE ANÁLISIS DE DATOS	5
OBJETIVOS DEL REPORTE	6
SECCIONES POR LENGUAJE.....	7
Python.....	7
R	8
Scala.....	9
SQL	10
Julia	10
Java	11
MATLAB	12
CONCLUSIONES Y PRÓXIMOS PASOS.....	13
REFERENCIAS Y FUENTES CONSULTADAS	15

ÍNDICE DE FIGURAS

Ilustración 1	8
---------------------	---

INTRODUCCIÓN

Hoy en día, trabajar con datos se ha vuelto parte esencial de casi cualquier área del conocimiento. Por esa razón, me parece importante conocer los lenguajes de programación y las bibliotecas que se utilizan para analizarlos y procesarlos. Estos lenguajes son la base para que un proyecto de inteligencia artificial, machine learning, minería de datos o big data pueda avanzar con buenos resultados. Como señalan Arcila Calderón, Barbosa Caro y Cabezuelo Lorenzo (2016), el crecimiento constante de la información hace necesario contar con tecnologías que permitan gestionarla de forma eficiente.

El propósito de este reporte es revisar los principales lenguajes que se aplican en el análisis de datos, junto con sus bibliotecas más usadas. En este caso decidí centrarme en Python, R, Scala, SQL y Julia, además de incluir referencias a Java, SAS y MATLAB porque siguen teniendo presencia en ámbitos académicos y profesionales. Lo que busco es mostrar qué ofrece cada uno, en qué situaciones conviene usarlos y qué tipo de proyectos se benefician de sus características.

Realizar esta investigación me ayuda a tener criterios más claros para elegir las herramientas adecuadas según el tipo de proyecto. Como mencionan Flórez et al. (2020), el uso correcto de un entorno y sus librerías facilita la resolución de problemas de manera práctica y replicable. Entender estos aspectos es una parte de mi formación profesional, ya que me prepara para tomar decisiones mejor fundamentadas en futuros proyectos.

También considero que este ejercicio me aporta experiencia personal, porque no se trata solo de enumerar lenguajes, sino de comprender qué ventajas y limitaciones tienen en el mundo real. Chancusig et al. (2024) destacan que trabajar con bibliotecas estadísticas en Python, por ejemplo, refuerza la toma de decisiones basada en evidencia. Para mí, esa es la clase de conocimiento que conecta lo académico con lo que se necesita en el ámbito laboral.

Por último, este reporte está organizado de la siguiente manera: primero presento cada lenguaje y algunas de sus bibliotecas más representativas, junto con

un fragmento sencillo de código. Después incluyo una breve comparación general entre ellos, resaltando similitudes y diferencias. Finalmente, cierro con una reflexión sobre qué lenguajes recomendaría para proyectos pequeños y cuáles funcionan mejor en proyectos a gran escala.

Hay que reconocer que la elección de un lenguaje no siempre se basa únicamente en cuestiones de eficiencia o rendimiento; también se basa en las condiciones del entorno donde se va a implementar. En organizaciones con infraestructura ya establecida, puede resultar más sencillo integrar SQL o SAS, mientras que en proyectos de investigación académica puede ser más común recurrir a R o Python. Estas decisiones reflejan la realidad de que no existe un único lenguaje perfecto; existen herramientas con propósitos distintos.

IMPORTANCIA DE CONOCER LOS LENGUAJES Y BIBLIOTECAS DE ANÁLISIS DE DATOS

Para mí es importante conocer los lenguajes y bibliotecas de análisis de datos porque son las herramientas que realmente hacen posible trabajar con la información. No basta con tener los datos, hay que saber cómo procesarlos y darles sentido. Arcila et al. (2016) mencionan que el valor de la información está en la capacidad de analizarla con técnicas adecuadas, y justo ahí entran estos lenguajes. Python y R, por ejemplo, han demostrado ser muy útiles porque cuentan con bibliotecas que permiten organizar, transformar y modelar los datos de distintas maneras (Ruiz, 2016). Coincido con lo que señalan Flórez et al. (2020): aprender estas herramientas no solo resuelve problemas inmediatos, también abre nuevas posibilidades para innovar en proyectos de diferentes áreas.

OBJETIVOS DEL REPORTE

El principal objetivo de este reporte es identificar y describir los lenguajes de programación y sus bibliotecas más utilizadas en el análisis y procesamiento de datos. Con ello busco mostrar las características de cada uno y los contextos en los que resultan más adecuados.

Otro de mis objetivos es reconocer cómo estas herramientas influyen en proyectos de inteligencia artificial, machine learning, minería de datos y big data. Tal como señalan Arcila et al. (2016), el crecimiento de los datos exige seleccionar tecnologías que permitan analizarlos de manera eficiente, y por eso me interesa revisar opciones diversas como Python, R, Scala, SQL, Julia y algunos lenguajes adicionales.

Finalmente, me propongo comparar estas alternativas para tener un criterio más claro sobre qué lenguajes conviene aplicar en proyectos de pequeña escala y cuáles son más apropiados en entornos de producción. Este ejercicio también forma parte de mi preparación profesional, ya que me da las bases para tomar decisiones más informadas en mi futuro laboral (Flórez et al., 2020).

SECCIONES POR LENGUAJE

Python

Python es un lenguaje de programación interpretado, de tipado dinámico y multiplataforma. Su sintaxis sencilla y su amplia comunidad lo han convertido en una de las opciones más utilizadas en ciencia de datos, inteligencia artificial y machine learning. Es versátil porque no solo se usa en análisis de datos, también en desarrollo web, automatización y back-end. Según Chimarro et al. (2023), el crecimiento de Python se explica justamente por su capacidad de adaptarse a distintos contextos y por el soporte de una gran cantidad de bibliotecas especializadas.

Bibliotecas y frameworks:

- ✓ **pandas:** permite trabajar con datos estructurados mediante DataFrames; se utiliza para limpiar, transformar y analizar información tabular. Por ejemplo, resulta muy útil al procesar hojas de cálculo o bases en formato CSV.
- ✓ **NumPy:** ofrece estructuras de datos eficientes y operaciones matemáticas avanzadas, esenciales en cálculos numéricos y manipulación de arreglos multidimensionales.
- ✓ **scikit-learn:** proporciona algoritmos de machine learning listos para usar, desde regresión hasta clustering y clasificación. De acuerdo con Flórez et al. (2020), la integración de estas bibliotecas facilita la resolución práctica de problemas complejos en entornos de datos reales.

Ilustración 1

Ejemplo de código

```
import pandas as pd

# Cargar archivo CSV
datos = pd.read_csv("ejemplo.csv")

# Mostrar primeras filas
print(datos.head())
```

Nombre	Edad	Ciudad
Ana	23	CDMX
Luis	30	Guadalajara
Maria	27	Monterrey
Carlos	35	Puebla
Giselle	22	Chihuahua

R

R es un lenguaje interpretado y de tipado dinámico, diseñado específicamente para la estadística y la visualización de datos. Ruiz (2016) menciona que su fortaleza está en el análisis cuantitativo y en el respaldo de una amplia comunidad académica.

Bibliotecas y frameworks:

- ✓ **ggplot2**: visualización de datos con gráficos personalizables.
- ✓ **dplyr**: manipulación eficiente de tablas de datos.
- ✓ **tidyverse**: organización y limpieza de datos.

Mini-ejemplo “Hola datos”:

```
datos <- read.csv("ejemplo.csv")  
  
head(datos)
```

Scala

Scala es un lenguaje compilado, de tipado estático, que combina programación orientada a objetos y funcional. Se utiliza principalmente en entornos de big data junto a Apache Spark, por su capacidad de manejar flujos de información masivos en tiempo real. Odersky et al. (2016) destacan su flexibilidad para desarrollar sistemas distribuidos y analíticos.

Bibliotecas / frameworks

- **Spark SQL**: procesamiento de datos estructurados.
- **Mlib**: algoritmos de machine learning escalables.
- **Akka**: desarrollo de aplicaciones concurrentes y distribuidas.

Mini-ejemplo “Hola datos”:

```
import org.apache.spark.sql.SparkSession  
  
val spark = SparkSession.builder.appName("HolaDatos").getOrCreate()
```

```
val datos = spark.read.option("header","true").csv("ejemplo.csv")
datos.show(5)
```

SQL

SQL (Structured Query Language) es un lenguaje declarativo, utilizado para gestionar y consultar bases de datos relacionales. Es compilado a nivel de motor y su tipado depende del sistema gestor. Stonebraker y Hellerstein (2015) señalan que SQL sigue siendo esencial porque ofrece un estándar confiable para almacenar y recuperar datos.

Funciones / extensiones:

- **SELECT**: consulta y filtrado de datos.
- **JOIN**: combinación de tablas.
- **GROUP BY**: agregación de resultados.

Mini-ejemplo “Hola datos”:

```
SELECT *
FROM empleados
LIMIT 5;
```

Julia

Julia es un lenguaje compilado y de tipado dinámico que se ha consolidado en el análisis numérico y la computación científica. Su principal ventaja es combinar velocidad cercana a C con una sintaxis similar a Python. Según The Julia Project (n.d.), está diseñado para cálculos intensivos y aplicaciones de modelado avanzado.

Bibliotecas:

- **DataFrames.jl**: manejo de tablas de datos.
- **CSV.jl**: lectura y escritura de archivos CSV.
- **StatsBase.jl**: funciones estadísticas.

Mini-ejemplo “Hola datos”:

```
using CSV, DataFrames
```

```
datos = CSV.read("ejemplo.csv", DataFrame)  
first(datos, 5)
```

Java

Java es un lenguaje compilado, de tipado estático, orientado a objetos y multiplataforma. Aunque no es tan flexible como Python o R en ciencia de datos, se utiliza en entornos empresariales que requieren estabilidad y escalabilidad.

Bibliotecas:

- **Weka**: plataforma de minería de datos con múltiples algoritmos.
- **Deeplearning4j**: framework de deep learning para Java.

Mini-ejemplo “Hola datos”:

```
import java.sql.*;
```

```
public class HolaDatos {  
    public static void main(String[] args) throws Exception {  
        Connection conn = DriverManager.getConnection("jdbc:sqlite:ejemplo.db");  
        Statement stmt = conn.createStatement();  
        ResultSet rs = stmt.executeQuery("SELECT * FROM empleados LIMIT 5");  
        while(rs.next()) {
```

```
        System.out.println(rs.getString("nombre"));

    }

}

}
```

MATLAB

MATLAB es un lenguaje interpretado, de tipado dinámico, especializado en matemáticas aplicadas, análisis numérico y simulaciones. MathWorks (n.d.) documenta su amplio uso en ingeniería y ciencias aplicadas.

Bibliotecas / toolboxes:

- **Statistics and Machine Learning Toolbox:** modelos estadísticos y predictivos.
- **Deep Learning Toolbox:** redes neuronales y aprendizaje profundo.

Mini-ejemplo “Hola datos”:

```
datos = readtable("ejemplo.csv");
head(datos, 5)
```

CONCLUSIONES Y PRÓXIMOS PASOS

Al terminar este reporte me doy cuenta de que cada lenguaje tiene un papel distinto en el análisis y procesamiento de datos, y que no se trata de ver cuál es mejor en general, sino cuál se adapta mejor al proyecto. Python y R son, sin duda, los más accesibles y con mayor comunidad. Python destaca por su versatilidad y por la cantidad de bibliotecas que tiene para casi cualquier tarea, mientras que R sigue siendo la opción más sólida cuando el trabajo es estrictamente estadístico y académico.

Scala y SQL representan un nivel diferente de uso. SQL es la base para trabajar con datos almacenados en sistemas relacionales y su importancia se mantiene porque ofrece un estándar claro y confiable. Scala, en cambio, se vuelve útil en entornos de big data donde el rendimiento y la escalabilidad son clave, sobre todo al trabajar con Spark. Julia me parece un lenguaje prometedor, pensado para cálculos numéricos pesados, y aunque todavía no tiene la comunidad de Python o R, es interesante por la rapidez que ofrece.

Cuando pienso en facilidad de uso, mi recomendación sería iniciar con Python o R, ya que la curva de aprendizaje es más corta y los recursos disponibles ayudan a avanzar rápido. Para proyectos pequeños o exploratorios, estos lenguajes me permiten obtener resultados claros sin complicaciones. En cambio, si se trata de proyectos de producción a gran escala, donde hay que procesar enormes volúmenes de datos y mantener rendimiento, elegiría lenguajes como Scala, acompañado de frameworks como Spark, o incluso SQL como soporte para bases de datos. Julia también puede entrar en este nivel cuando se requieren cálculos intensivos.

En general, esta comparación me deja claro que la elección del lenguaje no es arbitraria, sino estratégica. Aprender a valorar la facilidad de uso, el ecosistema de bibliotecas y el rendimiento en función del objetivo del proyecto es lo que realmente marca la diferencia. Con este ejercicio confirmo que debo seguir explorando estas herramientas para desarrollar la capacidad de elegir con criterio y

no solo por popularidad, porque al final, lo importante es que la solución responda a las necesidades reales del análisis de datos.

Otro punto importante es la relevancia de lenguajes como SAS y MATLAB en sectores específicos. SAS, por ejemplo, se ha consolidado en ámbitos empresariales y gubernamentales donde la confiabilidad de los procesos y la estandarización son prioritarias (SAS Institute, s.f.). MATLAB, por su parte, sigue siendo muy utilizado en ingeniería y áreas aplicadas que demandan simulaciones y análisis matemáticos avanzados, respaldado por su conjunto de toolboxes especializados (MathWorks, s.f.). Aunque su aprendizaje puede no ser tan inmediato como el de Python, ofrecen estabilidad y precisión en contextos que lo requieren.

También me parece valioso rescatar la visión de Odersky et al. (2016) sobre Scala. Ellos destacan que este lenguaje combina programación funcional y orientada a objetos, lo cual lo hace especialmente útil en proyectos distribuidos y de alta concurrencia. Esta característica lo coloca en un lugar distinto dentro del ecosistema, más enfocado en el rendimiento que en la facilidad de uso, lo que complementa bien la oferta de lenguajes como Python o R.

Finalmente, la diversidad de lenguajes que analicé en este reporte me deja claro que no existe una sola respuesta correcta al momento de elegir. Como indican Guerrero (2011) y Ruiz Ruano (2016), incluso opciones de software libre como R han demostrado que el acceso abierto y la comunidad científica pueden sostener un lenguaje en el tiempo y mantenerlo vigente frente a alternativas más comerciales. Esta pluralidad de opciones enriquece el aprendizaje, porque me obliga a no encasillarme en un solo entorno, sino a evaluar con criterio cuál es la herramienta más adecuada para el problema que busco resolver.

REFERENCIAS Y FUENTES CONSULTADAS

Arcila Calderón, C., Barbosa Caro, E., y Cabezuelo Lorenzo, F. (2016). Técnicas big data: análisis de textos a gran escala para la investigación científica y periodística. *El Profesional de la Información*, 25(4), 623–631.

<https://doi.org/10.3145/epi.2016.jul.12>

Chancusig López, M. B., Yauli Chicaiza, G. E., López Castillo, G. de las M., Andrade Valencia, J. A., y López Velasco, J. E. (2024). *Estadística descriptiva aplicada en Python para la investigación científica en ciencias sociales y educativas*. Editorial SciELa. <https://doi.org/10.62131/978-9942-7221-6-4>

Chimarro Amaguaña, J. D. (2023). El auge exponencial del lenguaje Python en la ciencia de datos y el análisis financiero. *Revista Ingeniar*, 7(1), 45–53.

<https://journalingeniar.org/index.php/ingeniar/article/view/152>

Flórez Martínez, A., Vargas Flórez, J. O., Perez Waltero, H. E., y Quintana Fuentes, L. F. (2020). Análisis de componentes principales utilizando Python para identificar clúster asociados a muestras de cacao seco sano e infectado con monilia en Norte de Santander. *AiBi Revista de Investigación, Administración e Ingeniería*, 8(2), 16–22.

<https://doi.org/10.15649/2346030X.712>

Guerrero, F. B. (2011). R Project: su aplicación como software libre para análisis en investigación. *Revista de la Universidad de Costa Rica*. Recuperado de <https://revistas.ucr.ac.cr/index.php/aie/article/view/10231>

MathWorks. (n.d.). *Documentación oficial de MATLAB*. Recuperado el 23 de septiembre de 2025 de <https://www.mathworks.com/help/matlab/index.html>

Odersky, M., Spoon, L., y Venners, B. (2016). *Programming in Scala* (3a ed.). Artima Inc.

Rodríguez Rivas, J. G., y Rodríguez Castillo, S. (2022). Uso de Python para el análisis de datos aplicado en la investigación. *Investigación y Ciencia Aplicada a la Ingeniería*, 5(34), 33–40.
<https://www.ojsincaing.com.mx/index.php/ediciones/article/view/188>

Ruiz Ruano, A. M. (2016). R como entorno para el análisis estadístico. *Revista Redalyc*. Recuperado de
<https://www.redalyc.org/journal/778/77844204010/html/>

SAS Institute. (s.f.). *Documentación oficial de SAS para análisis de datos*. Recuperado el 23 de septiembre de 2025 de <https://documentation.sas.com>

Stonebraker, M., y Hellerstein, J. M. (2015). *What goes around comes around*. En *Readings in database systems* (5th ed.). MIT CSAIL. Recuperado de <http://www.redbook.io>

The Julia Project. (s.f.). *Documentación oficial de Julia*. Recuperado el 23 de septiembre de 2025 de <https://docs.julialang.org>