

**UNIVERSIDAD TECNOLÓGICA DE CHIHUAHUA**  
**TECNOLOGÍAS DE LA INFORMACIÓN**



**Extracción de Conocimiento en Bases de Datos**

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

***IDGS81N***

**PRESENTA:**

**JORGE ALEJANDRO HERNANDEZ CONTRERAS**

**DOCENTE:**

**Enrique Mascote**

Chihuahua, Chih., 9 de Septiembre de 2025

<b>Introducción.....</b>	<b>4</b>
<b>Importancia de conocer los lenguajes y bibliotecas.....</b>	<b>4</b>
<b>Objetivos del reporte.....</b>	<b>4</b>
<b>Secciones por lenguaje.....</b>	<b>5</b>
Python.....	5
Descripción general.....	5
Ámbito de uso principal.....	5
Bibliotecas y frameworks clave.....	5
Pandas:.....	5
NumPy:.....	5
Scikit-learn:.....	5
TensorFlow / PyTorch:.....	5
R.....	6
Descripción general.....	6
Ámbito de uso principal.....	6
Bibliotecas y frameworks clave.....	6
tidyverse:.....	6
caret:.....	6
shiny:.....	6
Scala con Apache Spark.....	6
Descripción general.....	6
Ámbito de uso principal.....	7
Bibliotecas y frameworks clave.....	7
Apache Spark:.....	7
Akka:.....	7
SQL.....	7
Descripción general.....	7
Ámbito de uso principal.....	7
Herramientas y variantes clave.....	7
PostgreSQL / MySQL / SQL Server:.....	7
BigQuery / Snowflake:.....	8
Funcionalidades importantes.....	8
Julia.....	8
Descripción general.....	8
Ámbito de uso principal.....	8
Bibliotecas y frameworks clave.....	8
DataFrames.jl:.....	8
CSV.jl:.....	8
Flux.jl:.....	9
Java.....	9
Descripción general.....	9
Ámbito de uso principal.....	9
Bibliotecas y frameworks clave.....	9
Weka:.....	9

Deeplearning4j:.....	9
Apache Spark:.....	9
<b>Conclusión.....</b>	<b>10</b>
<b>Bibliografía.....</b>	<b>11</b>

# Introducción

En este reporte se presentan los principales lenguajes de programación y sus bibliotecas o frameworks más usados en tareas de manipulación, análisis y modelado de datos en los ámbitos de Inteligencia Artificial, Machine Learning, Data Mining y Big Data. El objetivo es ofrecer una visión práctica y comparativa que sirva como guía para elegir herramientas adecuadas según el tipo de proyecto.

Conocer estos lenguajes y sus ecosistemas facilita seleccionar soluciones eficientes, aprovechar bibliotecas optimizadas, y comunicar resultados con el equipo de datos o con infraestructura de producción.

## Importancia de conocer los lenguajes y bibliotecas

Saber cuáles lenguajes y bibliotecas están disponibles permite:

- Acelerar el desarrollo usando soluciones ya probadas.
- Escalar modelos de manera que podamos elegir herramientas que soporten distribución o GPU.
- Garantizar reproducibilidad, manejando dependencias y versiones.
- Facilitar el trabajo en equipo usando estándares y APIs conocidas.

## Objetivos del reporte

1. Describir al menos cinco lenguajes relevantes para análisis de datos.
2. Enumerar de 2 a 3 bibliotecas clave por lenguaje, explicar su funcionalidad y un caso de uso.
3. Mostrar un ejemplo por lenguaje que cargue un CSV y muestre las primeras filas.
4. Comparar similitudes y diferencias y dar recomendaciones para distintos tipos de proyectos.

# Secciones por lenguaje

## Python

### Descripción general

Python es un lenguaje interpretado, de tipado dinámico y con múltiples paradigmas como imperativo, orientado a objetos, funcional. Es el lenguaje dominante en Data Science por su sintaxis simple y su enorme ecosistema.

### Ámbito de uso principal

Sus principales usos son para Data Science, prototipado de modelos ML/IA, back-end y orquestación de pipelines.

### Bibliotecas y frameworks clave

#### Pandas:

- Manipulación de datos tabulares y DataFrames.

Caso de uso:

- Carga, limpieza y transformación de datasets tabulares.

#### NumPy:

- Arrays numéricos y operaciones vectorizadas.

Caso de uso:

- Operaciones matriciales y preparación de features.

#### Scikit-learn:

- Modelos clásicos de ML, preprocesamiento y evaluación.

Caso de uso:

- Entrenamiento de modelos de clasificación/regresión rápida.

#### TensorFlow / PyTorch:

- Deep learning y computación en GPU.

Caso de uso:

-Entrenamiento de redes neuronales complejas.

## R

### Descripción general

R es un lenguaje interpretado orientado a análisis estadístico, tiene tipado dinámico y está especialmente diseñado para estadística y visualización.

### Ámbito de uso principal

Estadística, análisis exploratorio, visualización, investigación y modelos clásicos de ML.

### Bibliotecas y frameworks clave

tidyverse:

- Conjunto de paquetes para manipulación, lectura y visualización.

Caso de uso:

- Análisis exploratorio reproducible.

caret:

- Framework para entrenamiento y comparación de modelos ML clásicos.

Caso de uso:

- Pipeline de modelado y selección de hiperparámetros.

shiny:

- Para crear apps web interactivas que muestran análisis y modelos.

## Scala con Apache Spark

### Descripción general

Scala es un lenguaje compilado a JVM y con múltiples paradigmas como funcional y orientado a objetos, con tipado estático. En Big Data se usa mayormente junto con Apache Spark.

## Ámbito de uso principal

Big Data distribuido, pipelines ETL, procesamiento en clúster y aplicaciones de alto rendimiento sobre JVM.

## Bibliotecas y frameworks clave

Apache Spark:

- Procesamiento distribuido de datos, DataFrames y librería ML para algoritmos escalables.

Caso de uso:

- Procesamiento de terabytes y entrenamiento distribuido.

Akka:

- Concurrencia y sistemas distribuidos es menos específico de datos, pero útil en infraestructuras de streaming.

## SQL

### Descripción general

SQL es un lenguaje declarativo para manipulación y consulta de bases de datos relacionales. Es esencial en análisis de datos.

## Ámbito de uso principal

Consultas, agregaciones, transformación de datos almacenados en RDBMS y Data Warehouses.

## Herramientas y variantes clave

PostgreSQL / MySQL / SQL Server:

- Sistemas RDBMS para almacenamiento y consultas.

Caso de uso:

- Ejecutar consultas analíticas sobre bases de datos empresariales.

BigQuery / Snowflake:

- Warehouses en la nube para análisis a escala.

Caso de uso:

- Análisis de grandes volúmenes de datos en entornos empresariales y en la nube.

## Funcionalidades importantes

JOINS, agregaciones, CTES, COPY para importar CSV.

## Julia

### Descripción general

Julia es un lenguaje de alto rendimiento diseñado para cómputo numérico y científico; compila JIT y tiene tipado dinámico con capacidad para tipado estático opcional.

### Ámbito de uso principal

Cómputo numérico intensivo, prototipado de algoritmos científicos y ML cuando se necesita velocidad cercana a C/Fortran.

### Bibliotecas y frameworks clave

DataFrames.jl:

- Estructuras tipo DataFrame para manipulación tabular.

Caso de uso:

- Exploración y limpieza de datos similares a pandas en Python.

CSV.jl:

- Lectura y escritura de CSV.

Caso de uso:

- Carga eficiente de archivos CSV grandes.

Flux.jl:

- Librería de machine learning/ deep learning en Julia.

Caso de uso:

- Construcción y entrenamiento de redes neuronales en proyectos de investigación.

## Java

### Descripción general

Java es un lenguaje compilado a bytecode sobre la JVM, con tipado estático. Es robusto para aplicaciones de producción y compatible con ecosistemas de Big Data.

### Ámbito de uso principal

Aplicaciones backend, integración en sistemas de producción, herramientas de Big Data.

### Bibliotecas y frameworks clave

Weka:

- Biblioteca y entorno para aprendizaje automático clásico, útil en contexto educativo/prototipos en Java.

Caso de uso:

- Experimentación rápida con algoritmos ML en entornos académicos o educativos.

Deeplearning4j:

- Framework de deep learning para JVM que soporta GPU a través de ND4J.

Caso de uso:

- Despliegue de modelos de deep learning en sistemas Java empresariales.

Apache Spark:

- Para procesamiento distribuido en entornos Java.

Caso de uso:

-Integración de pipelines Big Data en sistemas empresariales ya basados en Java.

## Conclusión

Los lenguajes cubiertos muestran distintos balances entre facilidad de uso, rendimiento y madurez del ecosistema:

Facilidad de uso:

Python y R son los más accesibles para análisis exploratorio y prototipos por su sintaxis y abundancia de tutoriales y paquetes. R destaca para análisis estadístico profundo y visualización; Python para integración con aplicaciones y deep learning.

Rendimiento:

Julia y Scala con Spark ofrecen mejor rendimiento en escenarios numéricos intensivos o datos a escala cuando se usan adecuadamente. Java y Scala son preferibles en sistemas que requieren alta estabilidad y despliegue en JVM.

Ecosistema:

Python tiene el ecosistema más amplio. R tiene un ecosistema sólido en estadísticas y visualización. Spark domina en procesamiento distribuido.

Al comparar todos estos lenguajes me doy cuenta de que no existe uno “mejor” en todo, sino que depende mucho del proyecto y de lo que se busque: unos son más fáciles de aprender como Python o R, otros son más rápidos y potentes como Scala o Julia y algunos se usan más en empresas grandes porque ya están integrados con sus sistemas como Java o SQL. En conclusión, lo importante no es saberse todos al 100%, sino conocer sus ventajas y limitaciones para elegir el adecuado según la situación, y creo que como estudiantes debemos practicar con varios para tener más opciones en el futuro.

# Bibliografía

pandas.read\_csv — pandas documentation.

[https://pandas.pydata.org/docs/reference/api/pandas.read\\_csv.html](https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html)

pandas.DataFrame.head — pandas documentation.

<https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.head.html>

scikit-learn — Getting Started. [https://scikit-learn.org/stable/getting\\_started.html](https://scikit-learn.org/stable/getting_started.html)

NumPy — Official documentation. <https://numpy.org/>

readr (tidyverse) — <https://readr.tidyverse.org/>

tidyverse packages — <https://www.tidyverse.org/packages/>

caret R package — <https://topepo.github.io/caret/>

Apache Spark CSV — <https://spark.apache.org/docs/latest/sql-data-sources-csv.html>

Spark DataFrameReader (JavaDoc) —

<https://spark.apache.org/docs/3.5.4/api/java/org/apache/spark/sql/DataFrameReader.html>

PostgreSQL COPY — <https://www.postgresql.org/docs/current/sql-copy.html>