Actividad 2 Base de datos Oracle

**Alumnos:** Restrepo Rosero Juan Piñeros Castro Carlos

Doria Atencia Joel

## Docente:

Sierra Galvis Martín Vladimir Alonso

Gestión de datos Maestría en ciencia de datos Universidad Javeriana de Cali

Octubre, 2024



## Contexto

Imaginen que determinada Universidad desarrolló una plataforma educativa que contiene un gran número de funcionalidades, entre las que se encuentra un módulo de registro académico. Gracias a este módulo la Universidad ha podido almacenar gran cantidad de datos relacionados con carreras, cursos, profesores y estudiantes. Actualmente los directivos se encuentran interesados en llevar a cabo procesos de análisis que les permita visualizar ciertas tendencias y cuestiones académicas, por lo que requieren del conocimiento de un científico de datos para realizar esta labor.

## Consultas

Con el objetivo de seleccionar la información pertinente para los procesos de análisis que requiere la Universidad los directivos nos piden efectuar las siguientes consultas en SQL sobre los datos del módulo de registro académico:

1. Listar todos los cursos ofrecidos en la carrera de Ingeniería de Sistemas. Para este caso se debe tener en cuenta que un mismo curso puede ser dictado por profesores diferentes, en salones diferentes y en horarios diferentes. Se deben obtener los siguientes datos: el nombre del curso, el nombre completo del profesor que dicta el curso, el salón en que se dicta el curso y la hora en la que se dicta del curso.

Sentencia SQL:

### SELECT

ca.nombre AS nombre\_carrera, cc.semestre, cal.ID\_CURSO,

c.nombre AS nombre\_curso, cal.dia AS dia,

cal.hora\_inicio, cal.hora\_fin,

u.nombre || ' ' || u.apellido AS nombre\_completo\_profesor, s.nombre AS salon

### FROM

Cursos c

INNER JOIN Cursos\_Carreras cc ON c.id\_curso = cc.id\_curso INNER JOIN Calendario\_Cursos cal ON cc.id\_curso = cal.id\_curso INNER JOIN Salones s ON cal.id\_salon = s.id\_salon

INNER JOIN Usuarios u ON cal.id\_profesor = u.id\_usuario

INNER JOIN Carreras ca ON cc.ID\_CARRERA = ca.ID\_CARRERA WHERE

-- cc.id\_carrera = (SELECT cc.ID\_CARRERA FROM Carreras WHERE nombre = 'Ingenieria de Sistemas') cc.id\_carrera = 1

### ORDER BY

ca.nombre, c.nombre, cal.hora\_inicio;

### RESULTADO:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NOMBRE\_CA RRERA | SEME STRE | ID\_CURSO | NOMBRE\_CURS O | DIA | HORA\_INICIO | HORA\_FI N | NOMBRE\_COMPLETO\_P ROFESOR | SAL ON |
| **Ingeniería de Sistemas** | 7 | 20 | Aprendizaje Automático | Martes | 4:00 p. m. | 6:00  p. m. | Robert Vance | 1.1 |
| **Ingeniería de Sistemas** | 7 | 20 | Aprendizaje Automático | Jueves | 4:00 p. m. | 6:00  p. m. | Robert Vance | 1.1 |
| **Ingeniería de Sistemas** | 4 | 10 | Arquitectura del Computador | Lunes | 4:00 p. m. | 6:00  p. m. | Grace Stone | 2.6 |
| **Ingeniería de Sistemas** | 4 | 10 | Arquitectura del Computador | Viernes | 4:00 p. m. | 6:00  p. m. | Grace Stone | 2.6 |
| **Ingeniería de Sistemas** | 4 | 28 | Cinemática y Dinámica | Miércoles | 2:00 p. m. | 4:00  p. m. | Saanvi Bahl | 2.2 |
| **Ingeniería de Sistemas** | 4 | 28 | Cinemática y Dinámica | Lunes | 2:00 p. m. | 4:00  p. m. | Saanvi Bahl | 2.2 |
| **Ingeniería de Sistemas** | 4 | 28 | Cinemática y Dinámica | Viernes | 2:00 p. m. | 4:00  p. m. | Saanvi Bahl | 2.2 |
| **Ingeniería de Sistemas** | 1 | 44 | Cálculo Diferencial | Jueves | 8:00 a. m. | 10:00  a. m. | Grace Stone | 2.5 |
| **Ingeniería de Sistemas** | 1 | 44 | Cálculo Diferencial | Martes | 8:00 a. m. | 10:00  a. m. | Grace Stone | 2.5 |
| **Ingeniería de Sistemas** | 2 | 47 | Cálculo Integral | Martes | 10:00 a. m. | 12:00  p. m. | Jared Vï¿½squez | 3.3 |
| **Ingeniería de Sistemas** | 2 | 47 | Cálculo Integral | Jueves | 10:00 a. m. | 12:00  p. m. | Jared Vï¿½squez | 3.3 |
| **Ingeniería de Sistemas** | 3 | 48 | Cálculo Multivariado | Jueves | 2:00 p. m. | 4:00  p. m. | Jared Vï¿½squez | 2.1 |
| **Ingeniería de Sistemas** | 3 | 48 | Cálculo Multivariado | Martes | 2:00 p. m. | 4:00  p. m. | Jared Vï¿½squez | 2.1 |
| **Ingeniería de Sistemas** | 9 | 26 | Desarrollo de Videouegos | Martes | 2:00 p. m. | 5:00  p. m. | Ben Stone | 3.2 |
| **Ingeniería de Sistemas** | 9 | 26 | Desarrollo de Videouegos | Viernes | 2:00 p. m. | 5:00  p. m. | Ben Stone | 3.2 |
| **Ingeniería de Sistemas** | 5 | 34 | Electricidad y Magnetismo | Miércoles | 4:00 p. m. | 6:00  p. m. | Saanvi Bahl | 1.5 |
| **Ingeniería de Sistemas** | 5 | 34 | Electricidad y Magnetismo | Lunes | 4:00 p. m. | 6:00  p. m. | Saanvi Bahl | 1.5 |
| **Ingeniería de Sistemas** | 5 | 34 | Electricidad y Magnetismo | Viernes | 4:00 p. m. | 6:00  p. m. | Saanvi Bahl | 1.5 |
| **Ingeniería de Sistemas** | 2 | 3 | Estructuras de Datos | Jueves | 10:00 a. m. | 12:00  p. m. | Ben Stone | 3.4 |
| **Ingeniería de Sistemas** | 2 | 3 | Estructuras de Datos | Martes | 10:00 a. m. | 12:00  p. m. | Ben Stone | 3.4 |
| **Ingeniería de Sistemas** | 1 | 1 | Introducción a la Programación | Martes | 8:00 a. m. | 10:00  a. m. | Ben Stone | 1.4 |
| **Ingeniería de Sistemas** | 1 | 1 | Introducción a la Programación | Jueves | 8:00 a. m. | 10:00  a. m. | Ben Stone | 1.4 |
| **Ingeniería de Sistemas** | 8 | 22 | Introducción a la Seguridad  Informática | Miércoles | 4:00 p. m. | 6:00  p. m. | Robert Vance | 3.5 |
| **Ingeniería de Sistemas** | 8 | 22 | Introducción a la  Seguridad Informática | Viernes | 4:00 p. m. | 6:00  p. m. | Robert Vance | 3.5 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Ingeniería de Sistemas** | 3 | 6 | Lógica Digital y Lenguaje de Máquina | Viernes | 2:00 p. m. | 4:00  p. m. | Grace Stone | 1.2 |
| **Ingeniería de Sistemas** | 3 | 6 | Lógica Digital y Lenguaje de Máquina | Miércoles | 2:00 p. m. | 4:00  p. m. | Grace Stone | 1.2 |
| **Ingeniería de Sistemas** | 1 | 53 | Optimización Matemática | Miércoles | 2:00 p. m. | 4:00  p. m. | Jared Vï¿½squez | 2.2 |
| **Ingeniería de Sistemas** | 1 | 53 | Optimización Matemática | Lunes | 2:00 p. m. | 4:00  p. m. | Jared Vï¿½squez | 2.2 |
| **Ingeniería de Sistemas** | 1 | 53 | Optimización Matemática | Miércoles | 8:00 a. m. | 10:00  a. m. | Ulrich Nielsen | 1.3 |
| **Ingeniería de Sistemas** | 1 | 53 | Optimización Matemática | Viernes | 8:00 a. m. | 10:00  a. m. | Ulrich Nielsen | 1.3 |
| **Ingeniería de Sistemas** | 7 | 19 | Programación Paralela | Jueves | 8:00 a. m. | 10:00  a. m. | Ulrich Nielsen | 3.3 |
| **Ingeniería de Sistemas** | 7 | 19 | Programación Paralela | Martes | 8:00 a. m. | 10:00  a. m. | Ulrich Nielsen | 3.3 |
| **Ingeniería de Sistemas** | 10 | 71 | Práctica Estudiantil | Viernes | 2:00 p. m. | 4:00  p. m. | Jared Vï¿½squez | 1.6 |
| **Ingeniería de Sistemas** | 8 | 24 | Sistemas Inteligentes | Lunes | 10:00 a. m. | 12:00  p. m. | Robert Vance | 2.2 |
| **Ingeniería de Sistemas** | 8 | 24 | Sistemas Inteligentes | Miércoles | 10:00 a. m. | 12:00  p. m. | Robert Vance | 2.2 |
| **Ingeniería de Sistemas** | 2 | 57 | Teología | Lunes | 10:00 a. m. | 12:00  p. m. | Zeke Landon | 1.8 |
| **Ingeniería de Sistemas** | 2 | 4 | Técnicas y Prácticas de Programación | Miércoles | 4:00 p. m. | 6:00  p. m. | Ben Stone | 3.4 |
| **Ingeniería de Sistemas** | 2 | 4 | Técnicas y  Prácticas de Programación | Lunes | 4:00 p. m. | 6:00  p. m. | Ben Stone | 3.4 |
| **Ingeniería de Sistemas** | 2 | 45 | Álgebra Lineal | Jueves | 10:00 a. m. | 12:00  p. m. | Grace Stone | 3.1 |
| **Ingeniería de Sistemas** | 2 | 45 | Álgebra Lineal | Martes | 10:00 a. m. | 12:00  p. m. | Grace Stone | 3.1 |
| **Ingeniería de Sistemas** | 9 | 67 | Ética | Miércoles | 8:00 a. m. | 10:00  a. m. | Mikaela Stone | 3.1 |
| **Ingeniería de Sistemas** | 9 | 67 | Ética | Viernes | 8:00 a. m. | 10:00  a. m. | Jared Vï¿½squez | 3.7 |

1. Obtener la lista de profesores que dictan cursos pertenecientes a la facultad de

Humanidades. Se deben obtener los siguientes datos: el nombre completo del profesor y el nombre del curso que dicta.

### SENTENCIA SQL:

SELECT

f.NOMBRE AS Facultad, c.nombre AS nombre\_curso,

u.nombre || ' ' || u.apellido AS nombre\_completo\_profesor

### FROM

Usuarios u

INNER JOIN Calendario\_Cursos cal ON u.ID\_USUARIO = cal.id\_profesor INNER JOIN Cursos c ON cal.id\_curso = c.ID\_CURSO

INNER JOIN Facultades f ON c.id\_facultad = f.id\_facultad WHERE

f.ID\_FACULTAD=2

-- f.nombre LIKE '%Humanidades%' ORDER BY

u.nombre, u.apellido, c.nombre;

# RESULTADO:

|  |  |  |
| --- | --- | --- |
| FACULTAD | NOMBRE\_CURSO | NOMBRE\_COMPLETO\_PROFESOR |
| **Humanidades y Ciencias Sociales** | Ética | Jared Vásquez |
| **Humanidades y Ciencias Sociales** | Ética | Mikaela Stone |
| **Humanidades y Ciencias Sociales** | Antropología Filosófica | Zeke Landon |
| **Humanidades y Ciencias Sociales** | Filosofía Antigua | Zeke Landon |
| **Humanidades y Ciencias Sociales** | Filosofía de la Ciencia | Zeke Landon |
| **Humanidades y Ciencias Sociales** | Teología | Zeke Landon |

1. Obtener la lista de profesores que dictan cursos en dos carreras diferentes. Se deben obtener los siguientes datos: el nombre completo del profesor, el nombre del curso que dicta y el nombre de la carrera a la que pertenece el curso. Pista: averigua cómo funciona la sentencia WITH.

### SENTENCIA SQL:

WITH Profesores\_Multiples\_Carreras AS ( SELECT

u.ID\_USUARIO AS id\_profesor, u.nombre || ' ' || u.apellido AS

nombre\_completo\_profesor FROM Usuarios u

INNER JOIN Calendario\_Cursos cal ON u.ID\_USUARIO = cal.id\_profesor INNER JOIN Cursos\_Carreras cc ON cal.id\_curso = cc.id\_curso

### GROUP BY

u.ID\_USUARIO, u.nombre, u.apellido HAVING

COUNT(DISTINCT cc.id\_carrera) = 2

)

### SELECT DISTINCT

pmc.nombre\_completo\_profesor, c.ID\_CURSO AS id\_curso,

c.nombre AS nombre\_curso, ca.nombre AS nombre\_carrera FROM

Profesores\_Multiples\_Carreras pmc

INNER JOIN Calendario\_Cursos cal ON pmc.id\_profesor = cal.id\_profesor INNER JOIN Cursos c ON cal.id\_curso = c.id\_curso

INNER JOIN Cursos\_Carreras cc ON cal.id\_curso = cc.id\_curso INNER JOIN Carreras ca ON cc.id\_carrera = ca.ID\_CARRERA ORDER BY

pmc.nombre\_completo\_profesor,id\_curso;

# RESULTADO:

|  |  |  |  |
| --- | --- | --- | --- |
| NOMBRE\_COMPLETO\_PROFESOR | ID\_CURSO | NOMBRE\_CURSO | NOMBRE\_CARRERA |
| **Ben Stone** | 1 | Introducción a la Programación | Ingenieria de Sistemas |
| **Ben Stone** | 3 | Estructuras de Datos | Ingenieria de Sistemas |
| **Ben Stone** | 3 | Estructuras de Datos | Matemáticas Aplicadas |
| **Ben Stone** | 4 | Técnicas y Prácticas de Programación | Ingenieria de Sistemas |
| **Ben Stone** | 26 | Desarrollo de Videouegos | Ingenieria de Sistemas |
| **Claudia Tiedemann** | 29 | Química y Ciencia de Materiales | Ingeniería Mecánica |
| **Claudia Tiedemann** | 32 | Propiedades de los Materiales | Ingeniería Mecánica |
| **Claudia Tiedemann** | 39 | Diseño Mecánico | Ingeniería Mecánica |
| **Claudia Tiedemann** | 40 | Máquinas Térmicas e Hidráulicas | Ingeniería Mecánica |
| **Claudia Tiedemann** | 49 | Álgebra Moderna | Matemáticas Aplicadas |
| **Ulrich Nielsen** | 19 | Programación Paralela | Ingenieria de Sistemas |
| **Ulrich Nielsen** | 53 | Optimización Matemática | Ingenieria de Sistemas |
| **Ulrich Nielsen** | 53 | Optimización Matemática | Matemáticas Aplicadas |

1. Para un estudiante en particular, obtener el listado de cursos que puede matricular. Para este caso particular se debe tener en cuenta que los cursos se dictan en semestres diferentes, pertenecen a carreras diferentes y un estudiante no puede matricular un curso que ya se encuentre matriculado. Se deben obtener los siguientes datos: el nombre del curso, el nombre de la carrera a la que pertenece y el semestre en el que se ubica. Pista: averigua el uso de la sentencia NOT EXISTS.

### SENTENCIA SQL:

WITH CarrerasEstudiante AS ( SELECT id\_carrera,id\_usuario FROM Carreras\_Estudiantes WHERE id\_usuario = :id\_estudiante

),

CursosMatriculados AS ( SELECT cal.id\_curso

FROM Cursos\_Estudiantes ce

INNER JOIN Calendario\_Cursos cal ON ce.ID\_CALENDARIO = cal.ID\_CALENDARIO WHERE ce.id\_usuario = :id\_estudiante

### ) SELECT

u.nombre || ' ' || u.apellido AS nombre\_estudiante, cc.ID\_CURSO,

c.nombre AS nombre\_curso, ca.nombre AS nombre\_carrera, cc.semestre

### FROM

usuarios u

INNER JOIN CarrerasEstudiante ce ON u.id\_usuario = ce.id\_usuario INNER JOIN Carreras ca ON ce.id\_carrera = ca.id\_carrera

INNER JOIN Cursos\_Carreras cc ON ca.id\_carrera = cc.id\_carrera INNER JOIN Cursos c ON cc.id\_curso = c.id\_curso

### WHERE

NOT EXISTS (

SELECT 1 FROM CursosMatriculados cm WHERE c.id\_curso = cm.id\_curso

)

AND u.id\_usuario = :id\_estudiante order by cc.SEMESTRE;

# RESULTADO:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID\_ESTUDIANT E | NOMBRE\_ESTUDIANT E | ID\_CURS O | NOMBRE\_CURSO | NOMBRE\_CARRERA | SEMESTR E |
| **17** | Franziska Doppler | 2 | Introducción al Modelado de Sistemas | Ingeniería de Sistemas | 1 |
| **17** | Franziska Doppler | 53 | Optimización Matemática | Ingeniería de Sistemas | 1 |
| **17** | Franziska Doppler | 57 | Teología | Ingeniería de Sistemas | 2 |
| **17** | Franziska Doppler | 7 | Programación Funcional | Ingeniería de Sistemas | 3 |
| **17** | Franziska Doppler | 5 | Programación Orientada a Objetos | Ingeniería de Sistemas | 3 |
| **17** | Franziska Doppler | 8 | Lógica para Ciencias de la Computación | Ingeniería de Sistemas | 4 |
| **17** | Franziska Doppler | 9 | Arboles y Grafos | Ingeniería de Sistemas | 4 |
| **17** | Franziska Doppler | 11 | Diseño de Interfaces HumanoComputador | Ingeniería de Sistemas | 4 |
| **17** | Franziska Doppler | 12 | Computabilidad y Complejidad | Ingeniería de Sistemas | 5 |
| **17** | Franziska Doppler | 34 | Electricidad y Magnetismo | Ingeniería de Sistemas | 5 |
| **17** | Franziska Doppler | 14 | Comunicación de Datos | Ingeniería de Sistemas | 5 |
| **17** | Franziska Doppler | 13 | Computación Gráfica | Ingeniería de Sistemas | 5 |
| **17** | Franziska Doppler | 16 | Análisis y Diseño de Algoritmos | Ingeniería de Sistemas | 6 |
| **17** | Franziska Doppler | 18 | Sistemas Operativos | Ingeniería de Sistemas | 6 |
| **17** | Franziska Doppler | 15 | Desarrollo Formal de Sistemas | Ingeniería de Sistemas | 6 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **17** | Franziska Doppler | 17 | Gestión y Modelación de Datos | Ingeniería de Sistemas | 6 |
| **17** | Franziska Doppler | 21 | Animación y Simulación | Ingeniería de Sistemas | 7 |
| **17** | Franziska Doppler | 19 | Programación Paralela | Ingeniería de Sistemas | 7 |
| **17** | Franziska Doppler | 20 | Aprendizaje Automático | Ingeniería de Sistemas | 7 |
| **17** | Franziska Doppler | 22 | Introducción a la Seguridad Informática | Ingeniería de Sistemas | 8 |
| **17** | Franziska Doppler | 24 | Sistemas Inteligentes | Ingeniería de Sistemas | 8 |
| **17** | Franziska Doppler | 23 | Construcción de Software y Pruebas | Ingeniería de Sistemas | 8 |
| **17** | Franziska Doppler | 26 | Desarrollo de Videouegos | Ingeniería de Sistemas | 9 |
| **17** | Franziska Doppler | 70 | Trabajo de Grado | Ingeniería de Sistemas | 9 |
| **17** | Franziska Doppler | 25 | Computación en la Nube | Ingeniería de Sistemas | 9 |
| **17** | Franziska Doppler | 67 | Ética | Ingeniería de Sistemas | 9 |
| **17** | Franziska Doppler | 71 | Práctica Estudiantil | Ingeniería de Sistemas | 10 |

1. Listar los estudiantes que se han inscrito a un curso determinado. Se deben obtener los siguientes datos: el nombre completo del estudiante y el nombre del curso.

### SENTENCIA SQL:

SELECT distinct cc.id\_curso,

c.nombre AS nombre\_curso,

u.nombre || ' ' || u.apellido AS nombre\_estudiante FROM

USUARIOS u

INNER JOIN Cursos\_Estudiantes ce ON u.id\_usuario = ce.id\_usuario INNER JOIN Calendario\_Cursos cc ON ce.id\_calendario = cc.id\_calendario INNER JOIN Cursos c ON cc.id\_curso = c.id\_curso WHERE

c.id\_curso= :idcurso order by nombre\_estudiante;

# RESULTADO:

|  |  |  |
| --- | --- | --- |
| **ID\_CURSO** | **NOMBRE\_CURSO** | **NOMBRE\_ESTUDIANTE** |
| **57** | Teología | Angelina Meyer |
| **57** | Teología | Bartosz Tiedemann |
| **57** | Teología | Jonas Kahnwald |
| **57** | Teología | Magnus Nielsen |

|  |  |  |
| --- | --- | --- |
| **57** | Teología | Martha Nielsen |
| **57** | Teología | Olive Stone |

## Conclusiones:

Las consultas SQL son una herramienta fundamental en el mundo de las bases de datos relacionales. Su sintaxis, aunque pueda parecer compleja al principio, permite a los usuarios interactuar de manera eficiente y precisa con la información almacenada en bases de datos.

Ventajas de las consultas SQL:

**Flexibilidad**: Permiten realizar una amplia variedad de operaciones, desde simples selecciones hasta complejas uniones y agrupaciones.

**Potencia**: Pueden manipular grandes volúmenes de datos de forma rápida y eficiente.

**Estandarización**: El lenguaje SQL es un estándar ampliamente adoptado, lo que facilita su uso en diferentes sistemas de gestión de bases de datos.

**Facilidad de aprendizaje**: Aunque tiene su propia sintaxis, SQL es relativamente fácil de aprender y su lógica es intuitiva para quienes están familiarizados con la programación.

A lo largo de esta actividad, se ha demostrado la capacidad de SQL para gestionar bases de datos académicas complejas, proporcionando resultados eficientes y adaptados a las necesidades de análisis de los directivos universitarios. El uso de consultas con múltiples tablas y la implementación de cláusulas como WITH no solo han facilitado la legibilidad del código, sino que también han optimizado la eficiencia de las consultas.

**Uso de WITH para simplificación**: En las consultas que requieren manipulación de datos complejos, la utilización de subconsultas comunes mediante la sentencia WITH permite estructurar de forma más clara los pasos intermedios. Esto fue especialmente útil en la consulta que buscaba identificar profesores que imparten clases en diferentes carreras, permitiendo agrupar y filtrar los datos de manera eficaz.

**Consultas con múltiples JOIN**: El uso de combinaciones de tablas (JOIN) es esencial en una base de datos relacional como la de este proyecto. En particular, las consultas que involucraban la relación entre cursos, profesores y carreras dependían de múltiples uniones de tablas para proporcionar resultados útiles y completos, como se observó en el caso de la consulta sobre los cursos de la carrera de **Ingeniería de Sistemas**.

**Condiciones avanzadas como NOT EXISTS**: Esta cláusula ayudó a filtrar cursos que un estudiante aún no había matriculado, asegurando que los resultados sean precisos y sin duplicados. Esto resultó crítico en consultas donde era necesario eliminar posibilidades que ya habían sido registradas.

En resumen, las consultas SQL no solo son una herramienta potente para la gestión y análisis de datos, sino que también son altamente adaptables a las necesidades del negocio o del análisis académico. A través del uso de cláusulas avanzadas como WITH y NOT EXISTS, además de la capacidad de combinar datos de múltiples tablas mediante JOIN, se puede optimizar el rendimiento de las consultas y proporcionar información de alta calidad a quienes toman decisiones.

## Referencias:

1. «¿Qué es SQL? - Explicación de lenguaje de consulta estructurado (SQL) - AWS», Amazon Web Services, Inc. Accedido: 4 de octubre de 2024. [En línea]. Disponible en: https://aws.amazon.com/es/what-is/sql/
2. «Oracle Database SQL Language Reference, 23ai». Accedido: 4 de octubre de 2024. [En línea].

Disponible en: https://docs.oracle.com/en/database/oracle/oracle-database/23/sqlrf/index.html

1. «Database SQL Language Reference». Accedido: 4 de octubre de 2024. [En línea]. Disponible en: https://docs.oracle.com/cd/B28359\_01/server.111/b28286/toc.htm
2. «SQL», *Wikipedia, la enciclopedia libre*. 18 de junio de 2024. Accedido: 4 de octubre de 2024. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=SQL&oldid=160817572

