Actividad de cartera: Aplica filtros a consultas SQL

Escenario

Analiza el siguiente caso. Luego, completa las instrucciones paso a paso.
Eres un/a profesional de la seguridad en una gran organización. Parte de tu trabajo consiste en investigar los problemas de seguridad para ayudar a mantener el sistema seguro.
Recientemente descubriste algunos potenciales problemas de seguridad relacionados con los intentos de inicio de sesión y las máquinas de los/las empleados/as.
Tu tarea consiste en examinar los datos de la organización en sus tablas **`employees`** y **`log_in_attempts`**. Tendrás que usar filtros SQL para recuperar registros de diferentes conjuntos de datos e investigar los posibles problemas de seguridad.
Nota: Este escenario involucra las mismas consultas que las del lab Filtra con AND, OR y NOT. Puedes repasar el lab para obtener capturas de pantalla e incluirlas en el documento de tu cartera. Si lo deseas, también puedes completar esta actividad sin volver a consultar el lab escribiendo tus consultas en la plantilla.

---

1. Recuperar intentos fallidos de inicio de sesión fuera del horario laboral

Para identificar intentos de inicio de sesión fallidos fuera del horario laboral (por ejemplo, fuera de 9:00 a.m. a 6:00 p.m.), puedes usar esta consulta SQL. Aquí consideramos que el horario laboral es de 09:00 a 18:00 (6 p.m.).

```
SELECT event_id, username, login_date, login_time, country, ip_address
FROM log_in_attempts
WHERE success = FALSE
AND (login_time < '09:00:00' OR login_time > '18:00:00');
```

Explicación:

- Se filtran los intentos de inicio de sesión fallidos (`success = FALSE`).
- Usamos la condición `OR` para identificar los intentos fuera del rango de horas laborales (antes de las 9:00 a.m. o después de las 6:00 p.m.).

---

2. Recuperar intentos de inicio de sesión en fechas específicas

Si necesitas obtener los intentos de inicio de sesión en fechas específicas, puedes usar la siguiente consulta. Por ejemplo, para buscar los intentos realizados el 15 de octubre de 2023 y el 1 de noviembre de 2023:

```
SELECT event_id, username, login_date, login_time, country,
ip_address, success
FROM log_in_attempts
WHERE login_date IN ('2023-10-15', '2023-11-01');
```

Explicación:

- La cláusula `IN` permite especificar múltiples fechas, lo que simplifica la búsqueda de intentos en fechas específicas.

---

3. Recuperar intentos de inicio de sesión fuera de México

Para encontrar intentos de inicio de sesión que no se realizaron en México:

```
SELECT event_id, username, login_date, login_time, country,
ip_address, success
FROM log_in_attempts
WHERE country != 'Mexico';
```

Explicación:

- Usamos `!=` para excluir los registros donde el país es México, es decir, buscamos intentos realizados fuera de México.

---

4. Recuperar empleados del departamento de Marketing

Para recuperar los empleados del departamento de Marketing, asumiendo que la tabla `employees` tiene una columna llamada `department`:

```
SELECT employee_id, name, department
FROM employees
WHERE department = 'Marketing';
```

Explicación:

- La consulta filtra todos los empleados cuyo departamento es "Marketing".

---

5. Recuperar empleados de Finanzas o Ventas

Para recuperar los empleados que pertenecen a los departamentos de Finanzas o Ventas:

```
SELECT employee_id, name, department
FROM employees
WHERE department IN ('Finanzas', 'Ventas');
```

Explicación:

- Usamos la cláusula `IN` para buscar empleados en cualquiera de los dos departamentos, lo que simplifica la consulta.

---

6. Recuperar todos los empleados que no pertenecen al departamento de IT

Para recuperar empleados que no pertenecen al departamento de IT:

```
SELECT employee_id, name, department
FROM employees
WHERE department != 'IT';
```

Explicación:

- Usamos el operador `!=` para excluir a los empleados del departamento de IT.

---

7. Resumen

En este proyecto, hemos usado consultas SQL para filtrar diferentes conjuntos de datos con el fin de investigar problemas de seguridad relacionados con los intentos de inicio de sesión y la ubicación de los empleados. Hemos aplicado operadores como `AND`, `OR`, `NOT`, y utilizado filtros por fecha, hora, y patrones para obtener información relevante.

Estos filtros ayudan a identificar comportamientos inusuales, como intentos de inicio de sesión fallidos fuera del horario laboral, accesos desde ubicaciones no autorizadas, y la identificación de empleados en diferentes departamentos. Las consultas SQL nos permiten analizar grandes volúmenes de datos de forma eficiente, contribuyendo a mantener la seguridad de la organización.

Detalles adicionales sobre consultas SQL:

Uso de LIKE para buscar un patrón:
El operador `LIKE` se usa para buscar coincidencias parciales dentro de una columna. Por ejemplo, si queremos buscar empleados cuyo nombre comienza con "A", podemos usar:

```
SELECT * FROM employees WHERE name LIKE 'A%';
```

Filtrar por fechas y horas:
Las consultas pueden incluir filtros por fechas y horas usando operadores como `>=`, `<=`, o rangos específicos:

```
SELECT * FROM log_in_attempts WHERE login_date BETWEEN '2023-10-01'
AND '2023-10-31';
```

Uso de AND y OR para múltiples condiciones:
Las combinaciones de `AND` y `OR` permiten aplicar múltiples condiciones:

```
SELECT * FROM employees WHERE department = 'Ventas' AND country =
'USA';
```

Uso de NOT para exclusiones:
El operador `NOT` es útil para excluir ciertos resultados:

```
SELECT * FROM log_in_attempts WHERE NOT success = TRUE;
```