Guía completa: DynamoDB con PartiQL en NoSQL Workbench

1. Crear una tabla en DynamoDB

Nota: La creación de tablas no se puede hacer con PartiQL. Debes usar la interfaz gráfica de NoSQL Workbench:

1. Ve a **"Data modeler"**
2. Crea un nuevo modelo llamado "Usuarios"
3. Agrega una tabla con:
   - **Nombre:** Usuarios
   - **Partition key:** UsuarioID (String)
   - **Capacidad:** Read=5, Write=5
4. Haz "Commit" a tu conexión local

2. Insertar datos (put-item)

Comando AWS CLI original:

```bash
aws dynamodb put-item \
--table-name Usuarios \
--item '{"UsuarioID": {"S": "12345"}, "Nombre": {"S": "Carlos Pérez"}, "Edad": {"N": "30"}}'
```

Equivalente en PartiQL:

```sql
INSERT INTO Usuarios VALUE {
    'UsuarioID': '12345',
    'Nombre': 'Carlos Pérez',
    'Edad': 30
}
```

Insertar múltiples usuarios:

```sql
```

```
INSERT INTO Usuarios VALUE {
    'UsuarioID': '67890',
    'Nombre': 'Ana García',
    'Edad': 25
}
```

sql

```
INSERT INTO Usuarios VALUE {
    'UsuarioID': '11111',
    'Nombre': 'Luis Martínez',
    'Edad': 35
}
```

3. Consultar datos específicos (query)

Comando AWS CLI original:

bash

```
aws dynamodb query \
--table-name Usuarios \
--key-condition-expression "UsuarioID = :id" \
--expression-attribute-values '{":id": {"S": "12345"}}'
```

Equivalente en PartiQL:

sql

```
SELECT * FROM Usuarios WHERE UsuarioID = '12345'
```

Consultas más específicas:

sql

```
-- Obtener solo nombre y edad
SELECT Nombre, Edad FROM Usuarios WHERE UsuarioID = '12345'

-- Consultar por rango de edad (si tienes GSI)
SELECT * FROM Usuarios WHERE Edad > 25

-- Consultar múltiples usuarios
SELECT * FROM Usuarios WHERE UsuarioID IN ['12345', '67890']
```

4. Obtener todos los elementos (scan)

Comando AWS CLI original:

bash

```
aws dynamodb scan --table-name Usuarios
```

Equivalente en PartiQL:

sql

```
SELECT * FROM Usuarios
```

Variaciones útiles:

sql

```
-- Obtener solo nombres
SELECT Nombre FROM Usuarios

-- Filtrar por edad
SELECT * FROM Usuarios WHERE Edad >= 30

-- Contar elementos
SELECT COUNT(*) FROM Usuarios
```

5. Obtener un ítem específico (get-item)

Comando AWS CLI original:

bash

```
aws dynamodb get-item \
--table-name Usuarios \
--key '{"UsuarioID": {"S": "12345"}}'
```

Equivalente en PartiQL:

sql

```
SELECT * FROM Usuarios WHERE UsuarioID = '12345'
```

6. Actualizar un ítem (update-item)

Comando AWS CLI original:

bash

```
aws dynamodb update-item \
--table-name Usuarios \
--key '{"UsuarioID": {"S": "12345"}}' \
--update-expression "SET Edad = :nuevaEdad" \
--expression-attribute-values '{":nuevaEdad": {"N": "31"}}'
```

Equivalente en PartiQL:

sql

```
UPDATE Usuarios SET Edad = 31 WHERE UsuarioID = '12345'
```

Actualizaciones más complejas:

sql

```
-- Actualizar múltiples campos
UPDATE Usuarios SET Edad = 32, Nombre = 'Carlos Pérez García' WHERE UsuarioID = '12345'

-- Agregar un nuevo atributo
UPDATE Usuarios SET Email = 'carlos@email.com' WHERE UsuarioID = '12345'

-- Actualizar con condición
UPDATE Usuarios SET Edad = 33 WHERE UsuarioID = '12345' AND Edad < 35
```

7. Eliminar un ítem (delete-item)

Comando AWS CLI original:

bash

```
aws dynamodb delete-item \
--table-name Usuarios \
--key '{"UsuarioID": {"S": "12345"}}'
```

Equivalente en PartiQL:

sql

```
DELETE FROM Usuarios WHERE UsuarioID = '12345'
```

Eliminaciones condicionales:

```
sql

-- Eliminar solo si cumple condición
DELETE FROM Usuarios WHERE UsuarioID = '12345' AND Edad > 30

-- Eliminar múltiples elementos (cuidado!)
DELETE FROM Usuarios WHERE Edad < 25
```

8. Eliminar tabla

Nota: No se puede hacer con PartiQL. Debes usar la interfaz gráfica de NoSQL Workbench o AWS CLI:

```
bash

aws dynamodb delete-table --table-name Usuarios --endpoint-url http://localhost:8000
```

Ejercicios prácticos completos

Secuencia completa de pruebas:

1. **Crear usuarios:**

```
sql

INSERT INTO Usuarios VALUE {'UsuarioID': '001', 'Nombre': 'María López', 'Edad': 28, 'Ciudad': 'Madrid'}
INSERT INTO Usuarios VALUE {'UsuarioID': '002', 'Nombre': 'Pedro Sánchez', 'Edad': 35, 'Ciudad': 'Barcelona'}
INSERT INTO Usuarios VALUE {'UsuarioID': '003', 'Nombre': 'Laura Jiménez', 'Edad': 22, 'Ciudad': 'Valencia'}
```

2. **Consultar todos:**

```
sql

SELECT * FROM Usuarios
```

3. **Consultar específico:**

```
sql
```

```
SELECT * FROM Usuarios WHERE UsuarioID = '001'
```

4. **Filtrar por edad:**

```
sql

SELECT * FROM Usuarios WHERE Edad > 25
```

5. **Actualizar usuario:**

```
sql

UPDATE Usuarios SET Edad = 29, Ciudad = 'Sevilla' WHERE UsuarioID = '001'
```

6. **Verificar actualización:**

```
sql

SELECT * FROM Usuarios WHERE UsuarioID = '001'
```

7. **Eliminar usuario:**

```
sql

DELETE FROM Usuarios WHERE UsuarioID = '003'
```

8. **Verificar eliminación:**

```
sql

SELECT * FROM Usuarios
```

Ventajas de PartiQL sobre AWS CLI

1. **Sintaxis familiar:** Similar a SQL estándar
2. **Más intuitivo:** Para desarrolladores con experiencia en SQL
3. **Menos verboso:** No necesitas especificar tipos de datos
4. **Mejor para consultas complejas:** Especialmente con filtros y joins

Notas importantes

- **Tipos de datos:** PartiQL maneja automáticamente los tipos (no necesitas especificar {"S": "valor"})
- **Comillas:** Usa comillas simples para strings en PartiQL
- **Números:** Los números se escriben directamente sin comillas
- **Booleanos:** true/false sin comillas
- **Listas:** [1, 2, 3] o ['a', 'b', 'c']
- **Objetos:** {'campo': 'valor'}

¡Ahora puedes ejecutar todas estas consultas en la pestaña "PartiQL editor" de NoSQL Workbench!