Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2115 - Programación como Herramienta para la Ingeniería

Consultas en SQL

Profesora: Francesca Lucchini
Prof. Coordinador: Hans Löbel

Structured Query Language (SQL)

Tiene dos componentes principales

- Lenguaje de definición de datos (DDL)
  - Creación de tablas (CREATE)
  - Eliminación detablas (DROP)
  - **Modificación de definiciones de tablas (ALTER)**

  *Las restricciones de integridad se pueden definir en tablas, ya sea cuando se crea la tabla o posteriormente.

- Lenguaje de manipulación de datos (DML)
  - *Data Manipulation Language*
  - Consultas (SELECT)
  - Inserción de datos (INSERT)
  - Actualización datos (UPDATE)
  - Eliminar datos (DELETE)

Alterar Tabla

Podemos modificar el esquema de la tabla: cambiar nombre de la tabla, agregar, eliminar o modificar columnas.

ALTER TABLE table_name **RENAME TO** new-table-name

ALTER TABLE table_name **RENAME COLUMN** column-name **TO** new-column-name

Num_Emp	Nombre	Edad	Departmento	Sueldo
001	Alex S.	26	Ventas	700000
002	Gerardo C.	32	Marketing	1200000
003	Regina G.	31	Marketing	1200000
004	Juana R.	26	Seguridad	520000

ALTER TABLE Empleados **RENAME TO** Empleades_2022
ALTER TABLE Empleades_2022 **RENAME COLUMN** Salario **TO** Sueldo

Alterar Tabla

Podemos modificar el esquema de la tabla: cambiar nombre de la tabla, agregar, eliminar o modificar columnas.

ALTER TABLE table_name **ADD COLUMN** column-name data-type [constraint-1 constraint-2]*

ALTER TABLE table_name **DROP COLUMN** column-name

Num_Emp	Nombre	Edad	Departmento	Salario	Bono	Sede
001	Alex S.	26	Ventas	700000	1.23	NULL
002	Gerardo C.	32	Marketing	1200000	1.23	NULL
003	Regina G.	31	Marketing	1200000	1.23	NULL
004	Juana R.	26	Seguridad	520000	1.23	NULL

ALTER TABLE Empleados **ADD COLUMN** Bono **REAL DEFAULT** 1.23
ALTER TABLE Empleades **ADD COLUMN** Sede **TEXT**

*Lista de Constraints de SQLite

Alterar Tabla

Podemos modificar el esquema de la tabla: cambiar nombre de la tabla, agregar, eliminar o modificar columnas.

ALTER TABLE table_name **DROP COLUMN** column-name

Num_Emp	Nombre	Departmento	Sueldo
001	Alex S.	Ventas	700000
002	Gerardo C.	Marketing	1200000
003	Regina G.	Marketing	1200000
004	Juana R.	Seguridad	520000

ALTER TABLE Empleados **DROP COLUMN** Edad

Structured Query Language (SQL)

Tiene dos componentes principales

- Lenguaje de definición de datos (DDL)
  - Creación de tablas (CREATE)
  - Eliminación detablas (DROP)
  - Modificación de definiciones de tablas (ALTER)

  *Las restricciones de integridad se pueden definir en tablas, ya sea cuando se crea la tabla o posteriormente.

- Lenguaje de manipulación de datos (DML)
  - *Data Manipulation Language*
  - **Consultas (SELECT)**
  - Inserción de datos (INSERT)
  - Actualización datos (UPDATE)
  - Eliminar datos (DELETE)

Consultas

SELECT [DISTINCT]
column_list
FROM
table_list
WHERE
row_filter

Num_Emp	Nombre	Edad	Departmento	Salario
001	Alex S.	26	Ventas	700000
002	Gerardo C.	32	Marketing	1200000
003	Regina G.	31	Marketing	1200000
004	Juana R.	26	Seguridad	520000

SELECT * FROM Empleados

Consultas

SELECT [DISTINCT]
column_list
FROM
table_list
WHERE
row_filter

Num_Emp	Nombre	Edad	Departmento	Salario
002	Gerardo C.	32	Marketing	1200000
003	Regina G.	31	Marketing	1200000

SELECT * FROM Empleados

SELECT * FROM Empleados E **WHERE** E.Edad > 30

Consultas

```
SELECT [DISTINCT]
    column_list
FROM
    table_list
WHERE
    row_filter
```

Nombre	Edad
Gerardo C.	32
Regina G.	31

SELECT * FROM Empleados

SELECT * FROM Empleados E **WHERE** E.Edad > 30

SELECT Nombre, Edad **FROM** Empleados E **WHERE** E.Edad > 30

Joins

Num_Emp	Nombre	Edad	Departmento_Id	Salario
001	Alex S.	26	1	700000
002	Gerardo C.	32	2	1200000
003	Regina G.	31	2	1200000
004	Juana R.	26	3	520000

Id	Departamento
1	Ventas
2	Marketing
3	Seguridad

SELECT Nombre **FROM** Empleados E, Departamentos D **WHERE** E.Departamento_id = D.id **AND** D.Departamento = “Marketing”

Nombre
Gerardo C.
Regina G.

Podemos usar AND, OR, NOT, BETWEEN e IN para nuestras condiciones de consulta.

Usos:

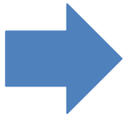
BETWEEN 25 AND 27

IN (25, 27, 30)

Anidación

Nombre	Edad	Salario
Alex S.	26	700000
Juana R.	26	520000

Tabla de intermedia de la anidación



Nombre	Salario
Alex S.	700000

Tabla final

SELECT Nombre, Salario **FROM** (**SELECT** Nombre, Edad, Salario **FROM** Empleados E **WHERE** E.Edad < 30) **WHERE** Salario >= 700000

SELECT Nombre, Salario **FROM** Empleados E **WHERE** E.Salario >= 700000 **AND** E.Edad < 30

Otras funciones importantes

ORDER BY

GROUP BY - HAVING

COUNT

SUM

AVG

MAX

MIN

Operadores Aritméticos (=, -, /, *, %, …)

Operadores Comparación (<, >, =<, ==, !=, …)

Documentación Oficial sobre la sintaxis de SQLite

https://www.sqlite.org/docs.html

Tutoriales, ejemplos del uso de SQLite:

https://www.tutorialspoint.com/sqlite/index.htm

SELECT [COUNT | SUM | MIN | MAX](column-list) **FROM** table_name

[**WHERE** condition]

[**GROUP BY** column1, column2....columnN]

[**ORDER BY** column1, column2, .. columnN] [**ASC** | **DESC**];

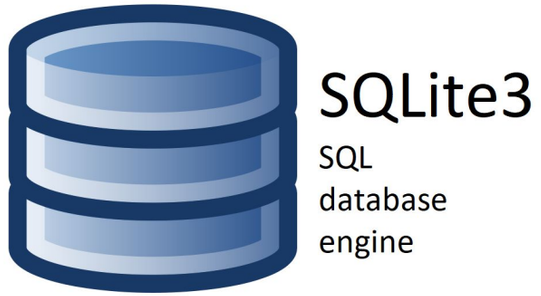
Limitaciones de SQLite

- Agregar columna nueva no permite
  - Restricciones de llave primaria y foránea

- Editar el esquema es limitado

  Usualmente se va a tener que eliminar una tabla para modificar cosas como

  - Tipo de datos
  - Restricciones de llaves



Uso en Python: DML

```
connection = sqlite3.connect("ejemplo.db")
cursor = connection.cursor()

sqlStatement = "SELECT * FROM Empleados"

cursor.execute(sqlStatement)

una_fila = cursor.fetchone()
todas_filas = cursor.fetchall()

connection.close()
```

Uso en Python: parametrización

```
def mayores_que(edad):
    connection = sqlite3.connect("ejemplo.db")
    cursor = connection.cursor()

    sqlStatement = "SELECT * FROM Empleados E WHERE E.Age > {}".format(edad)

    cursor.execute(sqlStatement)
    resp = cursor.fetchall()
    connection.close()
    return resp
```

Uso en Python: parametrización

```
def mayores_que(edad):
    connection = sqlite3.connect("ejemplo.db")
    cursor = connection.cursor()

    sqlStatement = f"SELECT * FROM Empleados E WHERE E.Age > {edad}"

    cursor.execute(sqlStatement)
    resp = cursor.fetchall()
    connection.close()
    return resp
```

Uso en Python: parametrización

```
def mayores_que(edad):
    connection = sqlite3.connect("ejemplo.db")
    cursor = connection.cursor()

    sqlStatement = "SELECT * FROM Empleados E WHERE E.Age > ?’

    cursor.execute(sqlStatement, (edad,))
    resp = cursor.fetchall()
    connection.close()
    return resp
```

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2115 - Programación como Herramienta para la Ingeniería

Consultas en SQL

Profesora: Francesca Lucchini
Prof. Coordinador: Hans Löbel