

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



IIC2115 - Programación como herramienta para la ingeniería

Capítulo 3: Bases de datos

Profesores: Francisco Garrido-Valenzuela / Hans Löbel

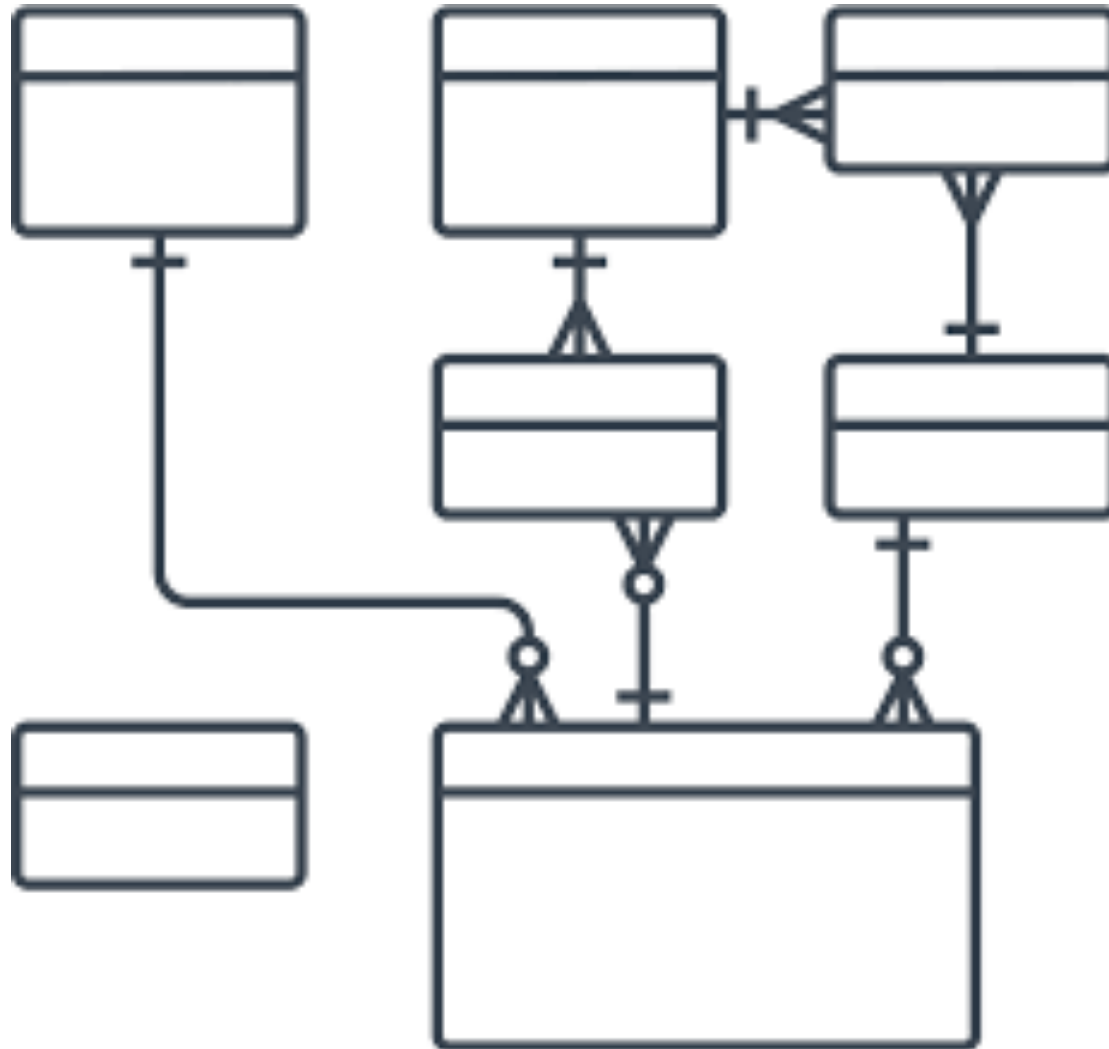
Contenidos

- ¿Qué es una base de datos?
- Table
- *Primary key y Foreign key*
- SQL
- Operatorias
 - Creación (CREATE)
 - Inserción (INSERT)
 - Modificación (UPDATE)
 - Eliminación (DELETE)
- Consultas, joins y anidación
- Uso en Python

¿Qué es una base de datos?

- 1. Corresponde un conjunto de datos de un mismo contexto y almacenados en cierta lógica e indexados para su posterior uso*
- 2. Es una colección de una o más relaciones, donde cada relación es una tabla con filas y columnas.*

¿Qué es una base de datos?



Database

Schema

Table

Table

Columna: Guarda un específico tipo de datos

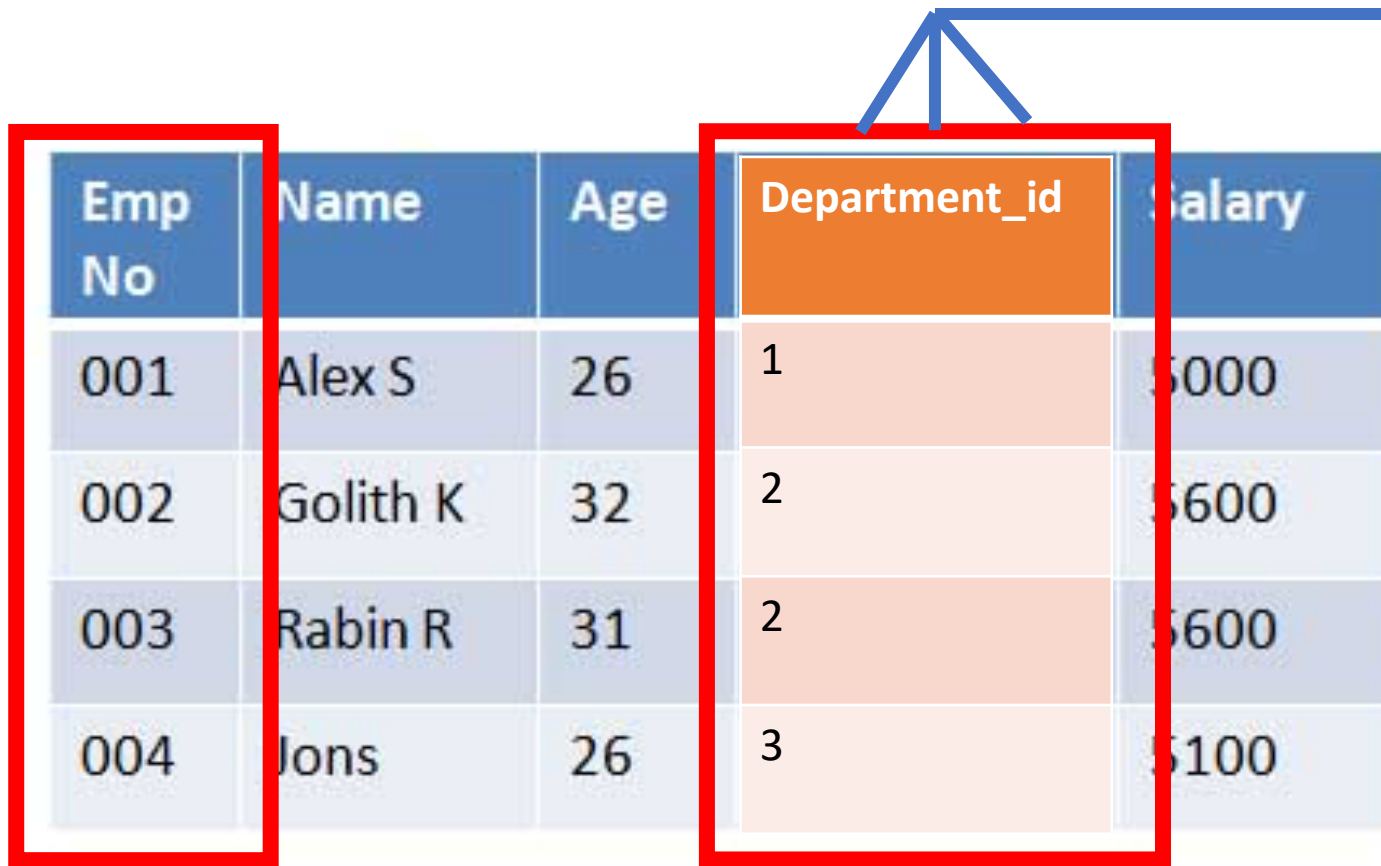
CHAR(N)
VARCHAR(N)
INTEGER
REAL
...

Fila:
Corresponde a
un registro o
instancia

Emp No	Name	Age	Department	Salary
001	Alex S	26	Store	5000
002	Golith K	32	Marketing	5600
003	Rabin R	31	Marketing	5600
004	Jons	26	Security	5100

Empleados (Emp No: CHAR(3), Name: VARCHAR(10), Age: INTEGER, Department: VARCHAR(20), Salary: REAL)

Primary Key y Foreign Key



Emp No	Name	Age	Department_id	Salary
001	Alex S	26	1	\$000
002	Golith K	32	2	\$600
003	Rabin R	31	2	\$600
004	Jons	26	3	\$100

PK

FK

Id	Department
1	Store
2	Marketing
3	Security

No hay FK

Structured Query Language (SQL)

Lenguaje de definición de datos (DDL)

Creación

Inserción

Eliminación

Modificación de definiciones de tablas.

Lenguaje de manipulación de datos (DML)

Consultas

Creación

CREATE TABLE [IF NOT EXISTS]

```
table_name (  
    column_1 data_type,  
    column_2 data_type  
)
```

Emp No	Name	Age	Department	Salary
001	Alex S	26	Store	5000
002	Golith K	32	Marketing	5600
003	Rabin R	31	Marketing	5600
004	Jons	26	Security	5100

CREATE TABLE Empleados (Emp_No CHAR(3), Name VARCHAR(20), Age INTEGER, Department VARCHAR(10), Salary REAL)

Inserción

INSERT INTO table_name (column1,column2 ,..) **VALUES**(value1, value2 ,...)

Emp No	Name	Age	Department	Salary
001	Alex S	26	Store	5000
002	Golith K	32	Marketing	5600
003	Rabin R	31	Marketing	5600

INSERT INTO Empleados (Emp_No, Name, Age, Department, Salary) **VALUES** ('004', 'Jons', 26, 'Security', 5100)

Modificación

UPDATE table_name **SET**

column_1 = new_value_1,

column_2 = new_value_2

WHERE

search_condition

Emp No	Name	Age	Department	Salary
001	Alex S	26	Store	5000
002	Golith K	32	Marketing	5600
003	Rabin R	31	Marketing	5600
004	Jons	26	Marketing	5100

UPDATE Empleados E **SET** E.Department = 'Marketing' **WHERE** E.Emp_No = '004'

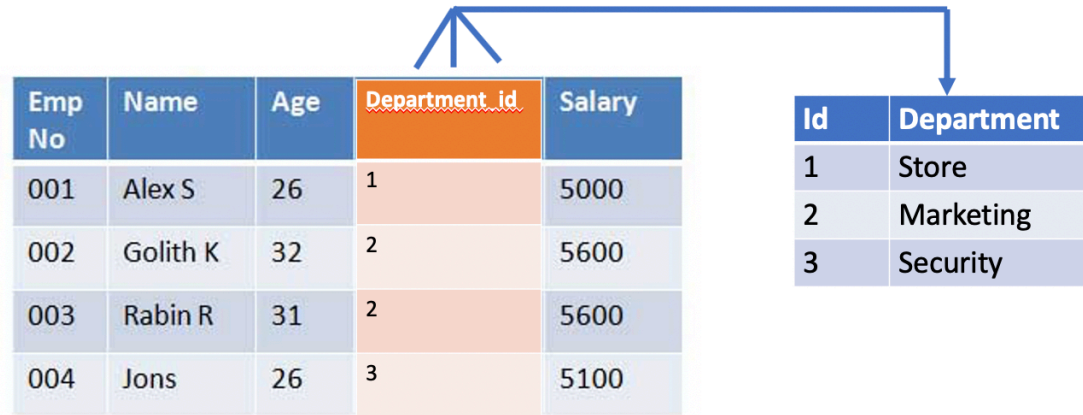
Eliminación

DELETE FROM table_name **WHERE** search_condition;

Emp No	Name	Age	Department	Salary
001	Alex S	26	Store	5000
002	Golith K	32	Marketing	5600
003	Rabin R	31	Marketing	5600

DELETE FROM Empleados E **WHERE** E.Emp_No = '004'

Creación con *Primary Key* y *Foreign Key*



```
CREATE TABLE Departments (Id INTEGER, Department VARCHAR(20), PRIMARY KEY(Id))
```

```
CREATE TABLE Empleados (Emp_No CHAR(3), Name VARCHAR(20), Age INTEGER, Department_id INTEGER, Salary REAL, PRIMARY KEY(Emp_No), FOREIGN KEY (Department_id) REFERENCES Departments.Id)
```

Consultas

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

Name	Age
------	-----

Golith K	32
Rabin R	31


SELECT * FROM Empleados

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

SELECT Name, Age **FROM** Empleados E **WHERE** E.Age > 30

Joins

SELECT [DISTINCT]
column_list
FROM
table_list
WHERE
row_filter



Emp No	Name	Age	Department_id	Salary
001	Alex S	26	1	5000
002	Golith K	32	2	5600
003	Rabin R	31	2	5600
004	Jons	26	3	5100

Id	Department
1	Store
2	Marketing
3	Security

SELECT Name **FROM** Empleados E, Departments D **WHERE** E.Department_id = D.id
AND D.Department = 'Store'

ALEX S

Anidación

Emp No	Name	Age	Department	Salary
001	Alex S	26	Store	5000
002	Golith K	32	Marketing	5600
003	Rabin R	31	Marketing	5600
004	Jons	26	Security	5100

SELECT Name **FROM** (SELECT Name, Age, Salary **FROM** Empleados
E **WHERE** E.Age < 30) **WHERE** E.Salary >= 5100

SELECT Name **FROM** Empleados E **WHERE** E.Salary >= 5100 **AND**
E.Age < 30)

Otras funciones importantes

ORDER BY

GROUP BY

COUNT

SUM

AVG

MAX

MIN

Uso en Python: DDL

```
import sqlite3

connection = sqlite3.connect('ejemplo.db')
cursor = connection.cursor()

sqlStatement = 'CREATE TABLE Empleados (Emp_No CHAR(3), Name VARCHAR(20), Age INTEGER, Department VARCHAR(10), Salary REAL)'
cursor.execute(sqlStatement)

Sql2 = 'INSERT INTO Empleados (Emp_No, Name, Age, Department, Salary) VALUES ('004', 'Jons', 26, 'Security', 5100)'

cursor.execute(Sql2)

connection.commit()
connection.close()
```

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 de Python: Parametrizando...

```
def mayores_que(edad):  
    connection = sqlite3.connect('ejemplo.db')  
    cursor = connection.cursor()  
  
    sqlStatement = f'SELECT * FROM Empleados E WHERE E.Age > {edad}'  
    sqlStatement = 'SELECT * FROM Empleados E WHERE E.Age > {}'.format(edad)  
  
    cursor.execute(sqlStatement)  
    resp = cursor.fetchall()  
    connection.close()  
    return resp
```

Manejo de errores

- Pueden encontrar errores que vienen de Python (de los que ya están familiarizados)
- Errores de la sintaxis de la base de datos (SQL)

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



IIC2115 - Programación como herramienta para la ingeniería

Capítulo 3: Bases de datos

Profesores: Francisco Garrido-Valenzuela / Hans Löbel