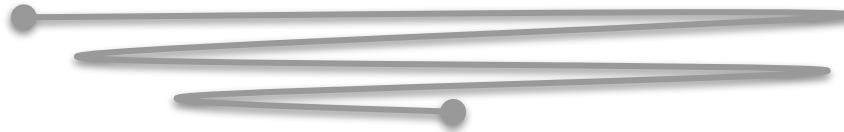


Laboratorio de Datos



AR - SQL - Parte 01

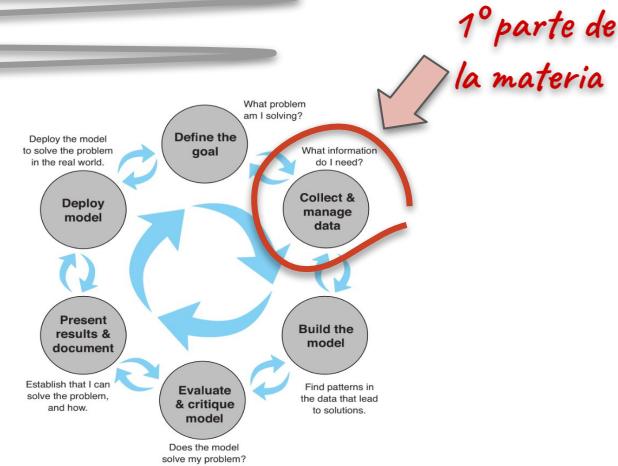


Recorrido de la materia (hasta ahora)

- ✓ Lenguaje de programación para trabajar en nuestros proyectos



- ✓ Etapas de un proyecto de Ciencias de Datos



- ✓ Modelado de Datos



- ✓ Representación de los Datos

Materia

Código	Nombre
1	Laboratorio de Datos
2	Análisis II
3	Álgebra Lineal

Unidad

Código	Materia	Título	Descripción
1	Administración de datos	Obtención y Manejo de los datos	
1	Modelos Explicativos	Construcción de modelos explicativos	
1	Modelos Predictivos	Construcción de modelos predictivos	
2	Integrales sobre curvas y	Integrales en múltiples variables	
2	Ecuaciones Diferenciales		

Tarea Procesamiento de Datos - Preguntas



Responder las siguientes preguntas (escribir las respuestas a modo de comentario dentro del script)

1. ¿Cómo afectó a la programación de la función cuando cambiaron levemente la matriz de empleado?
 - a. En el caso en que le agregaron más filas
 - b. En el caso en que le alteraron el orden de las columnas
2. ¿Y cuando a empleado le cambiaron la forma de representar las matrices (de lista de filas a lista de columnas)?
3. ¿Cuál es la ventaja, desde el punto de vista del usuario de la función, disponer de ella y no escribir directamente el código de la consulta dentro de su programa?

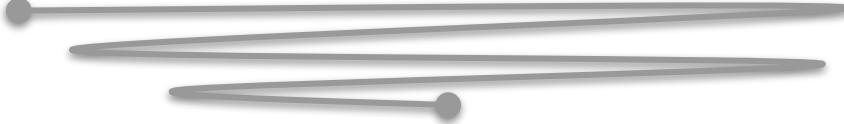
Tarea Procesamiento de Datos - Cierre



I. Modificaciones en estructura de datos

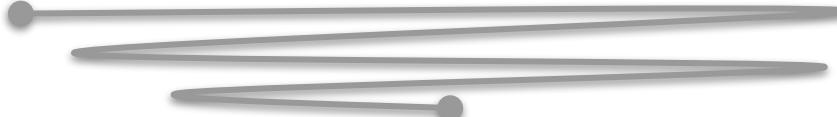
-> Cambios en programación de las consultas

Tarea Procesamiento de Datos - Reflexión



- ✓ Necesitamos “algo” que nos permita acceder a los datos independientemente de cómo se encuentran almacenados físicamente (matriz como lista de filas, matriz como lista de columnas, orden en que se encuentran almacenadas las columnas, etc.)
- ✓ Nos vendría bien un lenguaje que nos permita expresar el acceso a los datos, independientemente de su implementación
- ✓ El modelo relacional tiene un lenguaje desarrollado para ello llamado SQL y se basa en el Álgebra Relacional (AR)

Álgebra Relacional



Empecemos con ...

Álgebra Relacional

Verano - 2024

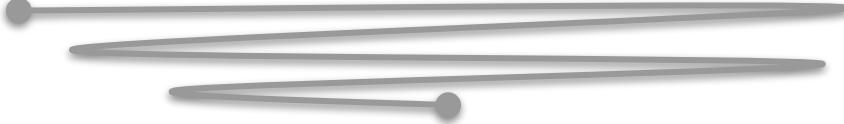


SQL - Introducción

Pasemos a ...



SQL - Introducción



- ✓ SQL = Structured Query Language
- ✓ Lenguaje universalmente más usado para bases de datos relacionales
- ✓ Lenguaje declarativo de alto nivel
- ✓ Desarrollado por IBM (1974-1977)
- ✓ Se convirtió en un standard definido por:
 - ANSI (American National Standards Institute)
 - ISO (International Standards Organization)
- ✓ El estándar actual es el SQL:1999 (aunque muchas DBMS no lo implementaron por completo aún. Existen revisiones del 2003 y 2006)

SQL - Introducción

✓ Sentencias del SQL se dividen en:

- Sentencias DDL** (Data Definition Language). Permiten crear/modificar/borrar estructuras de datos (Ej. Tablas). Acá también es donde se definen las restricciones (claves, foreign keys, etc.).
Ej. de comandos: CREATE/ALTER/DROP TABLE.

- Sentencias DML** (Data Manipulation Language). Posibilitan manipular datos.
Basado en Álgebra Relacional.

Ej. de comandos: SELECT/INSERT/UPDATE/DELETE. Si no se cumple la restricción avisa.

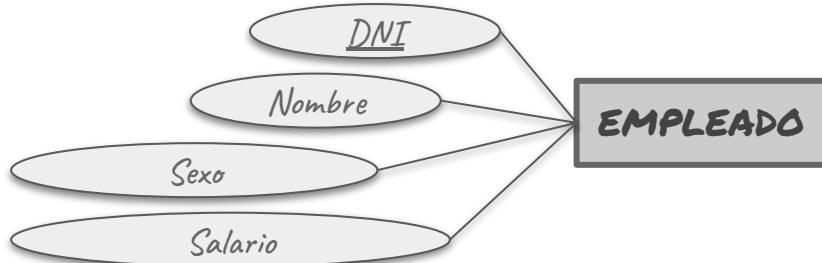
✓ También provee sentencias para:

- Definir permisos (control de acceso de usuarios)
- Manejo de transacciones
- Otros

Nos vamos a enfocar en DML.
¡En particular en SELECT!

AR <-> SQL

1. Contamos con el siguiente DER

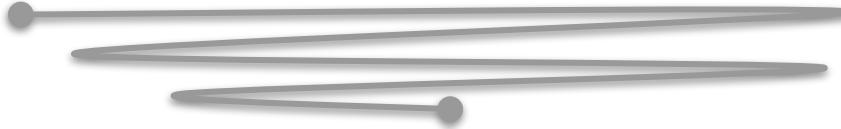


2. Lo mapeamos a la siguiente estructura del Modelo Relacional: **EMPLEADO(DNI, Nombre, Sexo, Salario)**

3. La tabla contiene los siguientes datos:

EMPLEADO			
DNI	Nombre	Sexo	Salario
20222333	Diego	M	\$20.000,00
33456234	Laura	F	\$25.000,00
45432345	Marina	F	\$10.000,00

AR <-> SQL

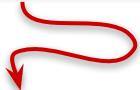


AR - Proyección <-> SQL - SELECT

AR - Proyección <-> SQL - SELECT

Consigna: Listar DNI y Salario de EMPLEADO

EMPLEADO			
DNI	Nombre	Sexo	Salario
20222333	Diego	M	\$20.000,00
33456234	Laura	F	\$25.000,00
45432345	Marina	F	\$10.000,00



DNI	Salario
20222333	\$20.000,00
33456234	\$25.000,00
45432345	\$10.000,00

Álgebra Relacional

$\pi_{DNI, Salario}(EMPLEADO)$

SQL

```
SELECT DISTINCT DNI, Salario  
FROM empleado;
```

SQL - SELECT <-> Python

```
1 # -*- coding: utf-8 -*-
2 """
3     Materia: Laboratorio de datos - FCEyN - UBA
4     Clase : Clase SQL. Script clase.
5     Autor : Pablo Turjanski
6     Fecha : 2023-03-07
7 """
8
9 # Importamos bibliotecas
10 import pandas as pd
11 from inline_sql import sql, sql_val
12
```

- } *Información sobre el programa*
- } *Bibliotecas.*
- Pandas. Para dataframes
 - inline_sql: Para lenguaje sql

SQL - SELECT <-> Python

Función para generar el dataframe empleado

```
159 # =====
160 # FUNCIONES PARA LA GENERACIÓN DE DATAFRAMES
161 #
162 def get_empleado():
163     # Genera el dataframe "empleado" que contiene las siguientes columnas
164     # (en el orden mencionado):
165     # 1. DNI
166     # 2. Nombre
167     # 3. Sexo
168     # 4. Salario
169
170     # ... Creamos el dataframe vacío (sólo con los nombres de sus columnas)
171     empleado = pd.DataFrame(columns = ['DNI', 'Nombre', 'Sexo', 'Salario'])
172     # ... Agregamos cada una de las filas al DataFrame
173     empleado = pd.concat([empleado,pd.DataFrame([
174         {'DNI' : 20222333, 'Nombre' : 'Diego', 'Sexo' : 'M', 'Salario' : 20000.0},
175         {'DNI' : 33456234, 'Nombre' : 'Laura', 'Sexo' : 'F', 'Salario' : 25000.0},
176         {'DNI' : 45432345, 'Nombre' : 'Marina', 'Sexo' : 'F', 'Salario' : 10000.0}
177     ]))
178
179     return empleado
180
```

Definición de estructura de dataframe (vacío)

} Se insertan 3 filas nuevas

Usamos la función

```
13 def main():
14
15     print()
16     print("# =====")
17     print("# Creamos los datasets que vamos a utilizar en este programa")
18     print("# =====")
19
20     # Ejercicios AR-PROJECT, SELECT, RENAME
21     empleado = get_empleado()
22     # Ejercicios AR-UNION, INTERSECTION, MINUS
```

Llamamos a la función y el resultado lo guardamos en un dataframe

SQL - SELECT <-> Python

Vemos el contenido del
dataframe original

La Consulta se define
como un string

```
45 print()  
46 print("# =====")  
47 print("# Ejemplo inicial")  
48 print("# =====")  
49  
50 print(empleado)  
51  
52 consultaSQL = """  
53     SELECT DISTINCT DNI, Salario  
54     FROM empleado;  
55 """  
56  
57  
58 dataframeResultado = sql^ consultaSQL  
59  
60 print(dataframeResultado)  
61
```

Se imprime el resultado

Se ejecuta la consulta
y se almacena en un
dataframe

SQL - SELECT <-> Python

```
# -----
# DEFINICIÓN DE FUNCIÓN DE IMPRESIÓN EN PANTALLA
# -----
# Imprime en pantalla en un formato ordenado:
# 1. Consigna
# 2. Cada dataframe de la lista de dataframes de entrada
# 3. Query
# 4. Dataframe de salida
def imprimirEjercicio(consigna, listaDeDataframesDeEntrada, consultaSQL, dataframeResultadoDeConsultaSQL):
    print("# -----")
    print("# Consigna: ", consigna)
    print("# -----")
    print()
    for i in range(len(listaDeDataframesDeEntrada)):
        print("# Entrada {},".format(i), sep='')
        print("# -----")
        print(listaDeDataframesDeEntrada[i])
        print()
    print("# SQL:")
    print("# ----")
    print(consultaSQL)
    print()
    print("# Salida:")
    print("# -----")
    print(dataframeResultadoDeConsultaSQL)
    print()
    print("# -----")
    print("# -----")
    print()
print()
```

Creamos una función a la que le pasamos:

1. una consigna (string)
2. una lista de dataframes de entrada (dataframe/s)
3. una consulta SQL (string)
4. el resultado de la consulta SQL (dataframe)

Y como resultado imprime en pantalla:

1. la consigna
2. cada dataframe de entrada (original)
3. la consulta SQL (string)
4. el resultado de la consulta SQL

SQL - SELECT <-> Python

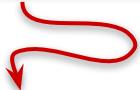
```
62
63     print()
64     print("# =====")
65     print("# Ejercicios AR-PROJECT <-> SELECT")
66     print("# =====")
67
68     consigna    = "a.- Listar DNI y Salario de empleados "
69     consultaSQL = """
70             SELECT DISTINCT DNI, Salario
71                 FROM empleado;
72             """
73
74     imprimirEjercicio(consigna, [empleado], consultaSQL, sql^consultaSQL)
```

Uso de la función que creamos

AR - Proyección <-> SQL - SELECT

Consigna: Listar Sexo de EMPLEADO

EMPLEADO			
DNI	Nombre	Sexo	Salario
20222333	Diego	M	\$20.000,00
33456234	Laura	F	\$25.000,00
45432345	Marina	F	\$10.000,00



Sexo
M
F

Álgebra Relacional

$\pi_{Sexo}(EMPLEADO)$

SQL

```
SELECT DISTINCT Sexo  
FROM empleado;
```

SQL - SELECT <-> Python

```
74
75      # -----
76      consigna    = "b.- Listar Sexo de empleados "
77      consultaSQL = """
78          SELECT DISTINCT Sexo
79          FROM empleado;
80      """
81      imprimirEjercicio(consigna, [empleado], consultaSQL, sql^consultaSQL)
```

AR - Proyección <-> SQL - SELECT

Consigna: Listar Sexo de EMPLEADO

EMPLEADO			
DNI	Nombre	Sexo	Salario
20222333	Diego	M	\$20.000,00
33456234	Laura	F	\$25.000,00
45432345	Marina	F	\$10.000,00



Sexo
M
F

Álgebra Relacional

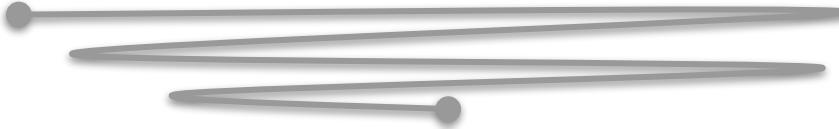
$\pi_{Sexo}(EMPLEADO)$

¿Qué sucede si removemos
DISTINCT?

SQL

```
SELECT DISTINCT Sexo  
FROM empleado;
```

AR <-> SQL



AR - Selección <-> SQL - WHERE

AR - Selección <-> SQL - WHERE

Consigna: Listar de EMPLEADO sólo aquellos cuyo sexo es femenino

EMPLEADO

DNI	Nombre	Sexo	Salario
20222333	Diego	M	\$20.000,00
33456234	Laura	F	\$25.000,00
45432345	Marina	F	\$10.000,00

Álgebra Relacional

$$\sigma_{Sexo=F}(EMPLEADO)$$


DNI	Nombre	Sexo	Salario
33456234	Laura	F	\$25.000,00
45432345	Marina	F	\$10.000,00

SQL

```
SELECT DISTINCT DNI, Nombre, Sexo, Salario  
FROM empleado  
WHERE Sexo='F';
```

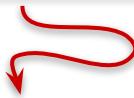
AR - Selección <-> SQL - WHERE

Consigna: Listar de EMPLEADO aquellos cuyo sexo es femenino y su salario es mayor a \$15.000

EMPLEADO

DNI	Nombre	Sexo	Salario
20222333	Diego	M	\$20.000,00
33456234	Laura	F	\$25.000,00
45432345	Marina	F	\$10.000,00

Álgebra Relacional

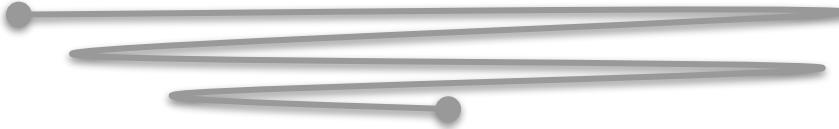
$$\sigma_{\text{Sexo} = F \text{ AND } \text{Salario} > \$15.000}(\text{EMPLEADO})$$


DNI	Nombre	Sexo	Salario
33456234	Laura	F	\$25.000,00

SQL

```
SELECT DISTINCT DNI, Nombre, Sexo, Salario  
FROM empleado  
WHERE Sexo='F' AND Salario>15000;
```

AR <-> SQL



AR - Renombre <-> SQL - AS

AR - Renombre \leftrightarrow SQL - AS

Consigna: Listar DNI y Salario de EMPLEADO, y renombrarlos como id e Ingreso

EMPLEADO

DNI	Nombre	Sexo	Salario
20222333	Diego	M	\$20.000,00
33456234	Laura	F	\$25.000,00
45432345	Marina	F	\$10.000,00

Álgebra Relacional

$EMPLEADO(id, Ingreso) \leftarrow \pi_{DNI, Salario}(EMPLEADO)$

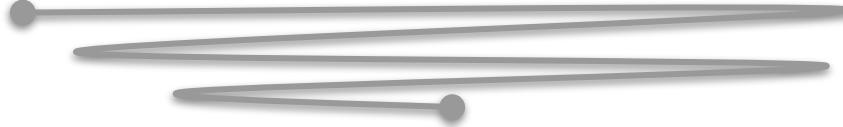


id	Ingreso
20222333	\$20.000,00
33456234	\$25.000,00
45432345	\$10.000,00

SQL

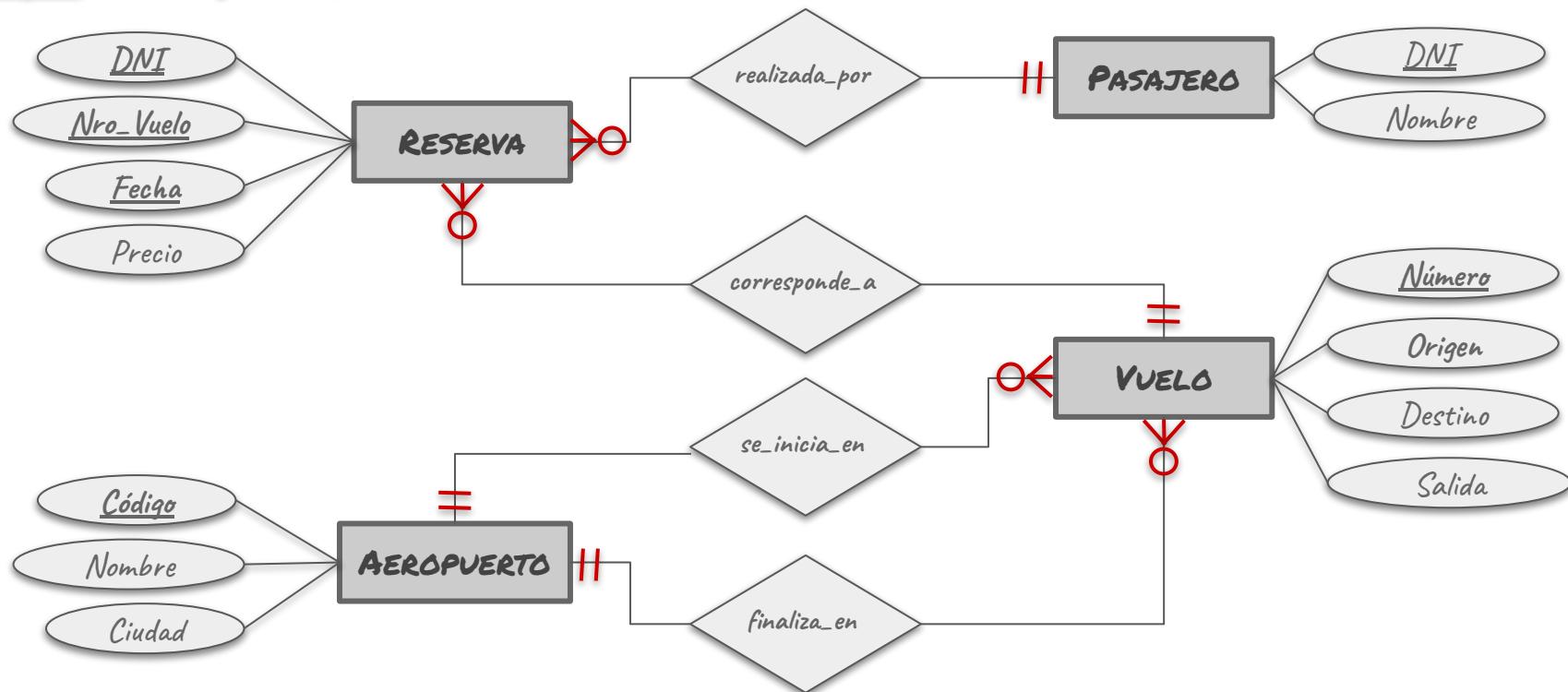
`SELECT DISTINCT DNI AS id, Salario AS Ingreso
FROM empleado;`

SQL - Ejercicio 1



SQL - Ejercicio 1

Consigna: Sea el siguiente DER



SQL - Ejercicio 1

VUELO

Número	Origen	Destino	Salida
345	MAD	CDG	12:30
321	MAD	ORY	19:05
165	LHR	CDG	09:55
903	CDG	LHR	14:40
447	CDG	LHR	17:00

AEROPUERTO

Código	Nombre	Ciudad
MAD	Barajas	Madrid
LGW	Gatwick	Londres
LHR	Heathrow	Londres
ORY	Orly	París
CDG	Charles de Gaulle	París

PASAJERO

DNI	Nombre
123	María
456	Pedro
789	Isabel

RESERVA

DNI	Nro_Vuelo	Fecha	Precio
789	165	07-01-11	210
123	345	20-12-10	170
789	321	15-12-10	250
456	345	03-11-10	190

1 Retornar Código y Nombre de los aeropuertos de Londres

2 ¿Qué retorna

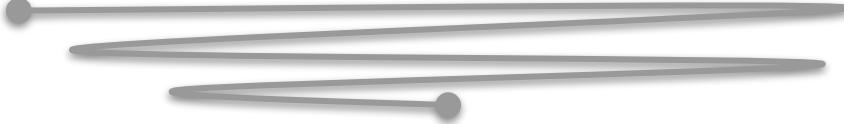
```
SELECT DISTINCT Ciudad AS City
FROM aeropuerto
WHERE Codigo='ORY' OR Codigo='CDG' ;
```

3 Obtener los números de vuelo que van desde CDG hacia LHR

4 Obtener los números de vuelo que van desde CDG hacia LHR o viceversa

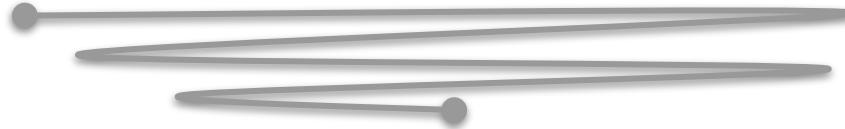
5 Devolver las fechas de reservas cuyos precios son mayores a \$200

Cierre



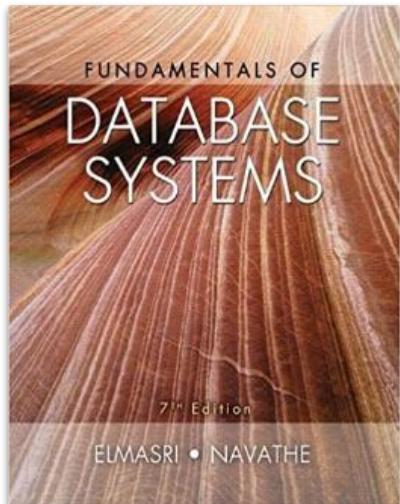
1. Álgebra Relacional. Provee fundamento formal a las operaciones asociadas al modelo relacional
2. SQL. Nos permite acceder a los datos independientemente de cómo se encuentran almacenados físicamente

Tareas para la próxima clase

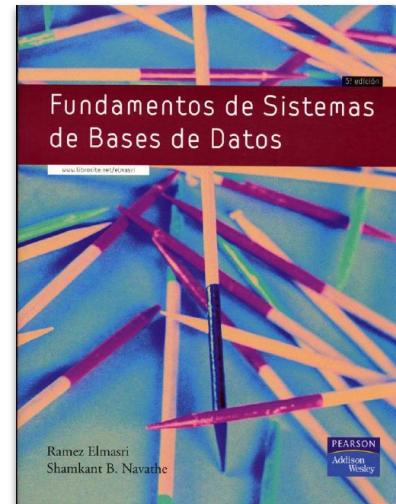


1. *Resolver la guía de ejercicios de “SQL”
(Sólo la sección A)*

Bibliografía



Elmasri/Navathe, *Fundamentals of Database Systems*,
7th. Ed., Pearson, 2016.



Elmasri/Navathe, *Fundamentos de Sistemas de Bases de Datos*,
5ta Ed., Pearson, 2007.

(Aviso. Difieren un poco en la notación)