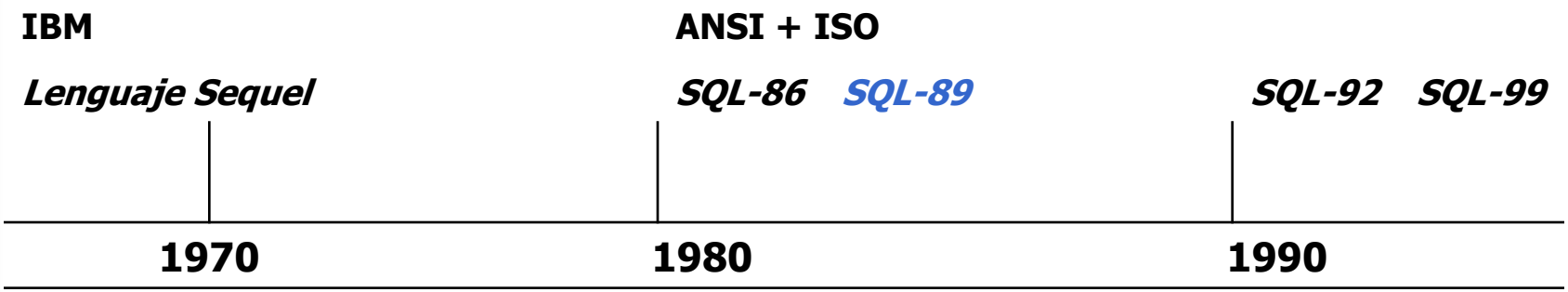
TECNICAS DE ALMACENAMIENTO DE DATOS

SQL - *Structured Query Language*



SQL:2003 - XML, sequence, columnas autonuméricas
SQL:2005 - SQL + XML, XQuery (World Wide Web Consortium)
SQL:2008 - ORDER BY, INSTEAD OF, TRUNCATE

ANSI (American National Standards Institute)
ISO (International Standards Organization)

SQL - *Structured Query Language*

- Lenguaje estándar de bases de datos relacionales
- Los SGBD son compatibles al menos con SQL-89
- Los SGBD ofrecen características no estándar

Lenguaje de manipulación de datos (DML)

INSERT

```
INSERT INTO CLIENTE
VALUES ('M','B','9',1);
```

UPDATE

```
UPDATE CLIENTE
   SET telefono = '4555-1234'
 WHERE cod_cliente = 1;
```

DELETE

```
DELETE
  FROM CLIENTE
 WHERE cod_cliente = 1;
```

SELECT

```
SELECT *
  FROM CLIENTE
 WHERE cod_cliente = 1;
```

DML - SELECT

```
--TODOS LOS CAMPOS
--TODOS LOS REGISTROS
SELECT *
  FROM TABLA;
```

```
--SOLO 2 CAMPOS
--TODOS LOS REGISTROS
SELECT CAMPO1, CAMPO2
  FROM TABLA;
```

```
--TODOS LOS CAMPOS
--FILTRO REGISTROS
SELECT *
  FROM TABLA
 WHERE CAMPO = VALOR;
```

```
--SOLO 2 CAMPOS
--FILTRO REGISTROS
--ORDENADO
SELECT CAMPO1, CAMPO2
  FROM TABLA
 WHERE CAMPO IS NULL
 ORDER BY CAMPO1;
```

DML – TABLA TEST

TEST

Columnas | Restricciones | Permisos | Es

Refrescar: 0

	COLUMN_NAME	DATA_TYPE
1	CAMPO1_TXT	VARCHAR2(10 BYTE)
2	CAMPO2_NUM	NUMBER
3	CAMPO3_DAT	DATE

DML – SELECT (EJEMPLOS)

```
SELECT *
  FROM TEST;

SELECT CAMPO1_TXT, CAMPO3_DAT, CAMPO2_NUM
  FROM TEST
 WHERE CAMPO2_NUM = 999
    OR CAMPO2_NUM = 888;

--CONCATENACION
SELECT 'Hoy es ' || TO_CHAR(SYSDATE,'DAY')
  FROM DUAL;
```

FUNCIONES UTILES

- NVL
- TO_CHAR
- TO_DATE
- TO_NUMBER
- SUBSTR
- TRIM
- UPPER / LOWER / INITCAP

http://www.techonthenet.com/oracle/functions/

NVL FUNCTION

NVL(string1, replace_with)

string1
The string to test for a null value.
replace_with
The value returned if string1 is null.

```
--EJEMPLO
SELECT NVL(STATE_PROVINCE,'DESCONOCIDA')
  FROM LOCATIONS;
```

TO_CHAR FUNCTION

TO_CHAR(value [, format_mask] [, nls_language])

value
A number or date that will be converted to a string.
format_mask
Optional. This is the format that will be used to convert value to a string.
nls_language
Optional. This is the nls language used to convert value to a string.

```
--EJEMPLO
SELECT SALARY,
       TO_CHAR(SALARY),
       TO_CHAR(SALARY,'$99,999.00'),
       TO_CHAR(SALARY,'$9,999.00'),
       HIRE_DATE,
       TO_CHAR(HIRE_DATE,'MONTH'),
       TO_CHAR(HIRE_DATE,'MONTH','nls_date_language=English')
  FROM EMPLOYEES;
```

TO_DATE FUNCTION

TO_DATE(string1 [, format_mask] [, nls_language])

string1
The string that will be converted to a date.
format_mask
Optional. This is the format that will be used to convert string1 to a date. *http://www.techonthenet.com/oracle/functions/to_date.php*
nls_language
Optional. This is the nls language used to convert string1 to a date.

```
--EJEMPLO
SELECT TO_DATE('20160101','YYYYMMDD')
  FROM DUAL;
```

TO_NUMBER FUNCTION

TO_NUMBER(string1 [, format_mask] [, nls_language])

string1
The string that will be converted to a number.
format_mask
Optional. This is the format that will be used to convert string1 to a number.
nls_language
Optional. This is the nls language used to convert string1 to a number.

```
--EJEMPLO
SELECT POSTAL_CODE,TO_NUMBER(POSTAL_CODE)
  FROM LOCATIONS
 WHERE CITY = 'Roma' ;
```

SUBSTR FUNCTION

SUBSTR(string, start_position [, length])

string
The source string.
start_position
The starting position for extraction. The first position in the string is always 1.
length
Optional. It is the number of characters to extract. If this parameter is omitted, the SUBSTR function will return the entire string.

```
--EJEMPLO
SELECT STREET_ADDRESS,
       SUBSTR(STREET_ADDRESS,5),
       SUBSTR(STREET_ADDRESS,5,3)
  FROM LOCATIONS;
```

DML – OPERADORES

--OPERADORES DE COMPARACION

=	igual
<>	distinto
>	mayor
<	menor
>=	mayor o igual
<=	menor o igual

LIKE '%STRING%'

NOT LIKE '%STRING%'

```
--EJEMPLO
SELECT *
  FROM EMPLOYEES
 WHERE FIRST_NAME LIKE 'Da%'
   AND SALARY >= 7000;
```

DML – CONDICIONES

```
WHERE condition1
  AND condition2
  ...
   OR condition_n;
```

- Las sentencias AND & OR permiten probar múltiples condiciones simultáneamente.
- Es muy importante el orden de los paréntesis para agrupar.

```
--EJEMPLO
SELECT *
  FROM EMPLOYEES
 WHERE ( FIRST_NAME LIKE 'Da%'
         AND
         SALARY >= 7000 )
    OR JOB_ID = 'AD_VP';
```

A	B	A AND B	A OR B	NOT A
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

DML – ORDER BY

```
SELECT expressions
  FROM tables
[WHERE conditions]
 ORDER BY expression [ ASC | DESC ];
```

expressions
The columns or calculations that you wish to retrieve.
tables
The tables that you wish to retrieve records from. There must be at least one table listed in the FROM clause.
WHERE conditions
Optional. The conditions that must be met for the records to be selected.
ASC (DEFAULT)
Optional. It sorts the result set in ascending order by expression (default, if no modifier is provider).
DESC
Optional. It sorts the result set in descending order by expression.

```
SELECT *
  FROM EMPLOYEES
 ORDER BY 2,6 DESC;
```

```
SELECT *
  FROM EMPLOYEES
 ORDER BY FIRST_NAME,HIRE_DATE DESC;
```

DML - INSERT

```
INSERT INTO TABLA
VALUES (VALOR1,VALOR2,VALOR3);

INSERT INTO TABLA (CAMPO1,CAMPO2,CAMPO3)
VALUES (VALOR1,VALOR2,VALOR3);

INSERT INTO TABLA (CAMPO1,CAMPO2)
VALUES (VALOR1,VALOR2);
```

DML – TABLA TEST

TEST

Columnas | Restricciones | Permisos | Es

Refrescar: 0

	COLUMN_NAME	DATA_TYPE
1	CAMPO1_TXT	VARCHAR2(10 BYTE)
2	CAMPO2_NUM	NUMBER
3	CAMPO3_DAT	DATE

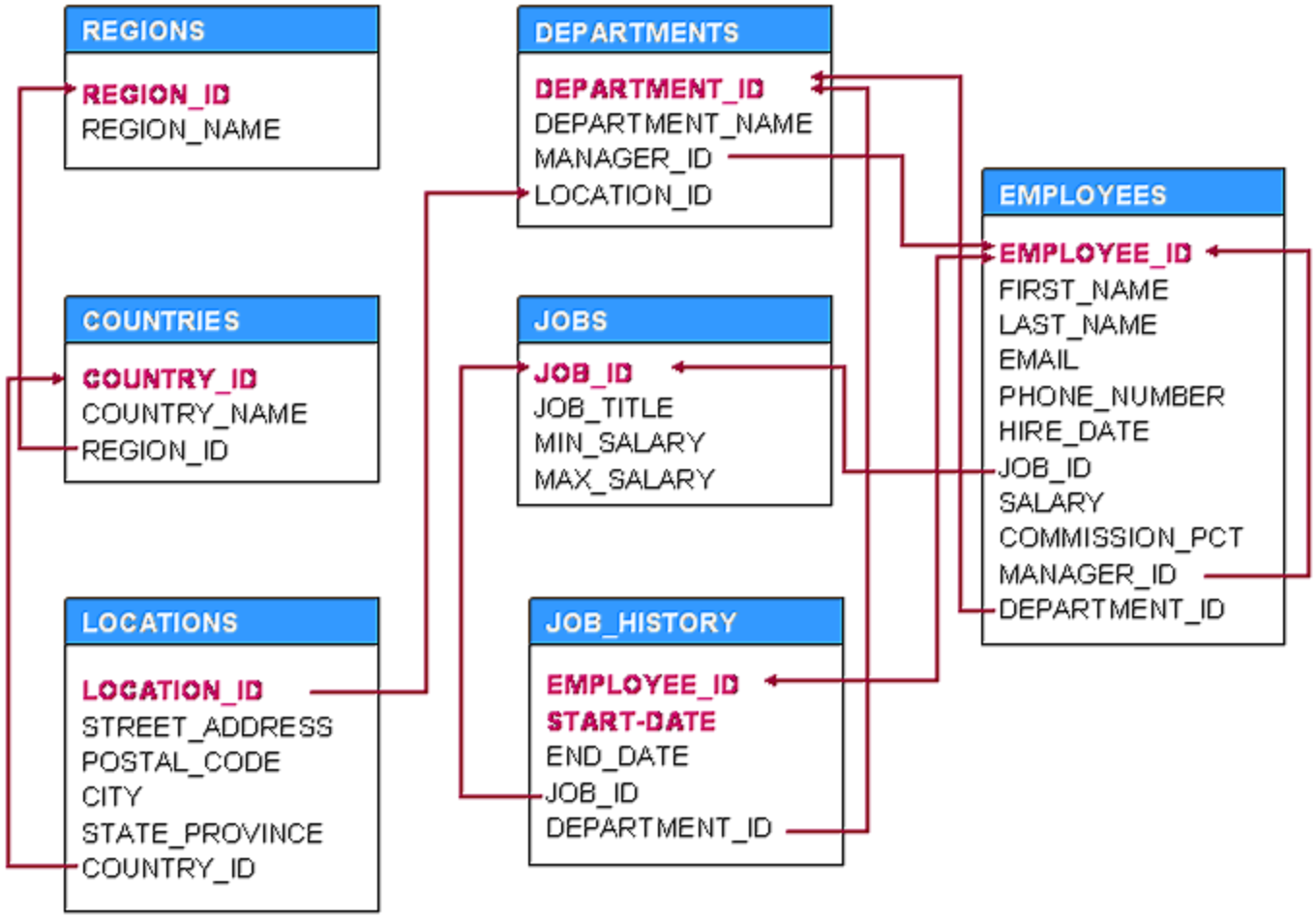
DML – INSERT (EJEMPLOS)

```
--INSERTS VARIOS
INSERT INTO TEST
VALUES ('ABC',123,SYSDATE);

INSERT INTO TEST (CAMPO1_TXT, CAMPO2_NUM)
VALUES ('DEF',456);

INSERT INTO TEST (CAMPO2_NUM,CAMPO1_TXT)
VALUES (789,'GHI');

INSERT INTO TEST
VALUES ('ABC',123,NULL);
```

DML – INSERT (EJEMPLOS)

```
--INSERTS ERRONEOS
INSERT INTO TEST
VALUES ('ABC',123);
Error SQL: ORA-00947: not enough values

INSERT INTO TEST (CAMPO2_NUM,CAMPO1_TXT)
VALUES ('DEF',456);
Error SQL: ORA-01722: invalid number

--AUTOCONVERSION
INSERT INTO TEST (CAMPO2_NUM,CAMPO1_TXT)
VALUES ('111',999);
```

DML – INSERT (EJERCICIO)



DML – INSERT (EJERCICIO)

1. Insertar la región Oceania – Tabla **REGIONS**
2. Insertar países New Zeland y Samoa de Oceania – Tabla **COUNTRIES**
3. Insertar las oficinas de Melbourne y Kelston – Tabla **LOCATIONS**

REGIONS
ID: 5 NOMBRE: Oceania

COUNTRIES
ID: NZ NOMBRE: New Zealand ID REGION: 5 ID: SA NOMBRE: Samoa ID REGION: 5

LOCATIONS
ID: 2201 DIRECCION: 367 Collins Street CP: 3000 CIUDAD: Melbourne PROVINCIA: Victoria ID PAIS: AU ID: 2250 DIRECCION: 65 Archibald Road CP: 0602 CIUDAD: Kelston PROVINCIA: Auckland ID PAIS: NZ

```
INSERT INTO TABLA (CAMPO1,CAMPO2)
VALUES (VALOR1,VALOR2);
```

DML - UPDATE

```
UPDATE TABLA
   SET CAMPO_X = VALOR_X;

UPDATE TABLA
   SET CAMPO_X = VALOR_X
 WHERE CAMPO = VALOR;

UPDATE TABLA
   SET CAMPO_X = VALOR_X
 WHERE CAMPO1 = VALOR1
    OR CAMPO2 = VALOR2;
```

DML – TABLA TEST

TEST

Columnas | Restricciones | Permisos | Es

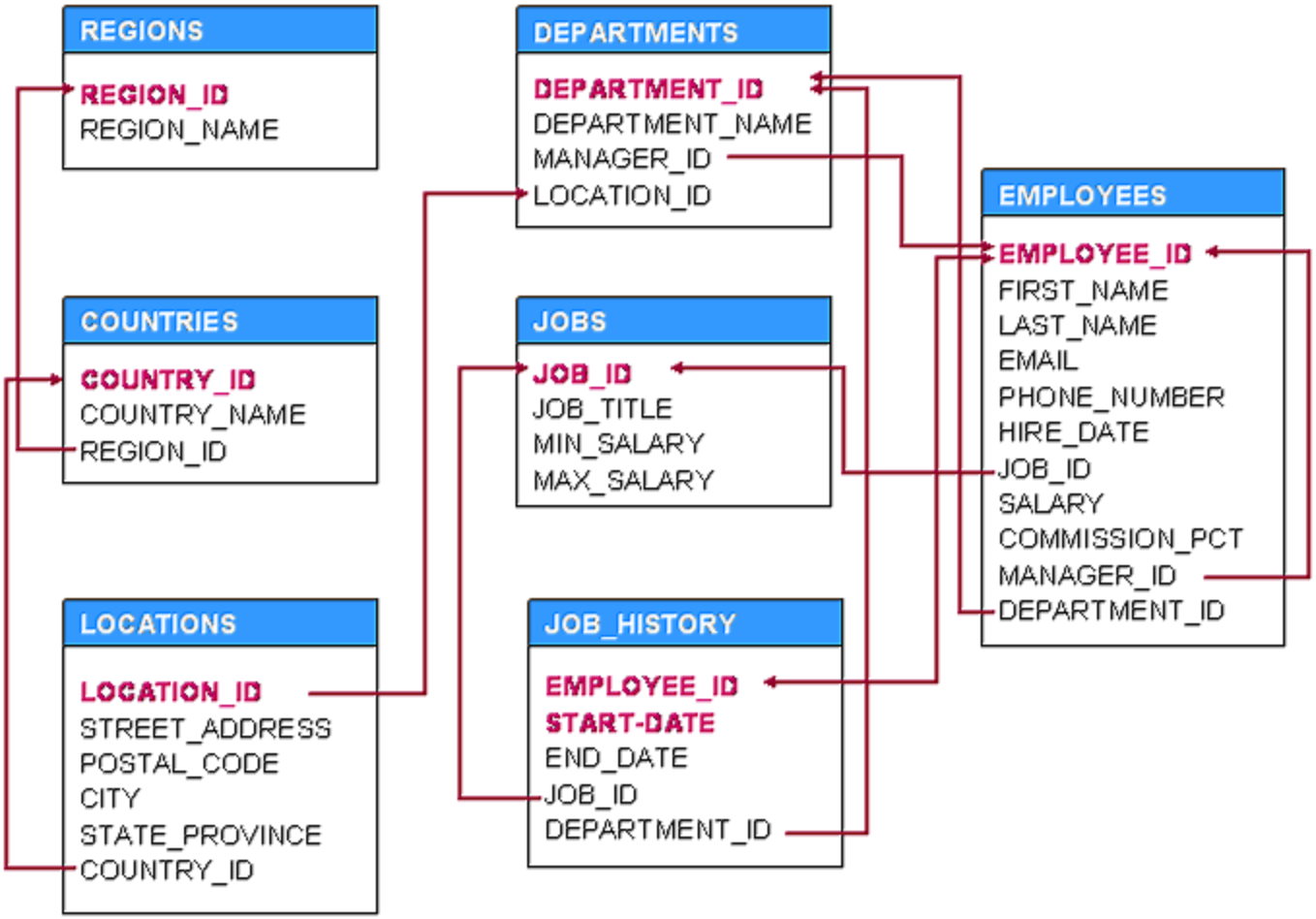
Refrescar: 0

	COLUMN_NAME	DATA_TYPE
1	CAMPO1_TXT	VARCHAR2(10 BYTE)
2	CAMPO2_NUM	NUMBER
3	CAMPO3_DAT	DATE

DML – UPDATE (EJEMPLOS)

```
UPDATE TEST
   SET CAMPO3_DAT = SYSDATE + INTERVAL '6' MONTH
 WHERE CAMPO1_TXT = 'ABC';

UPDATE TEST
   SET CAMPO2_NUM = TO_NUMBER(TO_CHAR(CAMPO3_DAT,'YYYYMMDD'))
 WHERE CAMPO1_TXT = 'ABC';
```

DML – UPDATE (EJERCICIO)



DML – UPDATE (EJERCICIO)

1. Actualizar la región de Australia a Oceania – Tabla **COUNTRIES**
2. Actualizar el código postal de la oficina de Londres (EC4R 9AB) – Tabla **LOCATIONS**

COUNTRIES
NOMBRE: Australia ID REGION: 5

LOCATIONS
ID: 2400 CP: 'EC4R 9AB' CIUDAD: London ID PAIS: UK

```
UPDATE TABLA
   SET CAMPO_X = VALOR_X
 WHERE CAMPO = VALOR;
```

DML - DELETE

```
DELETE
   FROM TABLA;

DELETE
   FROM TABLA
  WHERE CAMPO = VALOR;
```

DML – DELETE (EJEMPLOS)

```
DELETE
  FROM TEST_SOURCE;

DELETE
  FROM TEST
 WHERE CAMPO1_TXT = 'ABC';
```

COMMIT / ROLLBACK

COMMIT

Graba en forma permanente los datos en las tablas.

ROLLBACK

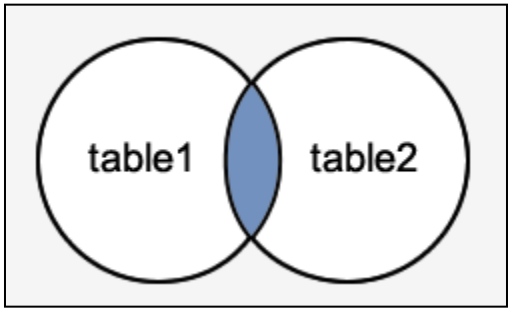
Deshace los cambios realizados recientemente (desde el ultimo commit).

DML – DELETE vs TRUNCATE

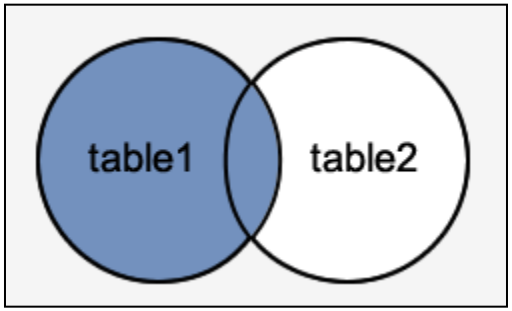
```
DELETE FROM TEST_SOURCE;

TRUNCATE TABLE TEST_SOURCE;
```

DML – JOINS

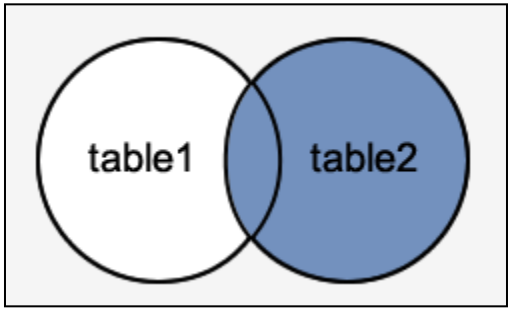
```
--INNER JOIN (SIMPLE JOIN)
SELECT columns
FROM table1
INNER JOIN table2
ON table1.column = table2.column;
```




```
--LEFT OUTER JOIN
SELECT columns
FROM table1
LEFT [OUTER] JOIN table2
ON table1.column = table2.column;
```




```
--RIGHT OUTER JOIN
SELECT columns
FROM table1
RIGHT [OUTER] JOIN table2
ON table1.column = table2.column;
```



DML – JOINS (EJEMPLOS)

```
SELECT *
  FROM REGIONS;
```

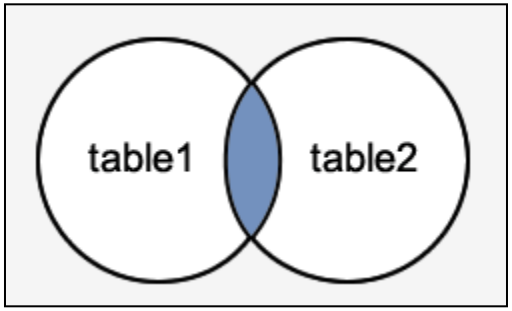
REGION_ID	REGION_NAME
1	Europe
2	Americas
3	Asia
4	Middle East and Africa
999	Antartida

```
SELECT *
  FROM COUNTRIES
 ORDER BY NVL(REGION_ID,0),
          COUNTRY_NAME;
```

COUNTRY_ID	COUNTRY_NAME	REGION_ID
AL	Albania	(null)
BE	Belgium	1
DK	Denmark	1
FR	France	1
DE	Germany	1
IT	Italy	1
NL	Netherlands	1
CH	Switzerland	1
UK	United Kingdom	1
AR	Argentina	2
BR	Brazil	2
CA	Canada	2

DML – INNER JOIN

```
--INNER JOIN (SIMPLE JOIN)
SELECT columns
FROM table1
INNER JOIN table2
ON table1.column = table2.column;
```

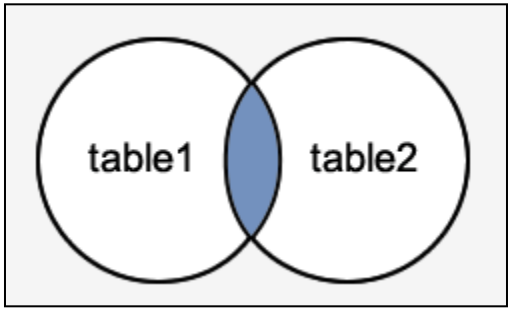



```
--EJEMPLO
SELECT R.REGION_NAME,
       C.COUNTRY_NAME
  FROM REGIONS R
 INNER JOIN COUNTRIES C
    ON R.REGION_ID = C.REGION_ID
 ORDER BY R.REGION_ID,
          C.COUNTRY_NAME;
```

REGION_NAME	COUNTRY_NAME
Europe	Belgium
Europe	Denmark
Europe	France
Europe	Germany
Europe	Italy
Europe	Netherlands
Europe	Switzerland
Europe	United Kingdom
Americas	Argentina
Americas	Brazil
Americas	Canada
Americas	Mexico
Americas	United States of America
Asia	Australia
Asia	China

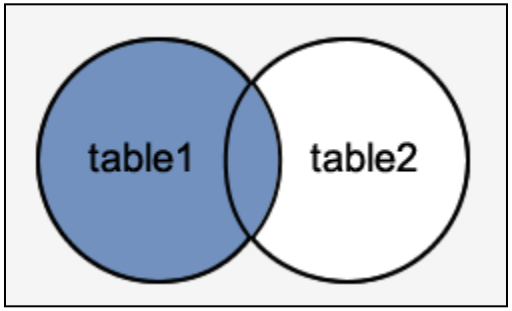
DML – INNER JOIN

```
-- SINTAXIS 1
SELECT R.REGION_NAME,
       C.COUNTRY_NAME
  FROM REGIONS R
 INNER JOIN COUNTRIES C
    ON R.REGION_ID = C.REGION_ID
 ORDER BY R.REGION_ID,
          C.COUNTRY_NAME;

-- SINTAXIS 2
SELECT R.REGION_NAME,
       C.COUNTRY_NAME
  FROM REGIONS R,
       COUNTRIES C
 WHERE R.REGION_ID = C.REGION_ID
 ORDER BY R.REGION_ID,
          C.COUNTRY_NAME;
```



DML – LEFT OUTER JOIN

```
--LEFT OUTER JOIN
SELECT columns
FROM table1
LEFT [OUTER] JOIN table2
ON table1.column = table2.column;
```

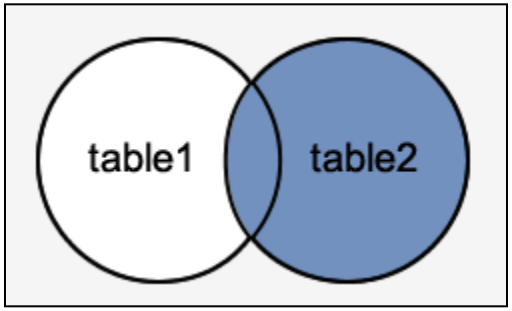



```
--EJEMPLO
SELECT R.REGION_NAME,
       C.COUNTRY_NAME
  FROM REGIONS R
  LEFT OUTER JOIN COUNTRIES C
    ON R.REGION_ID = C.REGION_ID
 ORDER BY R.REGION_ID,
          C.COUNTRY_NAME;
```

REGION_NAME	COUNTRY_NAME
Antartida	(null)
Europe	Belgium
Europe	Denmark
Europe	France
Europe	Germany
Europe	Italy
Europe	Netherlands
Europe	Switzerland
Europe	United Kingdom
Americas	Argentina
Americas	Brazil
Americas	Canada
Americas	Mexico
Americas	United States of America
Asia	Australia

DML – RIGHT OUTER JOIN

```
--RIGHT OUTER JOIN
SELECT columns
FROM table1
RIGHT [OUTER] JOIN table2
ON table1.column = table2.column;
```




```
--EJEMPLO
SELECT R.REGION_NAME,
       C.COUNTRY_NAME
  FROM REGIONS R
 RIGHT OUTER JOIN COUNTRIES C
    ON R.REGION_ID = C.REGION_ID
 ORDER BY R.REGION_ID,
          C.COUNTRY_NAME;
```

REGION_NAME	COUNTRY_NAME
(null)	Albania
Europe	Belgium
Europe	Denmark
Europe	France
Europe	Germany
Europe	Italy
Europe	Netherlands
Europe	Switzerland
Europe	United Kingdom
Americas	Argentina
Americas	Brazil
Americas	Canada
Americas	Mexico
Americas	United States of America
Asia	Australia