SQL from scratch

BoC Tech Talks

Nicolás Brailovsky

October 15, 2010

Oh, yes. Little Bobby tables we call him.

– XKCD

Outline

Historia

Structured Query Language:

- Es un hijo de. . . Prolog (Prolog -> Datalog -> SQL)
- Es declarativo
- Después se agregó un hack procedural. Se llama PLSQL.

Motores

SQL es una opción entre muchas

Big hashmap	Object DB	RDBMS
Berkeley DB (Ahora Oracle)	Gemstone	MySQL (Ahora Oracle)
Mongo DB (En serio)	Db4O	PostgreSQL

Outline

DML vs DDL

Hay dos tipos de sentencias SQL:

DML	DDL
Data Manipulation Language	Data Definition Language
Consultas a los datos	Metadata, definición de datos
Bastante estándar	Rezá por tener un buen manual
No debugeable	No debugeable

DML vs DDL

Hay dos tipos de sentencias SQL:

DML	DDL
Data Manipulation Language	Data Definition Language
Consultas a los datos	Metadata, definición de datos
Bastante estándar	Rezá por tener un buen manual
No debugeable	No debugeable

Vamos a usar DMLs el 99% del tiempo

DMLs

Hay cuatro tipos de DMLs:

- DELETE
- INSERT
- UPDATE
- SELECT

Deletes

Es el más simple:

- DELETE FROM tabla WHERE condicion

Tip: primero el where, después el delete.

Deletes

Es el más simple:

- DELETE FROM tabla WHERE condicion

Tip: primero el where, después el delete. Si, por experiencia.

Where

Los where también son simples

- WHERE (condiciones)
- Las condiciones son booleans
- WHERE x=1 AND (y=2 OR z=3)

Tip: El elemento neutro ahorra código.

Where

Los where también son simples

- WHERE (condiciones)
- Las condiciones son booleans
- WHERE x=1 AND (y=2 OR z=3)

Tip: El elemento neutro ahorra código. Un ejemplo?

Where

Los where también son simples

- WHERE (condiciones)
- Las condiciones son booleans
- WHERE x=1 AND (y=2 OR z=3)

Tip: El elemento neutro ahorra código.

- WHERE false OR (x=4 AND z=2) OR (x=42 AND z=42)
- WHERE user_id IN (NULL, 1, 2, 3);

Where

Los where también son simples

- WHERE (condiciones)
- Las condiciones son booleans
- WHERE x=1 AND (y=2 OR z=3)

Tip: El elemento neutro ahorra código.

- WHERE false OR (x=4 AND z=2) OR (x=42 AND z=42)
- WHERE user_id IN (NULL, 1, 2, 3);

Updates

Es el segundo más simple:

- UPDATE tabla SET key=val WHERE condicion

También sirve: primero el where, después el delete.

Insert

Es el tercero más simple (?). Ahora hay algunas variaciones:

- INSERT INTO tabla (col1, col2) VALUES (val1, val2);
- INSERT INTO tabla VALUES (val1, val2);
- INSERT INTO tabla SELECT (WTF!?)

Esta vez no hay tip loco

Insert

Es el tercero más simple (?). Ahora hay algunas variaciones:

- INSERT INTO tabla (col1, col2) VALUES (val1, val2);
  - No importa el orden
  - Puede estar incompleto
- INSERT INTO tabla VALUES (val1, val2);
- INSERT INTO tabla SELECT (WTF!?)

Insert

Es el tercero más simple (?). Ahora hay algunas variaciones:

- INSERT INTO tabla (col1, col2) VALUES (val1, val2);
  - No importa el orden
  - Puede estar incompleto
- INSERT INTO tabla VALUES (val1, val2);
  - Importa el orden
  - No puede estar incompleto
- INSERT INTO tabla SELECT (WTF!?)

Insert

Es el tercero más simple (?). Ahora hay algunas variaciones:

- INSERT INTO tabla (col1, col2) VALUES (val1, val2);
  - No importa el orden
  - Puede estar incompleto
- INSERT INTO tabla VALUES (val1, val2);
  - Importa el orden
  - No puede estar incompleto
- INSERT INTO tabla SELECT (WTF!?)
  - Ehh... lo vemos después

Insert II

Otra variación más:

```
INSERT INTO tabla (col1, col2) VALUES (val1, val2),
(bal1, bal2), (wal1, wal2);
```

Disminuye la carga en la DB!

Select

Es un bardo. Posta. Sirve para traer datos. Si, en serio.

En su forma APB:

- SELECT 42

Select

Es un bardo. Posta. Sirve para traer datos. Si, en serio.

O en su forma Oracle APB:

- SELECT 42
- SELECT 42 FROM DUAL (Gracias Oracle!)

Select

Es un bardo. Posta. Sirve para traer datos. Si, en serio.

Usando una función

- SELECT 42
- SELECT 42 FROM DUAL (Gracias Oracle!)
- SELECT NOW()

Select de verdad

```
SELECT * FROM users
```

Select de verdad

```
SELECT * FROM users
```

Si tenemos estos datos...

```
INSERT INTO USERS (name, age) VALUES
('Bart',10), ('Lisa',8),
('Maggie',2), ('Carl', 40);
INSERT INTO RELATIONSHIPS (user1_id, user2_id, rela
VALUES (1, 2, 'Hermanos'), (1,3, 'Hermanos'),
(2,1, 'Hermanos'), (2,3, 'Hermanos'),
(3,1, 'Hermanos'),(3,2, 'Hermanos');
```

Select de verdad II

```
SELECT *

FROM users

WHERE age < 10
```

#	id	name	age
1	2	Lisa	8
2	3	Maggie	2

Outline

Select de verdad III

Cómo encontramos todos los usuarios y sus hermanos?

Select de verdad III

Cómo encontramos todos los usuarios y sus hermanos?

```
SELECT *

FROM users Usr1

INNER JOIN relationships Rel

ON Rel.user1_id = Usr1.id

WHERE related = 'Hermanos'
```

Select de verdad III

Cómo encontramos todos los usuarios y sus hermanos?

name	age	id	user1_id	user2_id	related
Bart	10	1	1	2	Hermanos
Bart	10	2	1	3	Hermanos
Lisa	8	3	2	1	Hermanos
Lisa	8	4	2	3	Hermanos
Maggie	2	5	3	1	Hermanos
Maggie	2	6	3	2	Hermanos

Cómo nos traemos el nombre del otro hermano?

Select de verdad IV

- *Cómo encontramos todos los usuarios y sus hermanos?*
- *Cómo nos traemos el nombre del otro hermano?*

Select de verdad IV

- *Cómo encontramos todos los usuarios y sus hermanos?*
- *Cómo nos traemos el nombre del otro hermano?*

```
SELECT Usr1.name, Usr2.name

FROM users Usr1

INNER JOIN relationships Rel

ON Rel.user1_id = Usr1.id

INNER JOIN users Usr2

ON Rel.user2_id = Usr2.id

WHERE related = 'Hermanos'
```

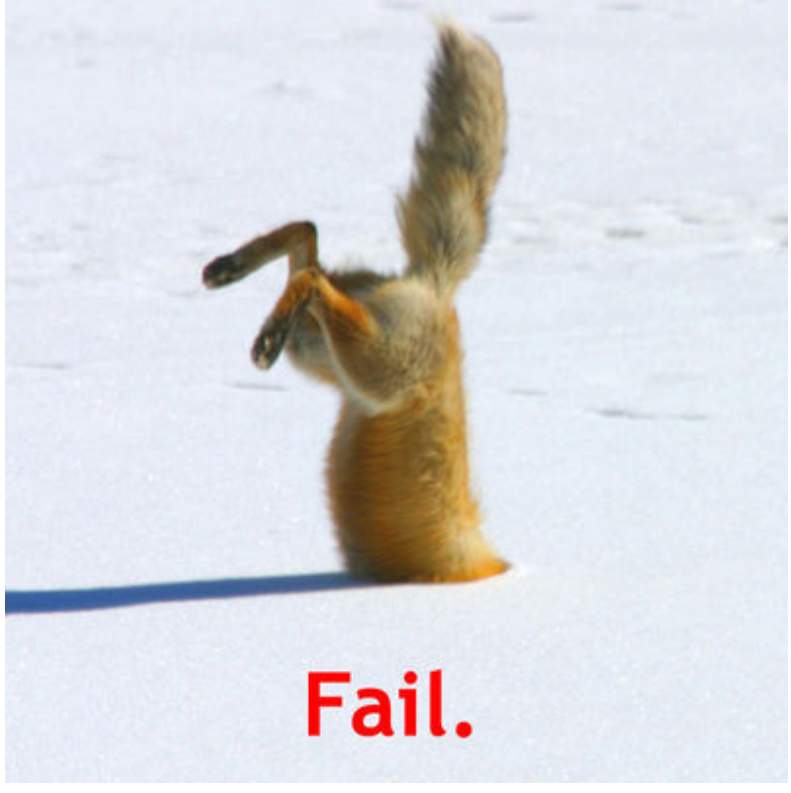
Select de verdad IV

- *Cómo encontramos todos los usuarios y sus hermanos?*
- *Cómo nos traemos el nombre del otro hermano?*

name	name
Bart	Lisa
Bart	Maggie
Lisa	Bart
Lisa	Maggie
Maggie	Bart
Maggie	Lisa

Select de verdad IV'

¿Qué pasa si yo quería TODOS los usuarios?
¿A dónde fue Carl?



Select de verdad IV

```
SELECT Usr1.name, Usr2.name

FROM users Usr1

LEFT JOIN relationships Rel

ON Rel.user1_id = Usr1.id

LEFT JOIN users Usr2 ON Rel.user2_id = Usr2.id
```

Select de verdad IV

name	name
Bart	Lisa
Bart	Maggie
Carl	NULL
Lisa	Bart
Lisa	Maggie
Maggie	Bart
Maggie	Lisa

Select de verdad V

¿Y si quiero traer sólo cuántos hermanos tienen?

Select de verdad V

```
SELECT Usr1.name, count(1) AS Hermanos

FROM users Usr1

LEFT JOIN relationships Rel

ON Rel.user1_id = Usr1.id

LEFT JOIN users Usr2 ON Rel.user2_id = Usr2.id

GROUP BY Usr1.id
```

Cuidado, COUNT+LEFT JOIN = cosas raras

Select de verdad V

name	Hermanos
Bart	2
Lisa	2
Maggie	2
Carl	1

Cuidado, COUNT+LEFT JOIN = cosas raras

Select de verdad VI

El GROUP tiene su WHERE, pero se llama HAVING

Select de verdad VI

```
SELECT Usr1.name, COUNT(1) AS Hermanos

FROM users Usr1

LEFT JOIN relationships Rel

ON Rel.user1_id = Usr1.id

LEFT JOIN users Usr2 ON Rel.user2_id = Usr2.id

GROUP BY Usr1.id

HAVING COUNT(1) > 1
```

Select de verdad VII, sólo para complicarla

```
SELECT Usr1.name, COUNT(1) AS Hermanos

FROM users Usr1

LEFT JOIN relationships Rel

ON Rel.user1_id = Usr1.id

LEFT JOIN users Usr2 ON Rel.user2_id = Usr2.id

GROUP BY Usr1.id

HAVING COUNT(1) > 1

ORDER BY Usr1.name LIMIT 3
```

Outline

COALESCE, NVL, IFNULL

- Son lo mismo pero cambia según el motor
- Es un if trucho
- IFNULL(A, B) <=> A!=NULL? A : B

Unions

Joins raros

Subqueries