Insertions

With `INSERT INTO` you can insert tuples providing all fields:

```
INSERT INTO users VALUES (0, 'Paul Garcia');
INSERT INTO users VALUES (1, 'Joel Spolsky');
```

If some fields can be omitted, we can indicate the given fields:

```
INSERT INTO users (name) VALUES ('Mark Zuckerberg');
```

Multiple Insertions

After the `VALUES` clause, we can use commas to enumerate many tuples to insert

```sql
INSERT INTO emails (user_id, email) VALUES
  (0, 'paul@gmail.com'),
  (0, 'pgarcia@hotmail.com'),
  (1, 'joel@spolsky.com'),
  (2, 'zuck@gmail.com'),
  (2, 'mark@facebook.com'),
  (2, 'm@zuckerberg.com');
```

Reads

With SELECT, we can list all tuples in a relation:

```
SELECT * FROM users;
```

We can choose which colums we want to see:

```
SELECT name, id FROM users;
```

To show only certain entities satisfying a condition, we can add a WHERE clause

```
SELECT name FROM users WHERE id = 1;
```

Deletions

With `DELETE FROM` we erase *all* elements in a relation

```
DELETE FROM users; /* oops, everything gone... */
```

Using a `WHERE` clause, we can choose which elements to delete

```
DELETE FROM users WHERE id = 2; /* Zuck is gone */
```

Updates

With `UPDATE` we can update entities

```
UPDATE users    /* same name everybody... */
  SET name = 'John Doe';
```

A `WHERE` clause chooses what entities to update

```
UPDATE users
  SET name = 'Zuck'
  WHERE name = 'Mark Zuckerberg';
```

RETURNING 1

Many commands will change data and we want to know the latest value:

```
INSERT INTO users (name) VALUES ('Jose Muñoz');
/* What is the assigned ID??? */
SELECT id FROM users WHERE name = 'Jose Muñoz';
/* -> 13 */
```

The **RETURNING** clause (ANSI SQL) allows us to collapse both commands:

```
INSERT INTO users (name)
  VALUES ('Jose Muñoz')
  RETURNING id;
```

RETURNING 2

RETURNING works with INSERT, UPDATE and DELETE.

```
INSERT INTO users (name)
  VALUES ('Richard Feynman')
  RETURNING *;
```

```
UPDATE users SET name = upper(name)
  WHERE id <> 10
  RETURNING id, name;
```

```
DELETE FROM users
  WHERE name like 'Z%'
  RETURNING id, name;
```