

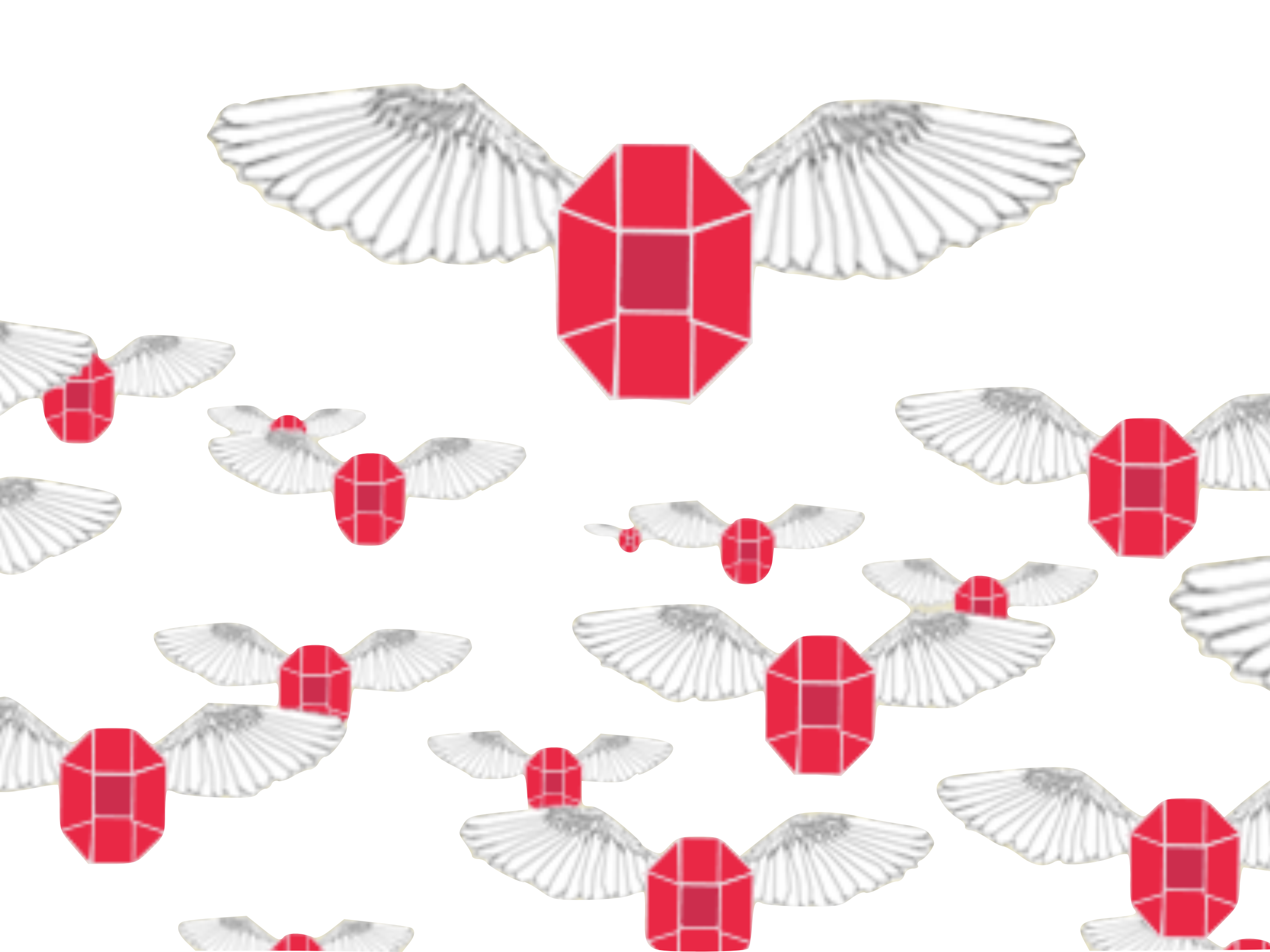


Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

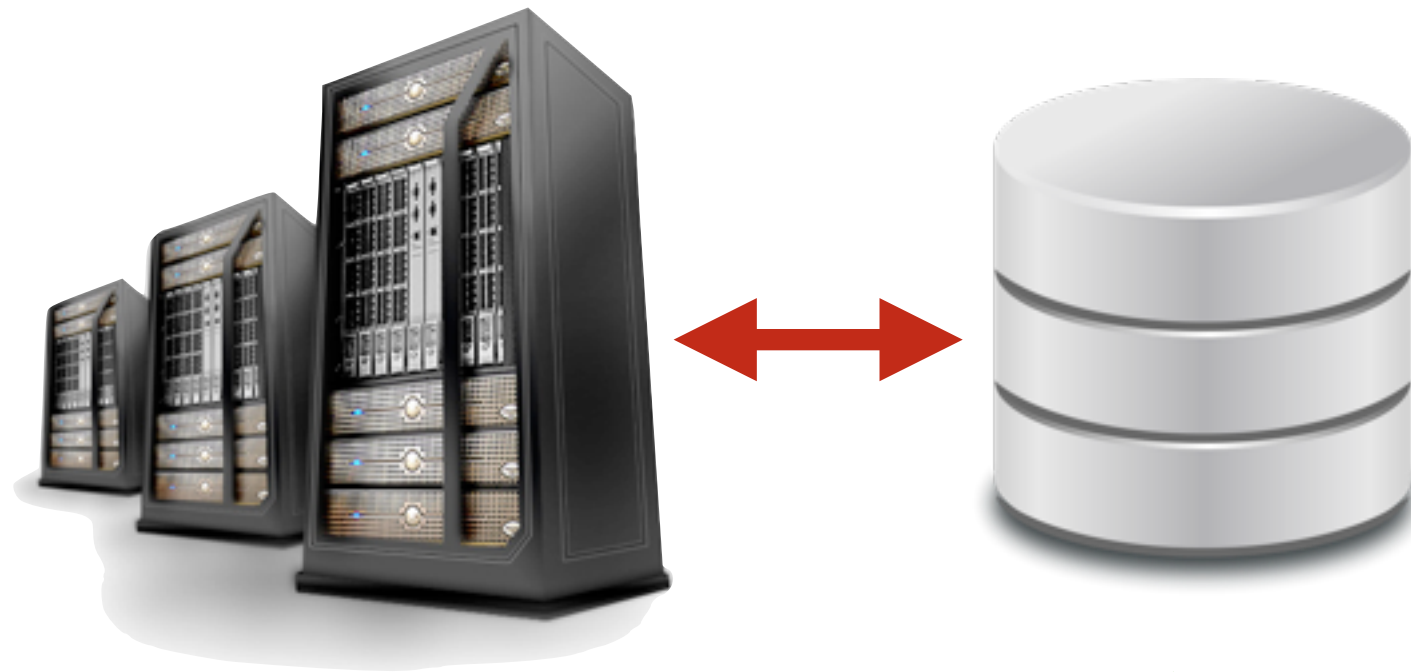
RAILS

MIGRATIONS

Raúl Montes T.



Back-end



BD relacional

Rails supone una BD relacional como motor de persistencia

El esquema de la BD va cambiando en el tiempo...

... tanto en etapas de desarrollo como en
futuras mantenciones

Las migraciones son scripts Ruby que nos permiten realizar todo tipo de cambios en la BD

crear/eliminar/cambiar tablas

crear/eliminar/cambiar columnas

crear/eliminar/cambiar datos

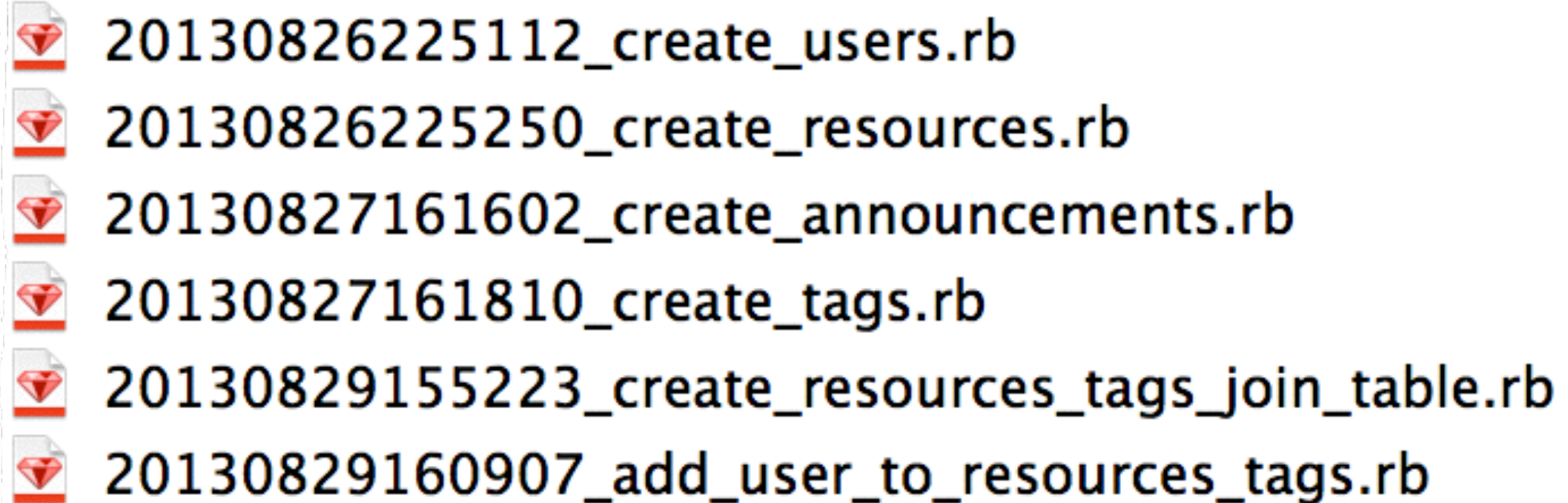
... en realidad... lo que queremos...

Antiguamente...



timestamp

nombre expresivo



- 20130826225112_create_users.rb
- 20130826225250_create_resources.rb
- 20130827161602_create_announcements.rb
- 20130827161810_create_tags.rb
- 20130829155223_create_resources_tags_join_table.rb
- 20130829160907_add_user_to_resources_tags.rb

Nos permiten...

- Que nuestra BD migre de un estado consistente a otro
- Tener registros de los cambios
- Que todos los desarrolladores puedan transformar la BD de la misma forma
- Realizar procesos migratorios más complejos (generar datos, traspasar info antes de eliminarla, etc.)

Principales métodos a definir en una migración

- up: se ejecuta al migrar
- down: se ejecuta al ejecutar un rollback de la migración
- change: la fusión de ambas. Se especifica sólo lo que se hace en up y funciona sólo si Rails puede “intuir” el rollback *automáticamente*

Principales métodos para manipular BD

- `create_table`: define una tabla
- `add_column`: agrega una columna (si hay que agregarla después de la creación)
- `add_index`: crea un índice
- `drop_table`: elimina una tabla
- `remove_column`: elimina una columna
Revisar `SchemaStatements` y `TableDefinition`