Cheatography

Active Record Migrations Cheat Sheet

by software_writer via cheatography.com/182173/cs/40093/

Update Table

```
change_table :products do |t|
    t.r emove :descr iption, :name
    t.s tring :servi ce_ number
    t.index :servi ce_ number
    t.r ename :creator, :maker
end
```

Use the `change _table` method to update existing tables. Like create_table, the change_table method accepts the name of the table and a block to add columns or indexes to the table:

Rename Table

```
rename_table('octopuses', 'octopi')
```

Use the `rename _ta ble (old, new)` to rename a table.

Drop Table

```
def change
    dro p_table :accounts
    dro p_table :posts do |t|
        t.s tring :title
        t.text :content
   end
end
```

To drop an existing table, you can use the drop_table method. It ignores the provided block and options when running the migration, but uses them when reverting the migration to generate the corresponding table.

Create Columns

```
# add_column(table_name, column_name, type,
**options)
add_co lum n(: users, :picture, :binary)
```

You can specify new columns while creating a new table. But if you want to add new columns to an existing table, use the add_column method.

Modify Columns

```
# change_column(table_name, column_name, type,
**options)
change _column :users, :name, :string, limit: 80
```

The `change _column` method allows you to modify the type and attributes of existing columns.

Rename Column

```
# rename_column(table_name, column_name,
new_column_name)
rename _co lum n(: sup pliers, :descr iption,
:name)
```

To rename a column, use the `rename _column` method.

Remove Column

```
remove_column :users, :password
```

To drop a column, you may use the remove_column method.

Referencing Tables

```
def change
    add _re ference :posts, :user # or

   # add_be lon gs_to :posts, :user
end
change _table :posts do |t|
    t.r efe rences :user # or
   # t.belo ngs_to :user
end
```

To add foreign key, use the `add_re ference :table _name, :reference` method. For example, if the posts table references a user, then posts will have a `user_id` column in it.

Migration Commands

rails generate migration	Adds a new migration
rails migrate	Applies the latest migration(s)
rails db:rollback STEP=n	Revert previous (n) migrations
rails db:migrate:redo STEP=n	Rollback and migrate the database
rails db:reset	Drop and recreate the database

Data Types

:integer	whole number without any fractions
:float	floating point number (do not use for calculations that need accuracy)
:decimal	specific precisions, used for financial calculations that need accuracy
:bigint	very large integer
:boolean	true / false
:string	limited to 255 characters.

Cheatography

Active Record Migrations Cheat Sheet

by software_writer via cheatography.com/182173/cs/40093/

Data Types (cont)

:text	unlimited (depending on the database)
:date	year, month, and day without time
:time	hour, minutes, and seconds wihout date
:datetime / :timestamp	date and time
:json	JSON data
:binary	raw binary data

Column Types

Type	MySQL	SQLite	PostgreSQL
:integer	int(11)	integer	integer
:bigint	?	?	?
:float	float	float	float
:decimal	decimal	decimal	decimal
:boolean	boolean	boolean	boolean
string	varchar (255)	varchar (255)	character varying (255)
:text	text	text	text
:date	date	date	date
:time	time	time	time
:datetime	:datetime	datetime	timestamp
:timestamp	datetime	datetime	timestamp
:json	?	?	?
:binary	blob	blob	bytea
:virtual	?	?	?

Create Index

```
# while creating a table
create _table :users do |t|
    t.s tring :name, index: true
    t.s tring :email, index: { unique: true }
    t.index [:title, :address], unique: true
end
# adds a new index on the  posts table for the titl
e column.
add_index :posts, :title
# adds a new index for the  company_id and manager_i
d columns;
```

Create Index (cont)

```
> add_index :suppliers, [:company_id, :manager_id]
```

ActiveRecord supports several types of indexes. Use the add_index or t.index method to add new index. Additionally, you can mark the index as unique or give it a different name.

Rename Index

```
rename_index :people, 'index_people_on_last_name',
'index_users_on_last_name'
# OR
change _table :users do |t|
    t.r ena me_ ind ex( :us er_id, :accou nt_id)
end
```

To rename an index, use the `rename _index` method. This method accepts the table name as its first argument and the current index name and the desired name as its second and third arguments, respectively.

Alternatively, use the `rename _index` method while changing a table. This is useful when you want to change multiple index names in a single migration.

Drop Index

```
remove_index :accounts, :branch_id
remove _index :accounts, column: [:bran ch_id,
:party_id]
remove _index :accounts, name: :by_br anc h_party
# or
change _table :users do |t|
    t.r emo ve_ ind ex( :br anc h_id)
    t.r emo ve_ ind ex( column: [:bran ch_id,
:party _id])
end
```

To drop an index, you may use the `remove _index` method, passing the name of the table and either the column name or the name of the index. Alternatively, use the `remove _index` method while changing the table.