

Update Table

```
change_table :products do |t|
  t.remove :description, :name
  t.string :service_number
  t.index :service_number
  t.rename :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_table` (old, new) to rename a table.

Drop Table

```
def change
  drop_table :accounts
  drop_table :posts do |t|
    t.string :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_column(: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_column(:suppliers, :description,
: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_reference :posts, :user # or

  # add_belongs_to :posts, :user
end

change_table :posts do |t|
  t.references :user # or
  # t.belongs_to :user
end
```

To add foreign key, use the `add_reference :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

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

Data Types

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



Data Types (cont)

:text	unlimited (depending on the database)
:date	year, month, and day without time
:time	hour, minutes, and seconds without 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.string :name, index: true
  t.string :email, index: { unique: true }
  t.index [:title, :address], unique: true
end

# adds a new index on the posts table for the title column.
add_index :posts, :title

# adds a new index for the company_id and manager_id 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.rename_index(:user_id, :account_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: [:branch_id, :party_id]
remove_index :accounts, name: :by_branch_party
# or
change_table :users do |t|
  t.remove_index(:branch_id)
  t.remove_index(column: [:branch_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.