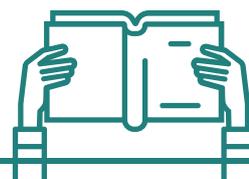


Waterline

Sails.js Cheat Sheet



Ordem	Finalidade	Comando	Observação
<i>As configurações de armazenamento estão localizadas no arquivo 'config/datastores.js' e podem ser acessadas pelo código como sails.config.datastores.<nome dicionário>.</i>			
1.	Localiza lista de registros que atendam ao critério.	<pre>await Modelo.find(critério);</pre> ou <pre>await Modelo.find({ where: {id: 42}, select: ['name', 'email'] });</pre>	O critério deve ser um objeto contendo um campo válido ou expressão. Ex.: {id: 42}. As respostas sempre serão JavaScript Promisses, portanto, pode-se utilizar tanto o await quanto o then().catch()
2.	Localiza um único registro que atenda ao critério.	<pre>await Modelo.findOne({});</pre>	Retorna <i>undefined</i> se nenhum registro correspondente for encontrado.
3.	Tenta localizar, mas cria um novo registro se não existir.	<pre>await Modelo .findOrCreate(critério, valoresIniciais);</pre> Ou <pre>Modelo .findOrCreate(critério, valoresIniciais) .exec(async((err, dado, criado)=> { if(criado) console.log('OK'); });</pre>	A função exec() retorna três valores para o <i>call-back</i> : erro (se houver ou <i>undefined</i>); o dado retornado (se for erro, retorna <i>undefined</i>) e um flag booleano informando se foi necessário criar o registro ou não.
4.	Localiza e retorna um conjunto de dados que atendam ao critério, em formato <i>Stream</i>	<pre>await Modelo.stream(criteria) .eachRecord(async (records)=>...)</pre> Ou <pre>Modelo .eachBatch(async (records)=>...</pre>	Isso é útil para trabalhar com conjuntos de resultados muito grandes, do tipo que pode transbordar a RAM disponível do servidor se você tentar manter todo o conjunto na memória ao mesmo tempo.
5.	Faz a validação de um valor para um determinado campo/ atributo	<pre>Modelo.validate(atributo, valor);</pre>	É um método assíncrono, portanto não necessita de await ou then(). Vide regras de validação mais abaixo

6.	Conta quantos registros atendem ao critério	<code>await Modelo.count(criterio);</code>	
7.	Cálculo de Valor Médio de valores nos registros que atendem ao critério	<code>await Modelo.avg(atributo, criterio);</code>	O atributo deve ser do tipo numérico
8.	Cálculo de Valor Somado de valores os registros que atendem ao critério	<code>await Modelo.sum(atributo, criterio);</code>	O atributo deve ser do tipo numérico
9.	Cria um ou mais registros no banco de dados	<code>await Modelo.create(valoresIniciais);</code> <code>await Modelo.createEach([valoresIniciais])</code>	valoresIniciais corresponde a um objeto contendo os pares atributo: valor do JavaScript.
10.	Remove um ou mais registros do banco de dados	<code>await Modelo.destroyOne(criterio)</code> <code>await Modelo.destroy(criterio)</code>	destroyOne() retornará erro se mais de um objeto atender ao critério;
11.	Atualiza um ou mais registros do banco de dados	<code>await Modelo.updateOne(criterio, valores)</code> <code>await Modelo.update(criterio, valores)</code>	updateOne() retornará erro se mais de um objeto atender ao critério; valores corresponde ao objeto contendo pares atributo:valor;

Dicas de Uso
Use: <code>Modelo.find({campo: criterio}).skip(numero).limit(numero)</code> para incorporar paginação
Use: <code>Modelo.[update() create() createEach() destroy()].fetch()</code> para recuperar o dado imediatamente.
Use: <code>Modelo.[método()].intercept(filtro, callback)</code> para capturar e manipular erros de banco de dados
Use: <code>Modelo.[método()].tolerate(filtro ou callback)</code> para tolerar um erro e retornar um novo valor.
Use: <code>Modelo.find({}).decrypt()</code> para decriptografar dados armazenados em formato encriptado
Use: <code>Modelo.find({}).populate('nome_do_atributo_que_representa_a_colecao')</code> para popular coleções. Ex. <code>Proprietario.find().populate('carros')</code>

Regras de Validação de Campos de Tabela / Atributos de Objeto Model	
Nome	Função:
<code>type</code>	Define o tipo da variável armazenada. Pode ser: string, number, Boolean, json e ref
<code>required</code>	Define se o campo é obrigatório (true) ou não (false). Pode existir com valor nulo
<code>allowNull</code>	Define se o campo obrigatório aceitará valores nulos. O padrão é false
<code>defaultsTo</code>	Valor atribuído como padrão, se omitido na criação
<code>isIn</code>	Verifica se o valor informado está contido na listagem.ex ['boring', 'happy', 'sad']
<code>unique</code>	Define que o valor atribuído deve ser único (true) na tabela.
<code>columnName</code>	Define o nome do campo da tabela que corresponde ao atributo do objeto
<code>columnType</code>	Define o tipo específico atributo do Modelo para o campo na tabela. Ex. 'CHAR(5)'
<code>autoIncrement</code>	Quando um registro é criado, o valor numérico do campo é incrementado de 1.
<code>encrypt</code>	Define se o campo deve ser Encriptado antes de ser armazenado no banco
<code>tableName</code>	Define o nome da tabela do banco de dados que corresponde ao objeto
Ex.:	nome: { type: 'string', required: true, unique: true, columnName: 'NM_USER'}, pais: { type: 'string', required: false, defaultsTo: 'brasileiro', encrypt: true }, nascimento: { type: 'number', columnType: 'date', columnName: 'DT_NASC' }