

Sequelize Migration

Sequelize-cli 로 DB 이력을 관리해보자

발표자 : 다람쥐

발표날짜 : 2021년 02월 20일 토요일




어느 날
사이드 프로젝트를 하던 중...



**: 추후에 로그인 들어가면
RDBMS 테이블 구조도 바뀔 것이고
현재 초기 데이터도 많이 넣어줘야 하는데
마이그레이션으로 관리하는 건 어떨까요~?**

 : 마이그레이션이 뭔가요?

 : ?

**코드도 깃으로 이력을
저장하는데**

**데이터베이스도
이력 관리가 필요하지
않겠습니까~?**

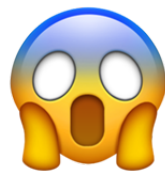
v 1.0

```
Emoji.init({
  id: {
    type: DataTypes.BIGINT.UNSIGNED,
    allowNull: false,
    autoIncrement: true,
    primaryKey: true,
  },
  unicode: {
    type: DataTypes.STRING(100),
    allowNull: false,
    unique: true,
  },
}, {
  sequelize,
  tableName: 'emojis',
  charset: 'utf8mb4',
  collate: 'utf8mb4_bin',
  timestamps: true,
  underscored: true,
});
```

v 1.1

```
Emoji.init({
  id: {
    type: DataTypes.BIGINT.UNSIGNED,
    allowNull: false,
    autoIncrement: true,
    primaryKey: true,
  },
  imageUrl: {
    type: DataTypes.STRING(255),
    allowNull: true,
  },
  unicode: {
    type: DataTypes.STRING(100),
    allowNull: false,
    unique: true,
  },
}, {
  sequelize,
  tableName: 'emojis',
  charset: 'utf8mb4',
  collate: 'utf8mb4_bin',
  timestamps: true,
  underscored: true,
});
```

기존 데이터가 있는데
테이블을 **드랍**하고 다시 만들기..?



실서버에서

**칼럼 추가/수정, 인덱스 추가,
초기 데이터 삽입을 과연 수동으로?**

답은 마이그레이션!

코드로 DB 이력을 관리할 수 있다!

sequelize-cli 으로 마이그레이션 생성 가능

`npx sequelize-cli migration:generate --name name`

`migrations` 폴더에 `{{TimeStamp}}-name.js` 파일 생성

칼럼 추가

```
'use strict';

module.exports = {
  up: async (queryInterface, Sequelize) => {
    /* Add altering commands here. */
    return queryInterface.addColumn(
      'countries',
      'image_url',
      {
        type: Sequelize.DataTypes.STRING,
        allowNull: true,
      },
    );
  },

  down: async (queryInterface, Sequelize) => {
    /* Add reverting commands here. */
    return queryInterface.removeColumn(
      'countries',
      'image_url',
    );
  },
};
```

데이터 수정

```
'use strict';
const {Emoji} = require('../index');

module.exports = {
  up: async (queryInterface, Sequelize) => {
    const t = await queryInterface.sequelize.transaction();
    try {
      const emojis = await Emoji.findAll({}, {transaction: t});

      for (const emoji of emojis) {
        const unicodeStr = emoji.unicode.codePointAt(0).toString(16);
        const imageUrl = `https://twemoji.maxcdn.com/v/latest/72x72/${unicodeStr.toLowerCase()}.png`;
        await emoji.update({
          'imageUrl': imageUrl,
        }, {
          transaction: t,
        });
      }

      await t.commit();
    } catch (error) {
      await t.rollback();
      throw error;
    }
  },

  down: async (queryInterface, Sequelize) => {
    const emojis = await Emoji.findAll({});
    const t = await queryInterface.sequelize.transaction();
    try {
      for (const emoji of emojis) {
        emoji.update({
          'imageUrl': '',
        }, {
          transaction: t,
        });
      }

      t.commit();
    } catch (error) {
      await t.rollback();
      throw error;
    }
  },
};
```

초기 데이터 추가

```
'use strict';
const {ReportReason} = require('../index');

module.exports = {
  up: async (queryInterface, Sequelize) => {
    const t = await queryInterface.sequelize.transaction();
    try {
      const reportReasons = [];
      const keys = [];

      keys.push('Spam');
      keys.push('Hateful words or metaphors');
      keys.push('Fraud or Lie');

      keys.forEach((key) => {
        reportReasons.push({
          key,
        });
      });

      await ReportReason.bulk_create(reportReasons, {
        transaction: t,
        returning: true,
      });

      await t.commit();
    } catch (error) {
      await t.rollback();
      throw error;
    }
  },

  down: async (queryInterface, Sequelize) => {
  },
};
```

sequelize-cli 으로 마이그레이션 적용하기

npx sequelize-cli db:migrate

sequelize-cli 설정

.sequelizerc 파일로 sequelize-cli Default 폴더 경로 수정

```
const path = require('path');

module.exports = {
  'config': path.resolve('src/configs/mysql_config.js'),
  'models-path': path.resolve('src/models'),
  'seeders-path': path.resolve('src/models/seeders'),
  'migrations-path': path.resolve('src/models/migrations')
}
```

**DB 마이그레이션은
실서버 운영에 필수!**

**유명한 웹 프레임워크마다
RDBMS 마이그레이션 기능/툴이 있음**

유명한 웹 프레임워크마다 RDBMS 마이그레이션 기능/툴이 있음

**Ruby on Rails, Django, Laravel, gin, Flyway, vapor ...
Node.js) Sequelize, TypeORM, Prisma**

**이번 매수업 프로젝트에
RDBMS 를 사용하신다면
마이그레이션 기능을
적극 활용해봅시다!**

감사합니다.