# 연습문제 13 풀이

## 파일구성

> `index.js`, `ErrorView.js`, `Spinner.js`, `Table.js`는 이전 예제들과 동일하기 때문에 제시하지 않습니다.

```
.
├── components
│   ├── ErrorView.js
│   ├── Spinner.js
│   └── Table.js
├── slices
│   └── TrafficAccidentSlice.js
├── App.js
├── index.js
└── store.js
```

## 소스코드

### slices/TrafficAccidentSlice.js

```
import { createSlice, createAsyncThunk } from '@reduxjs/toolkit'
import axios from 'axios';

const API_URL = 'http://localhost:3001/traffic_acc';

export const getTrafficAccident =
createAsyncThunk("TrafficAccidentSlice/getTrafficAccident", async
(payload, { rejectWithValue }) => {
    let result = null;

    try {
        const data = {};
        if (payload.year) {
            data.year = payload.year;
        }

        result = await axios.get(API_URL, {params: data});
    } catch (err) {
        result = rejectWithValue(err.response);
    }

    return result;
});

const TrafficAccidentSlice = createSlice({
    name: 'trafficAccident',
    initialState: {
```

```
        data: null,
        loading: false,
        error: null
    },
    reducers: {},
    extraReducers: {
        [getTrafficAccident.pending]: (state, { payload }) => {
            return { ...state, loading: true }
        },
        [getTrafficAccident.fulfilled]: (state, { payload }) => {
            return {
                data: payload?.data,
                loading: false,
                error: null
            }
        },
        [getTrafficAccident.rejected]: (state, { payload }) => {
            return {
                data: payload?.data,
                loading: false,
                error: {
                    code: payload?.status ? payload.status : 500,
                    message: payload?.statusText ? payload.statusText :
'Server Error'
                }
            }
        }
    },
});

export default TrafficAccidentSlice.reducer;
```

store.js

```
import { configureStore } from '@reduxjs/toolkit';

import TrafficAccidentSlice from './slices/TrafficAccidentSlice';

const store = configureStore({
    reducer: {
        trafficAccident: TrafficAccidentSlice
    },
    middleware: (getDefaultMiddleware) =>
getDefaultMiddleware({serializableCheck: false}),
    devTools: true
});

export default store;
```

App.js

```
import React, { memo } from 'react';
import styled from 'styled-components';
import { useSelector, useDispatch } from 'react-redux';
import { getTrafficAccident } from './slices/TrafficAccidentSlice';

import Spinner from './components/Spinner';
import ErrorView from './components/ErrorView';
import Table from './components/Table';

const SelectContainer = styled.div`
    position: sticky;
    top: 0;
    background-color: #fff;
    border-top: 1px solid #eee;
    border-bottom: 1px solid #eee;
    padding: 10px 0;
    margin: 0;

    select {
        margin-right: 15px;
        font-size: 16px;
        padding: 5px 10px;
    }
`;

const App = memo(() => {
    const dispatch = useDispatch();
    const { data, loading, error } = useSelector((state) =>
state.trafficAccident);
    const [year, setYear] = React.useState('');

    const onSelectChange = React.useCallback(e => {
        e.preventDefault();
        const current = e.target;
        const value = current[current.selectedIndex].value;
        setYear(value);
    }, []);

    React.useEffect(() => {
        dispatch(getTrafficAccident({ year: year }));
    }, [dispatch, year]);

    return (
        <div>
            <h1>Exam 13</h1>

            {/* 로딩바 */}
            <Spinner visible={loading} />

            {/* 검색 조건 드롭다운 박스 */}
            <SelectContainer>
                <select name='year' onChange={onSelectChange} value=
```

```
{year}>
                    <option value="">-- 년도 선택 --</option>
                    {[...new Array(2018 - 2005 + 1)].map((v, i) => {
                        return (<option key={i} value={2005 + i}>{2005 +
i}년도</option>)
                    })}
                </select>
            </SelectContainer>

            {error ? <ErrorView error={error} /> : (
                data && (
                    <Table>
                        <thead>
                            <tr>
                                <th>번호</th>
                                <th>년도</th>
                                <th>월</th>
                                <th>교통사고 건수</th>
                                <th>사망자 수</th>
                                <th>부상자 수</th>
                            </tr>
                        </thead>
                        <tbody>
                            {data.map(({ id, year, month, accident, death,
injury }, i) => {
                                return (
                                    <tr key={i}>
                                        <td>{id}</td>
                                        <td>{year}년</td>
                                        <td>{month}월</td>
                                        <td>{accident.toLocaleString()}건
</td>
                                        <td>{death.toLocaleString()}명</td>
                                        <td>{injury.toLocaleString()}명
</td>
                                    </tr>
                                );
                            })}
                        </tbody>
                        <tfoot>
                            <tr>
                                <th colSpan="3">합계</th>
                                <th>{data.map((v, i) =>
v.accident).reduce((p, c) => p + c, 0).toLocaleString()}건</th>
                                <th>{data.map((v, i) =>
v.death).reduce((p, c) => p + c, 0).toLocaleString()}명</th>
                                <th>{data.map((v, i) =>
v.injury).reduce((p, c) => p + c, 0).toLocaleString()}명</th>
                            </tr>
                        </tfoot>
                    </Table>
                )
            )}
        </div>
```

```
    );
});

export default App;
```