

2025年度

ソフトウェアシステム開発

第11,12回Webプラットフォーム1（開発環境設定編）

開発環境と本番環境

Web系システムの開発では開発者の端末にWebサーバーやデータベースなどの環境を組み込み、その端末でバックエンド及びフロントエンドシステムを開発を行うことが多くあります。

今回は開発環境として、自己のWindows PCに Visual Studio Code、Node.js、それに関連するシステムをインストールします。

また、本番環境として、Linux上にNode.js及び関連するシステムをインストールします。

(サーバ・ネットワーク履修者のLinuxを流用しますが、非履修者が設定したい場合は別途資料を参照)

開発演習では、各々の端末でバックエンドおよびフロントエンドのシステムを開発します。

実際のシステム移行では、Linuxサーバーにアプリを配置、Windows PCよりアクセスし、動くことを確かめます。

開発環境の設定

自分のPCを開発機としての環境を設定します。

① Visual Studio Code のインストール

<https://code.visualstudio.com/> にアクセス

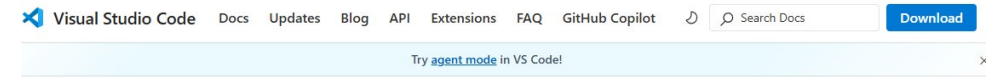
「Download for Windows」をクリック

VS Code 起動したら「Rich support for all your languages」にチェックを入れる。

② Node.js をインストール

<https://nodejs.org/ja> にアクセス

「Node.js (LTS) をダウンロードする」をクリック

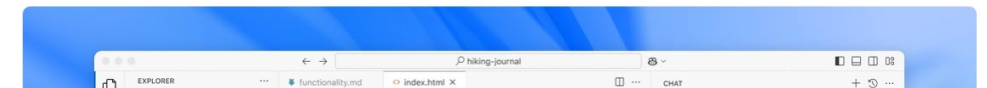


**Your code editor.
Redefined with AI.**

Download for Windows

Try agent mode

[Web](#), [Insiders edition](#), or [other platforms](#)



node

学ぶ

はじめに

ダウンロード

ブログ

ドキュメント

貢献

認定資格

Q 入力を開始...

Ctrl + K

Next-10
Survey

We need your feedback to help us shape
Node.js

どこでもJavaScript を使おう

Node.js®はクロスプラットフォームに対応した
フリーでオープンソースのJavaScript実行環境で
す。開発者にサーバー、ウェブアプリ、コマンド
ラインツール、スクリプトなどを開発する環境を
提供します。

Node.js (LTS) をダウンロードする

Create an HTTP Server Write Tests Read and Hash a File

```
1 // server.mjs
2 import { createServer } from 'node:http';
3
4 const server = createServer((req, res) => {
5   res.writeHead(200, { 'Content-Type': 'text',
6     res.end('Hello World!\n');
7 });
8
9 // starts a simple http server locally on port 3000
10 server.listen(3000, '127.0.0.1', () => {
11   console.log('Listening on 127.0.0.1:3000');
12 });
13
14 // run with 'node server.mjs'
```

JavaScript

クリップボードにコピー

Node.js 及び VS Code の確認

スタート→node.js command prompt を選択

mkdir Inshokuten → Inshokuten フォルダを作ります。

cd Inshokuten → Inshokuten フォルダの中に移動します。

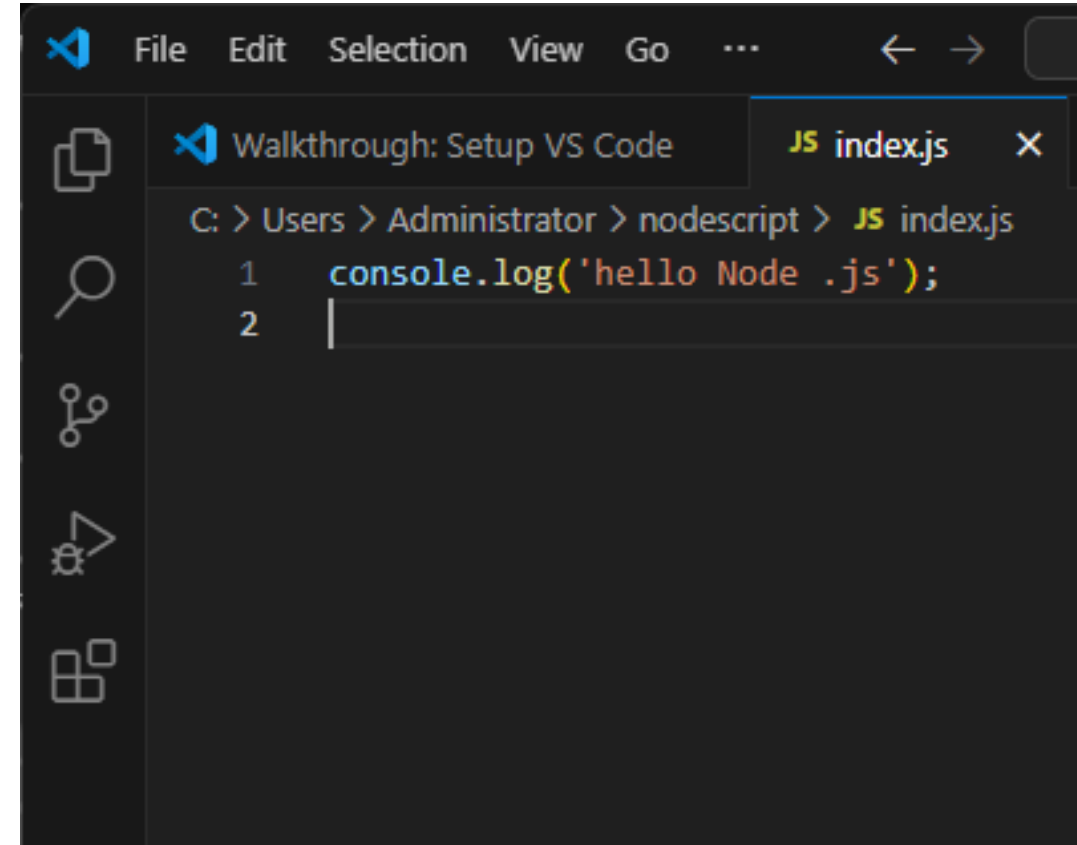
続いて、VS Code を起動。

File→New File... を選択。

名前を index.js とする。下記を入力後、

「Inshokuten」フォルダの中に保存。

Console.log('hello world');

A screenshot of the Visual Studio Code (VS Code) interface. The top menu bar shows 'File', 'Edit', 'Selection', 'View', 'Go', and a search icon. Below the menu, there are two tabs: 'Walkthrough: Setup VS Code' and 'JS index.js'. The 'JS index.js' tab is active, showing a code editor with the following content:

```
C: > Users > Administrator > nodescript > JS index.js
1 console.log('hello Node .js');
2 |
```

The left sidebar contains icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The background is dark, and the text is light-colored.

Node.js のテスト起動

① Node.js の起動

コマンドプロンプト内で下記を実行

```
node -v
```

v22.15.0などとバージョンが出てきたらOK.

② サンプルファイルの実行

```
node index.js
```

SQLite のインストール

<https://www.sqlite.org/index.html>

上記にアクセスする。

「DownLoad」をクリック

→「Precompiled Binaries for Windows」
の下にある、下記をダウンロードする。

「sqlite-dll-win-x64～」

「sqlite-tools-win-x64～」

Precompiled Binaries for Windows

[sqlite-dll-win-x86-
3490200.zip](#)
(1.02 MiB)

32-bit DLL (x86) for SQLite
(SHA3-256: 20048a06f76139e97)

[sqlite-dll-win-x64-
3490200.zip](#)
(1.28 MiB)

64-bit DLL (x64) for SQLite
(SHA3-256: 3a0f4933fd78a2491a)

[sqlite-tools-win-x64-
3490200.zip](#)
(6.13 MiB)

A bundle of command-line tools
(4) [sqlite3 rsync.exe](#). 64-bit
(SHA3-256: 8dc8764828f89c25ea)

SQLite3 のインストール

スタート右クリック→エクスプローラ→C:¥に移動。

何もないところを右クリック→新規作成→「フォルダ」
名前を「sqlite3」とする。

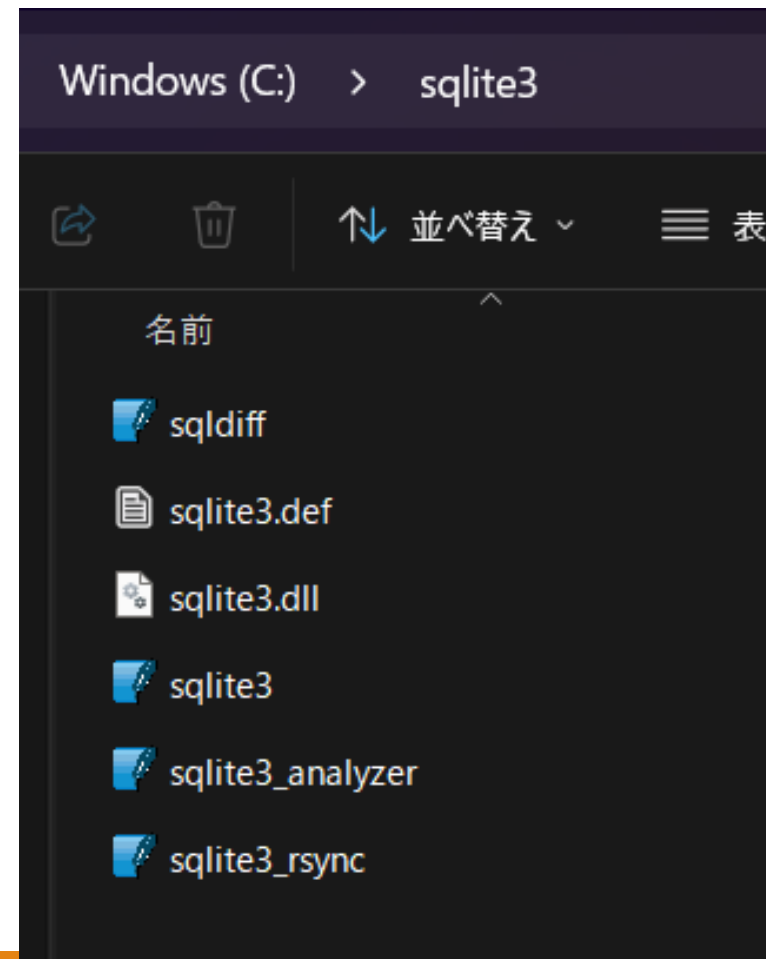
次に2つのZipファイルをダウンロードしたフォルダに移動。

Zipファイル1つずつ選択(クリック)→右クリック→展開

上記を両方と行う。

展開された二つのフォルダにあるすべてのファイルを、
C:¥sqlite3 の下にコピーする。

すると C:¥sqlite3 は右の通りになる。



SQLite3のインストール

スタートの隣の「検索」にて cmd と打つ→「コマンドプロンプト」を選択下記を実行する。

①(環境変数PATHのバックアップ) `echo %PATH% > C:¥sqlite3¥path_backup.txt`

②(バックアップファイルの確認) `more C:¥sqlite3¥path_backup.txt`

↑内容がバックアップされていることを確認してから③を実行する。

③(下記コマンドを打つときは決して間違えない!すべて半角、セミコロン、コロン、ダブルクォーテーションに注意! 自己責任)

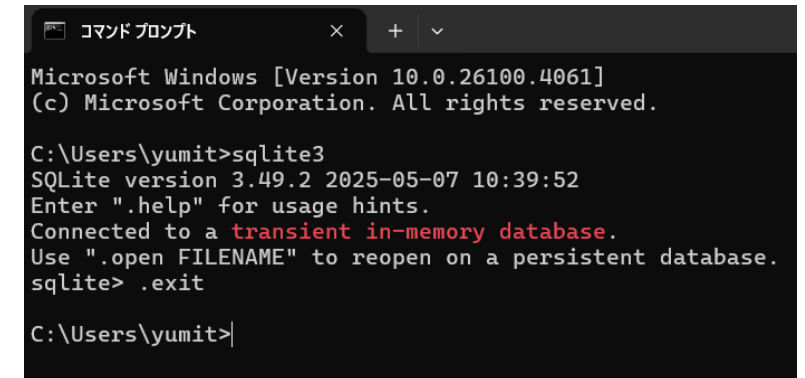
`setx PATH "%PATH%;C:¥sqlite3"`

④コマンドプロンプトをいったん閉じて、もう一度起動する。

⑤コマンドプロンプト上で下記のコマンドを実行

`sqlite3`

⑥ SQLite3 のプロンプトが出てきたら、「.exit」と入力しエンターキーを押して終了する。



```
コマンド プロンプト
Microsoft Windows [Version 10.0.26100.4061]
(c) Microsoft Corporation. All rights reserved.

C:\Users\yumit>sqlite3
SQLite version 3.49.2 2025-05-07 10:39:52
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .exit

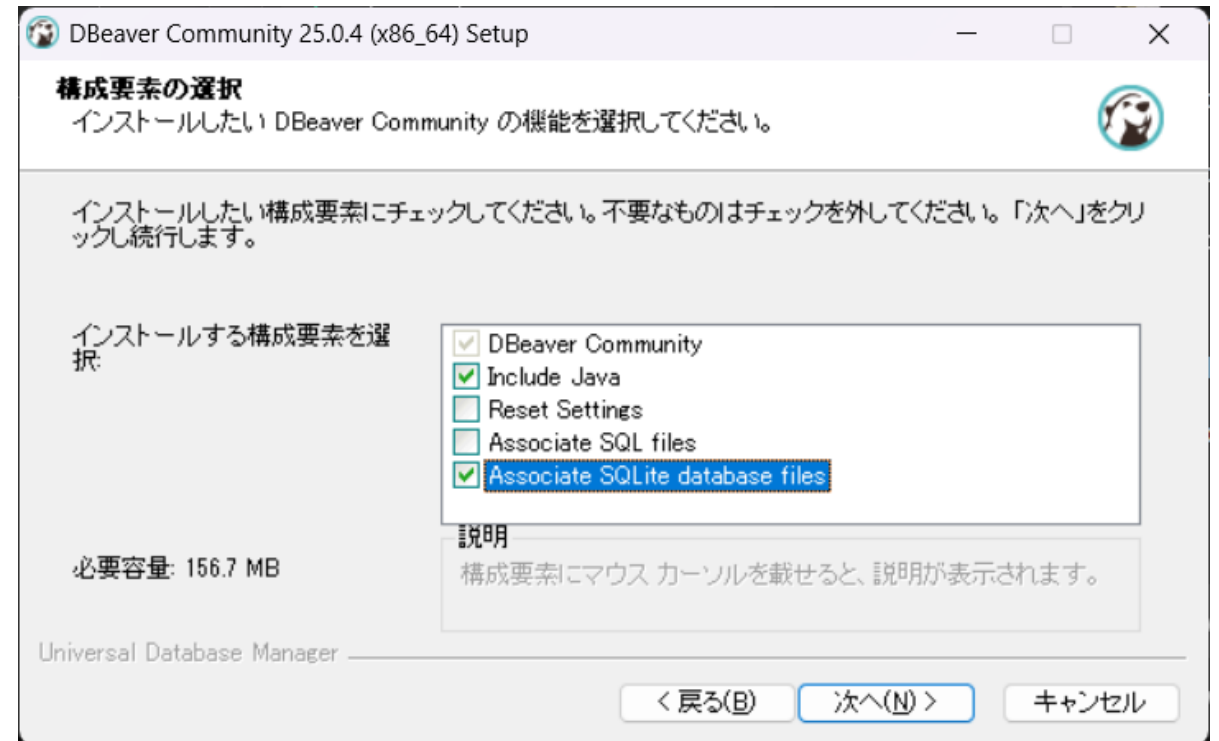
C:\Users\yumit>
```


DBeaver のインストール

ソフトウェアシステム開発の授業では SQLite3 を実習で使用しますが、DBファイルを簡単に作るためのツールとしてDbeaverがあると便利です。

<https://dbeaver.io/>

上記にアクセスし、Download をクリック。
→ Windows (installer) をクリックします。
ダウンロードしたインストーラを起動すると、
右の画面が出てきますので、
Associate SQLite database files にチェックを
入れてください。

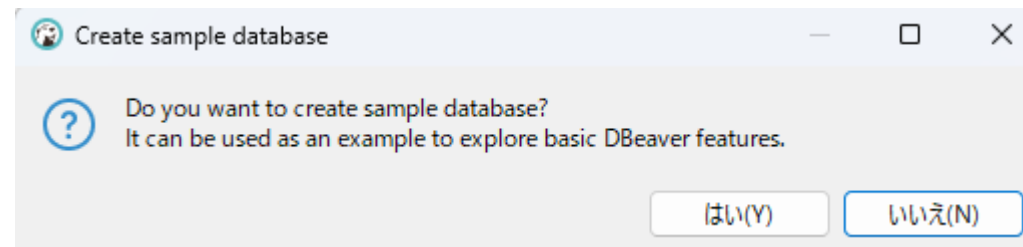
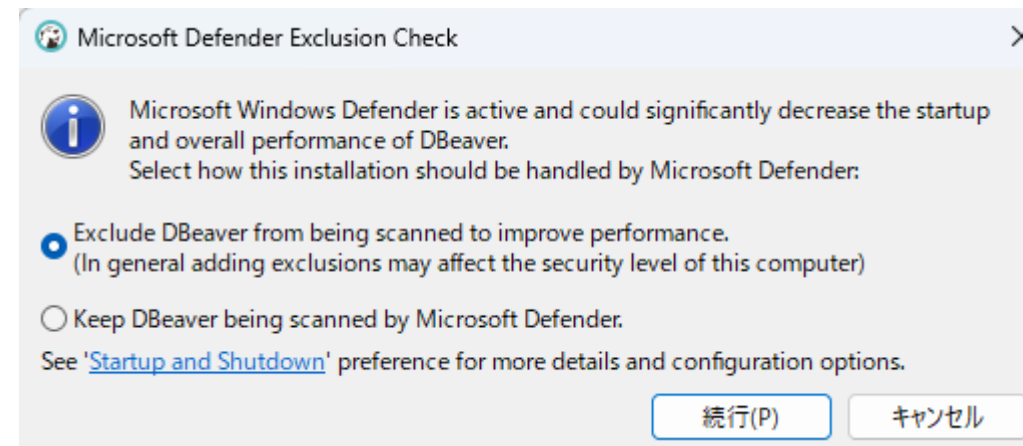


DBeaver設定

右のような画面が現れた場合、パフォーマンスを重視すると上を選択することになり、下を選ぶとセキュリティを重視することになります。

Create sample database

にて、「はい」を選択してください。



SQLite ドライバを設定する

SQLite ドライバをインストール場面でダウンロードして下さい。(基本的にこれで大丈夫です)

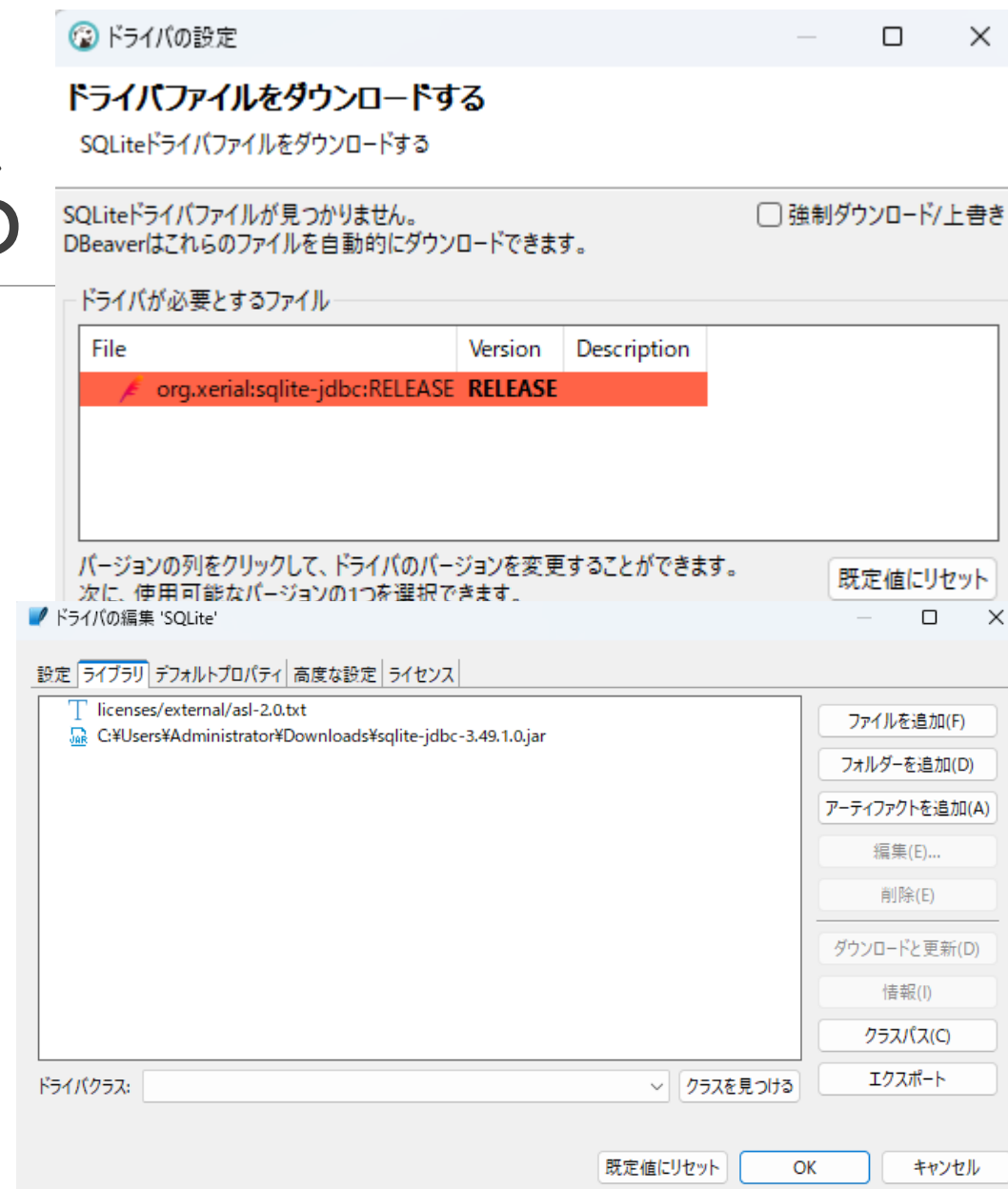
時折、右上のようにうまくいかない場合があります。
その際は、キャンセルした後、下記を実施して下さい。

① 下記をダウンロードして下さい。

<https://repo1.maven.org/maven2/org/xerial/sqlite-jdbc/3.49.1.0/sqlite-jdbc-3.49.1.0.jar>

② DBeaver を起動し、「データベース」→「ドライバマネージャー」→「SQLite」を選択→「編集」→「ライブラリ」

③ 赤い行を選択して「削除」。そのあと「ファイルを追加」をして、ダウンロードしたフォルダにて、sqlite-jdbc-3.49.1.0.jar を選択する。



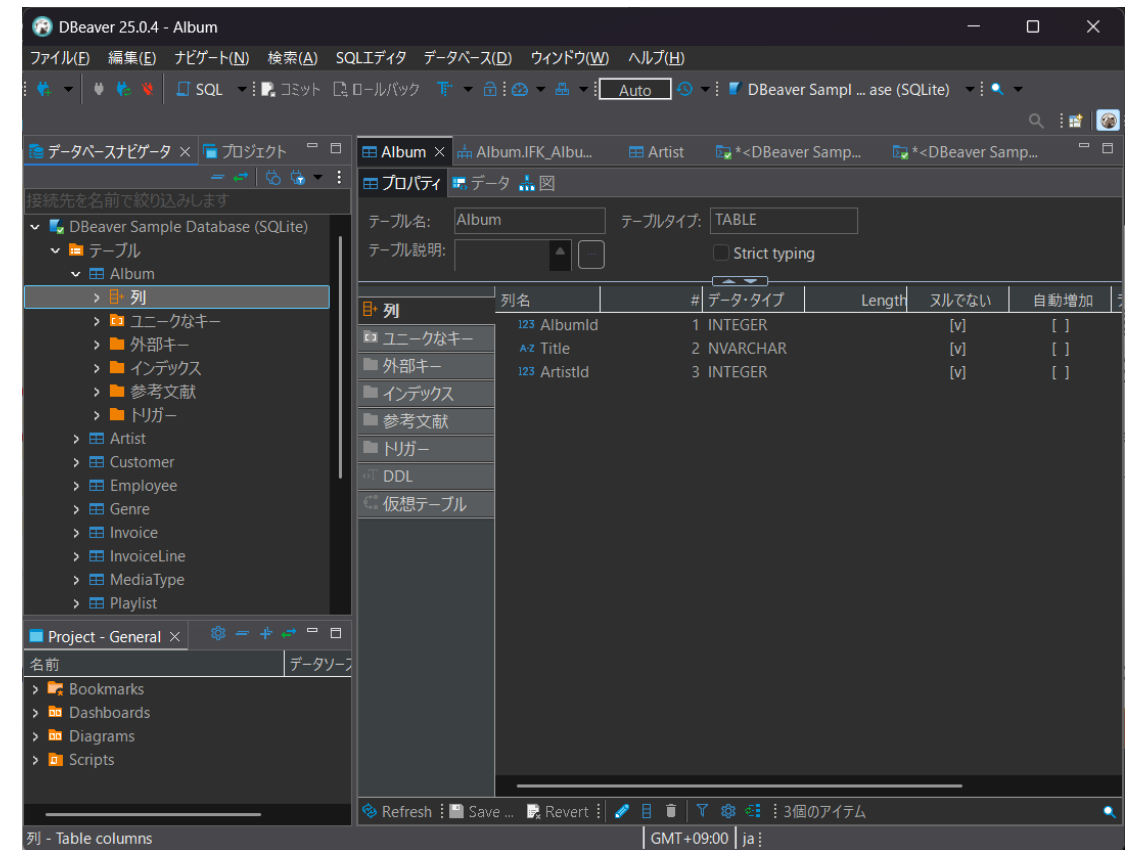
DBeaver のサンプルDB

DBeaver を起動しサンプルを表示するようにすると、
サンプルのDBが出てくる。

「DBeaver Sample Database (SQLite) 」の下を開き

→「テーブル」→Album→列を選択すると右の図のよ
うになる。

テーブル「Album」のスキーマが表示される



テーブルのデータの表示

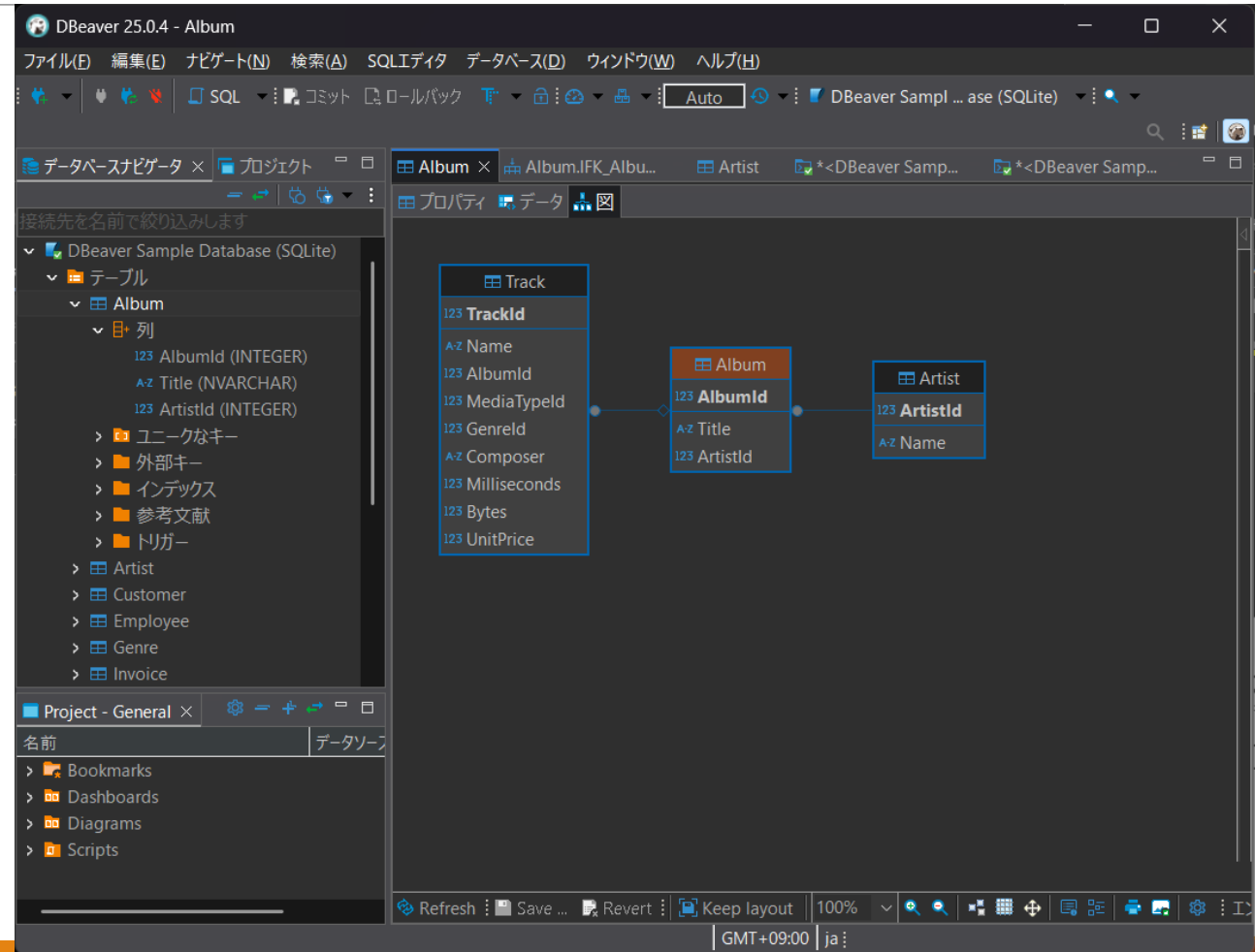
Albumを右クリック→「View Data」とすると
Albumテーブルに登録されているレコードが
表示される。Select * from Album;と同じ。

The screenshot shows the DBeaver 25.0.4 interface. On the left, the 'データベースナビゲータ' (Database Navigator) pane shows the 'Album' table selected under 'DBeaver Sample Database (SQLite)'. The 'プロパティ' (Properties) pane on the right shows the table's structure: 'AlbumId (INTEGER)', 'Title (NVARCHAR)', and 'ArtistId (INTEGER)'. The main window displays the 'Album' table data in a grid view. The status bar at the bottom indicates '347 row(s) fetched - 0.002s (0.001s fetch), on 2025-05-18 at 21:56:06'.

AlbumId	Title	ArtistId
1	For Those About	1
2	Balls to the Wall	2
3	Restless and Wild	2
4	Let There Be Rock	1
5	Big Ones	3
6	Jagged Little Pill	4
7	Facelift	5
8	Warner 25 Anos	6
9	Plays Metallica By	7
10	Audioslave	8
11	Out Of Exile	8
12	BackBeat Soundtr	9
13	The Best Of Billy	10
14	Alcohol Fueled B	11
15	Alcohol Fueled B	11
16	Black Sabbath	12
17	Black Sabbath Vc	12
18	Body Count	13
19	Chemical Weddir	14
20	The Best Of Budd	15

テーブルのリレーションの表示

Albumを右クリック→「Data Diagram」とすると
Albumテーブルにある外部参照キーに関する
テーブルが視覚的に表示される。(ER図)



DBファイルを作成

SQLite のDBファイルを作成します。

DBeaverで「ファイル」→「新規」→ウィザードの選択で
「DBeaver」→「データベース接続」→「次へ」

「SQLite」を選択し「次へ」

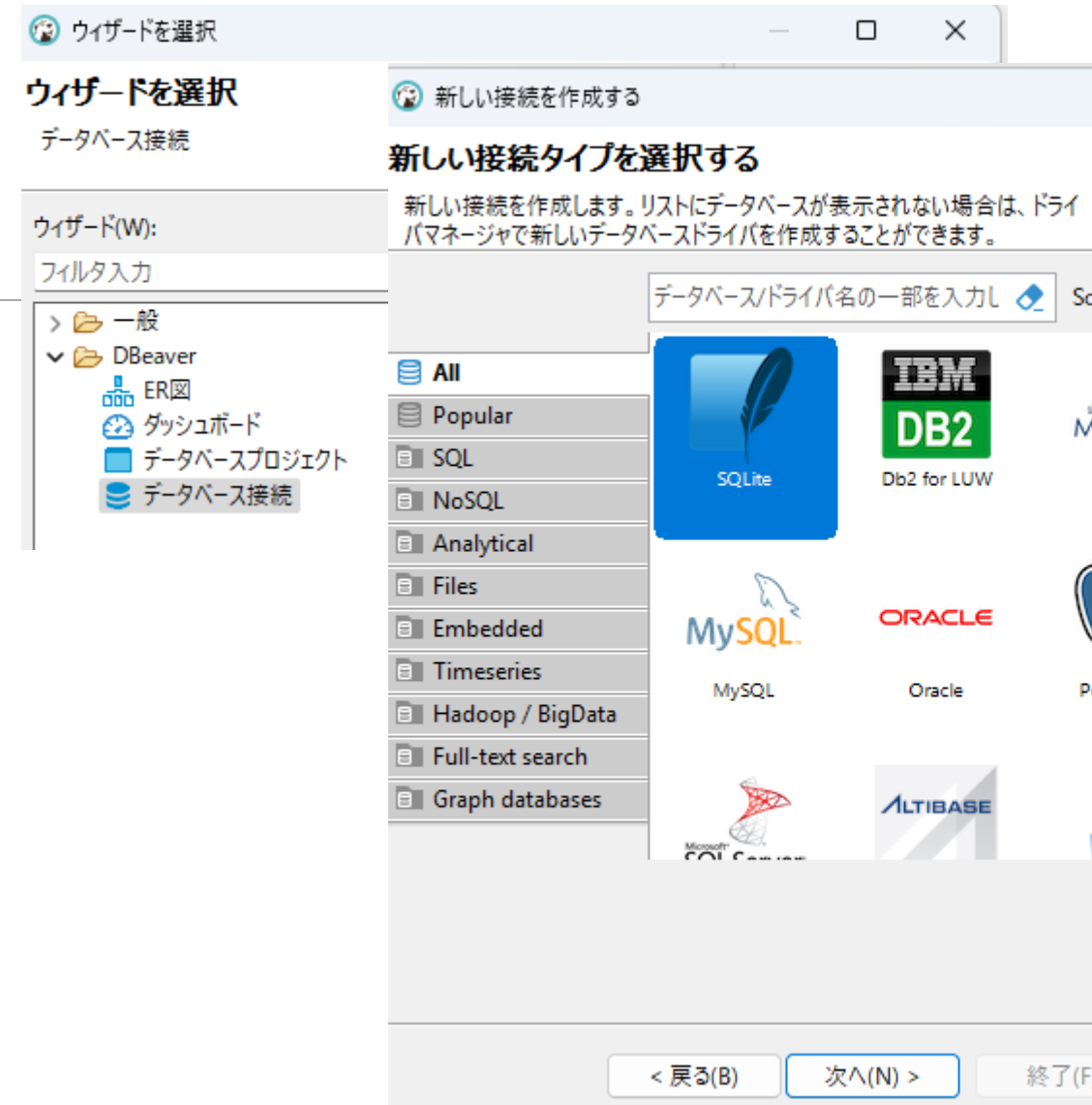
「JDBC接続設定」でパスの横のcreateをクリック。

→C:¥に Inshokuten というフォルダを作成

→その下にbackendというフォルダを作成

→Inshokuten.sqlite3 という名前で「保存」

→「終了」をクリック



テーブルの作り方

①「テーブル」を右クリック→「新しく作る表」→ テーブル名をTestTableと入力。

②「列」を選択し、右の何も無いところを右クリック→「新しく作る カラム」を選択

名前 ID データタイプ INTEGER

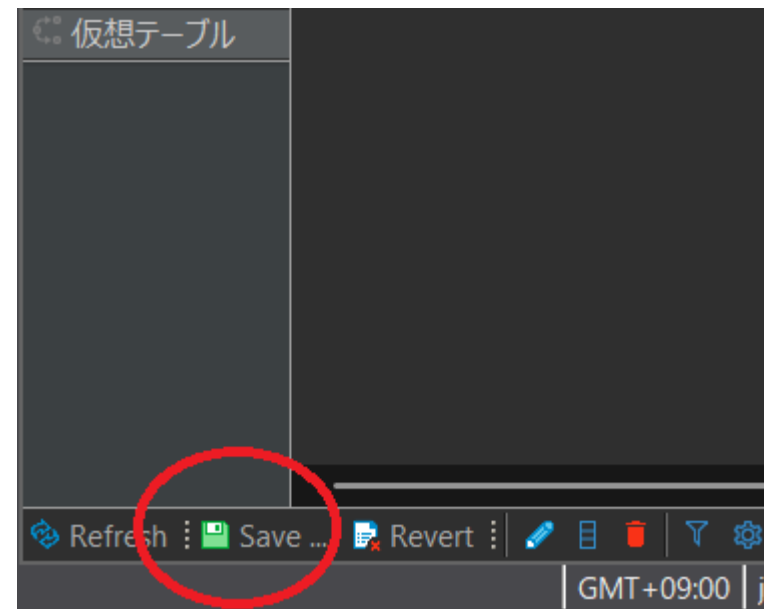
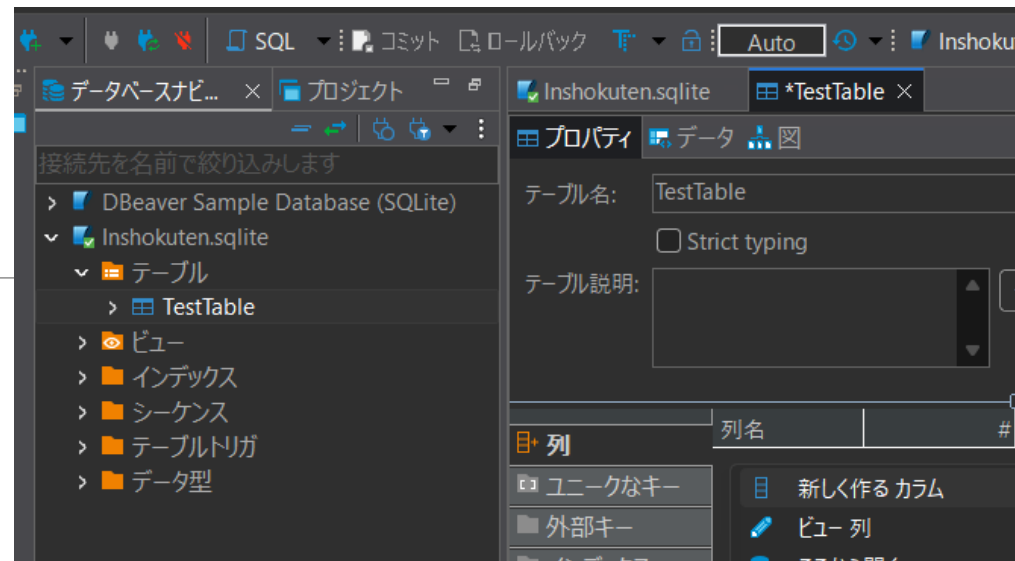
ヌルでないにチェック、自動増加にチェック OK

③「新しく作る カラム」を選択→ 名前 Name データタイプ TEXT とし、OK

④「新しく作る カラム」を選択→ 名前 Price データタイプ Integer とし、OK

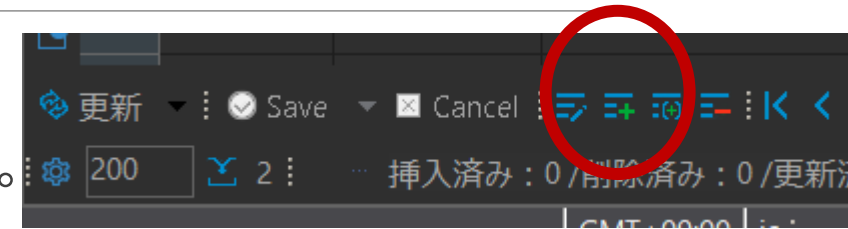
⑤下部の「Save」ボタンを押すと、「SQLをプレビュー」という画面が出てくるので「実行」を押す。

これで CREATE TABLE ができました。



TestTableのデータを挿入

- ① TestTableを右クリック→View Data下部の+をクリック。
 - ② ID に 1, Name に「あいうえお」を入力して、下部のSaveをクリック。
 - ③ 再び+をクリックし、Name に「かきくけこ」を入力して、下部のSaveをクリック。
- これで2件保存されました。



SQL文による入力

TestTableを右クリック→「SQL の生成」→INSERTを選択する。

SQLのプレビューが出てくるので、「コピー」をクリックする。

そのあと、一番上の「SQL エディタ」をクリック。

「新しいSQLエディタ」を選択する。

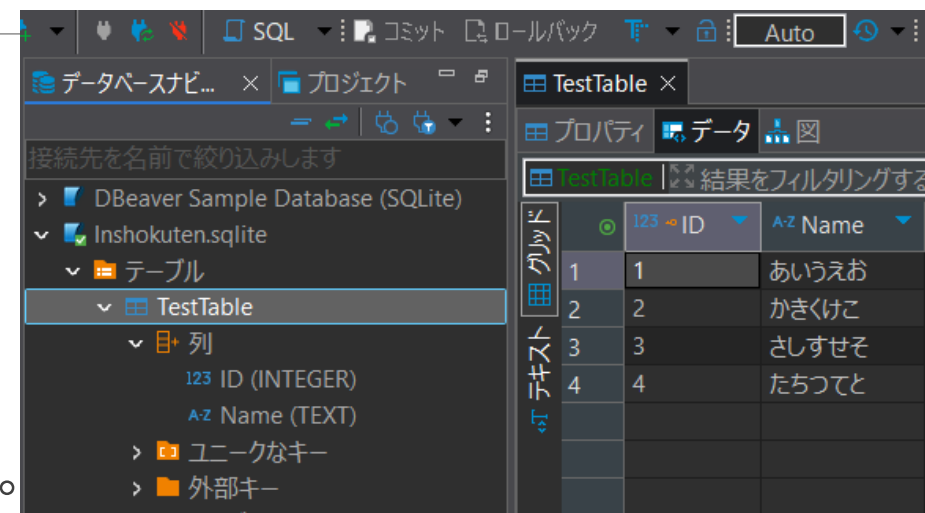
コピーしていたものをペーストした後、下記の通り編集する。

```
INSERT INTO TestTable (ID, Name) VALUES(3, 'さしすせそ'), (4,'たちつてと');
```

「SQL エディタ」→「SQL文を実行する」を選択。

右「TestTable」のウィンドウを消した後、左のペインの「TestTable」を右クリックし「View Data」とすると、データが4行になっていることがわかる。

その他、Update コマンドまたは DBeaver で Price にそれぞれ数値を入れて下さい。



React

コマンドプロンプトで C:\Inshokuten に移動

cd C:\Inshokuten

~~npm config set proxy http://username:password@proxy01.osaka.hal.ac.jp:8080~~

~~npm config set https-proxy http://username:password@proxy01.osaka.hal.ac.jp:8080~~

Reactのファイルを生成

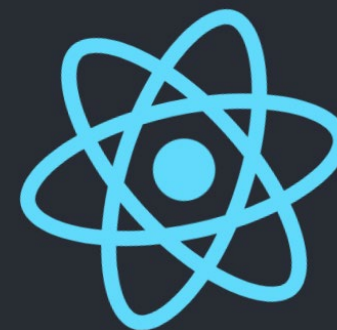
npx create-react-app frontend

時間がかかりますが、必要なファイルが生成されます。

cd frontend

React 起動

npm start →ブラウザで右の画面が確認できます。コマンドプロンプトで C-c とすると終了することができます。



Edit src/App.js and save to reload.

[Learn React](#)

Express の設定

コマンドプロンプトで下記を実行

```
cd C:¥Inshokuten¥backend
```

```
npm init
```

上記を実行するとpackage name 等
聞かれるので全部 Enterキーを押す。

```
npm install express --save
```

Express のテスト

VS Code で新規作成し、下記のコードをInshokuten¥backend のフォルダ内に server.js という名前で保存。

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

const app = express();

const PORT = 3000;

// テスト用ルート

app.get('/', (req, res) => {

  res.send('Hello from Express!');

});

app.listen(PORT, () => {

  console.log(`Server is running at http://localhost:${PORT}`);

});
```

テスト起動

コマンドプロンプトを起動し、下記を実行。

```
cd C:\Inshokuten\backend
```

```
node server.js
```

そのあと、ブラウザで localhost:3000 にアクセス
すると、Hello from Express! というものが表示される。

その他の設定

sqlite3 Node.js でSQLiteを取り扱うことができるように、モジュールをインストールします。

下記を実行する。

```
npm install sqlite3
```

cors Reactから別のポート(にする予定の)Expressに通信を許可するため、下記を実行する。

```
npm install cors
```

multer 画像などのファイルのアップロードを扱うミドルウェア

```
npm install multer
```

サンプルファイルの展開

※大事なファイルを消さないように十分気を付けて下記の通り実施して下さい。

- ① 「2025 Webプラットフォーム関係 Sample.zip」を解凍する。
- ② C:¥Inshokuten¥frontend¥src の下のApp.js を App_backup.jsとする。
- ③ C:¥Inshokuten¥frontend¥src の下に、解凍したフォルダ内の frontend¥src¥App.jsをコピーする。
- ④ C:¥Inshokuten¥backend¥ の下に、解凍したフォルダ内の backend¥server2.jsをコピーする。
- ⑤ C:¥Inshokuten¥backend¥ の下に、public というフォルダを作る。
- ⑥ C:¥Inshokuten¥backend¥public¥ の下に、解凍したフォルダ内のbackend¥public¥images のフォルダごとコピーする。

サンプルファイルの実行

スタートの隣の検索からコマンドプロンプトを2つ起動する。

1つは`cd C:\¥Inshokuten¥frontend`

もう1つは `cd C:\¥Inshokuten¥backend`

とする。

backend のコマンドプロンプトで `node server2.js` とする。

frontendのコマンドプロンプトで `npm start` とする。