プログラムからデータベースに接続する

MySQLデータベースは、様々なプログラムから接続できます。その接続を可能とするのが「ドライバ」や「ライブラリ」と呼ばれるプログラムです。今回はプログラムからMySQLに接続する方法を学習していきます。

# プログラムからMySQLに接続する仕組み

RDBMSとしてのMySQL(mysqld)はあくまで「データベースを操作するもの」であり、文字入力の機能や画面への表示機能は持っていません。送られてきた命令文(コマンド)を実行して結果を返すという処理を行っています。授業では、コマンドプロンプトで使用できる

mysqlコマンドを使って、データベースに命令文を送り、結果を表示しています。

MySQLサーバ

(mysqld)

MySQLクライアント

(mysql)

コマンド

結果

MySQLサーバにコマンドを送れるのはMySQLクライアントだけではありません。

**「データのやりとりの方法」が合っているなら、どのようなプログラムからでも接続することが出来ます。**この「データのやり取りの方法」を合わせたり、実際に通信したりするのが

「ドライバ」や「ライブラリ」と呼ばれるものです。

MySQLサーバ

(mysqld)

プログラム

ドライバ

ライブラリ

コマンド

結果

**各種プログラミング言語と「ドライバ」や「ライブラリ」を組み合わせると、自作のブログラムからMySQLに接続できるようになります。**

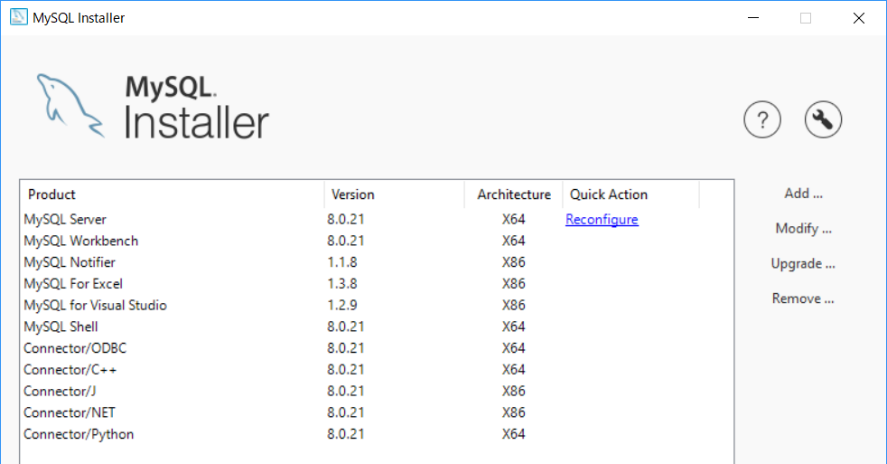
# MySQLコネクタ

そのようなドライバ・ライブラリが必要なのかは、利用するプログラミング言語によって異なります。共通のものもありますが、プログラミング言語ごとに専用の物が用意されていることが多いです。プログラミング言語側で準備しないといけない場合とMySQL側で準備しなといけない場合がありますが、**MySQL側で準備する「ドライバ」や「ライブラリ」を**

**MySQLコネクタ**と呼びます。

ハンズオン　MySQLコネクタを確認する

１．Windowsのメニューから、「MySQL Installer」をクリックしてインストーラーを起動



この部分がコネクタ

※画像はFullで入っている

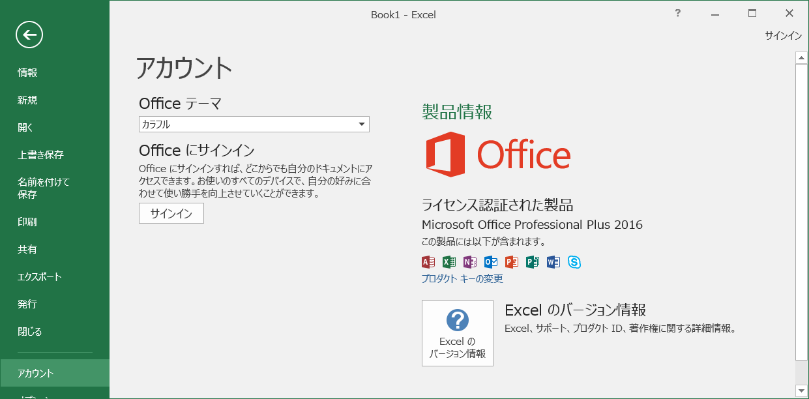
|  |  |
| --- | --- |
| コネクタ種類 | 説明 |
| Connector/ODBC | Open Database Connectivity  Windowsアプリケーションから  MySQLに接続するためのコネクタ |
| Connector/C++ | C++でMySQLに接続するためのコネクタ |
| Connector/J | JavaでMySQLに接続するためのコネクタ |
| Connector/NET | .NETアプリケーション(C#やVB)から  MySQLに接続するためのコネクタ |
| Connector/Python | PythonでMySQLに接続するためのコネクタ |

# EXCELからのデータベース接続

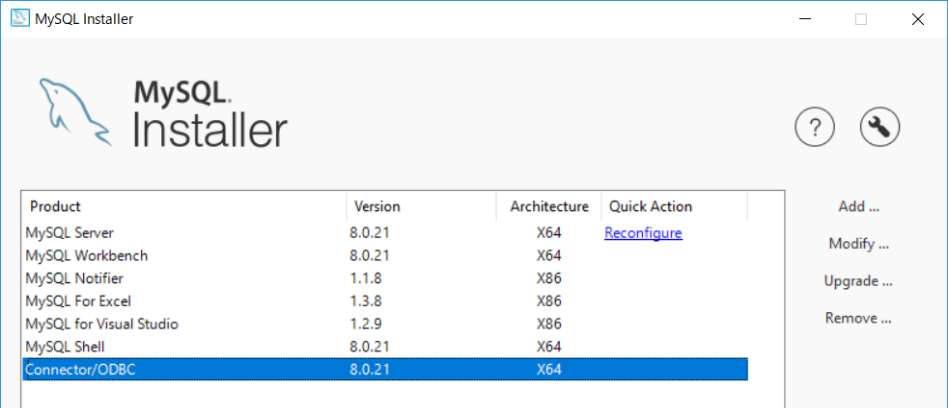
Windows環境でExcelやAccessからMySQLサーバに接続する場合は、Windowにおいて汎用的なデータベース接続をするODBCという仕組みを使います。ODBCでMySQLに接続するためには「Connector/ODBC」MySQLコネクタが必要です。

ハンズオン　Excelからデータベースに接続する

１．Excelを新規ファイルで開いて、Excelが何bit版か確認する。



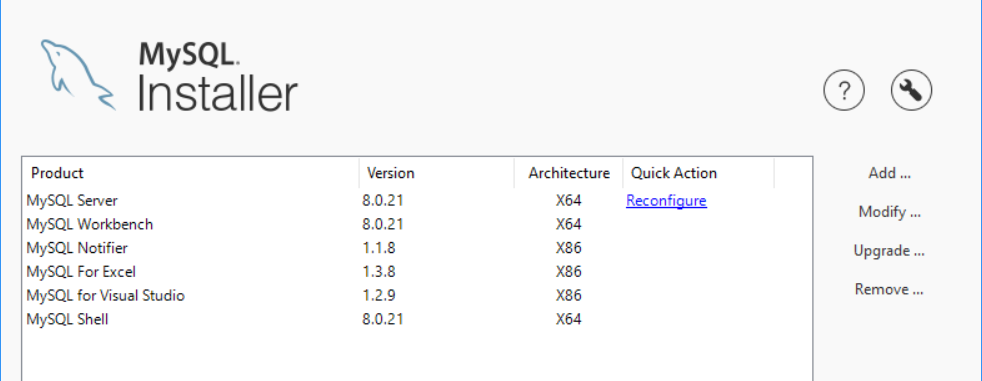
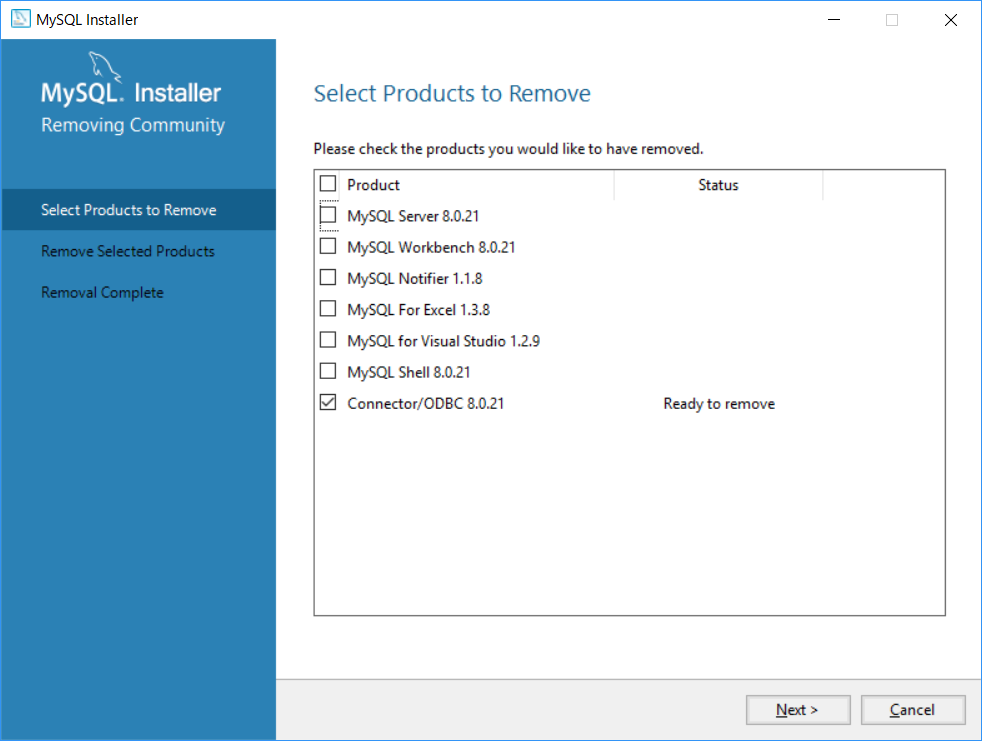
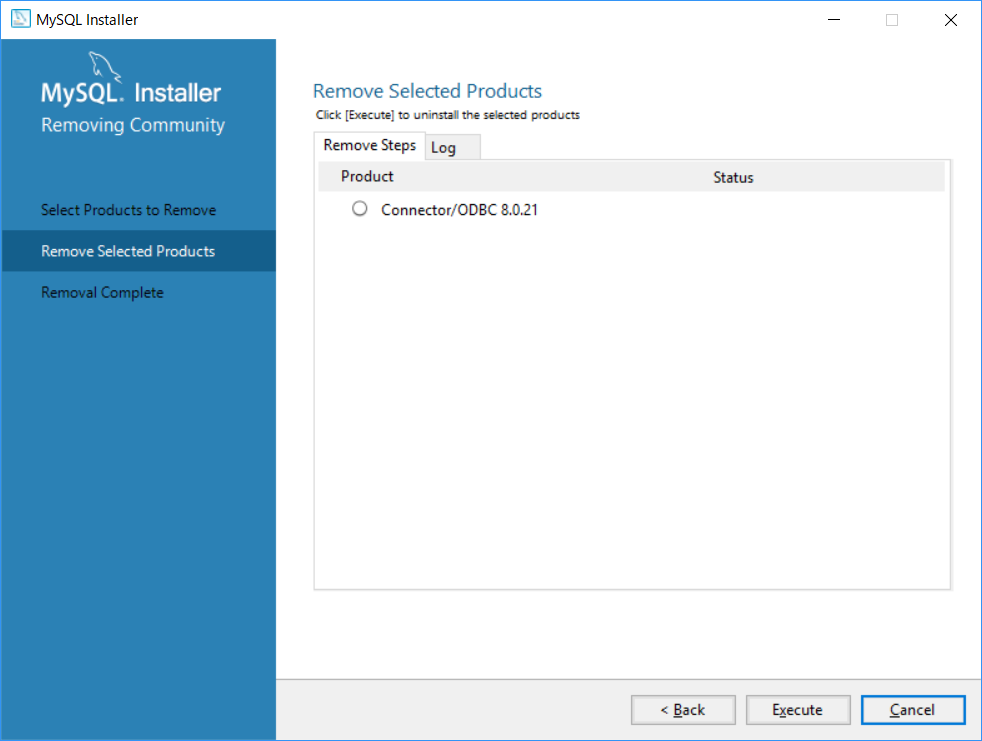
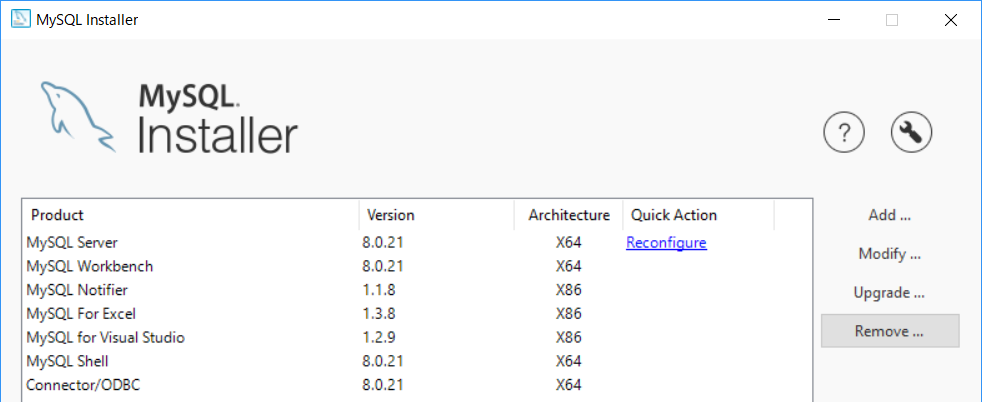
２．MySQL InstallerからODBCコネクタを確認する



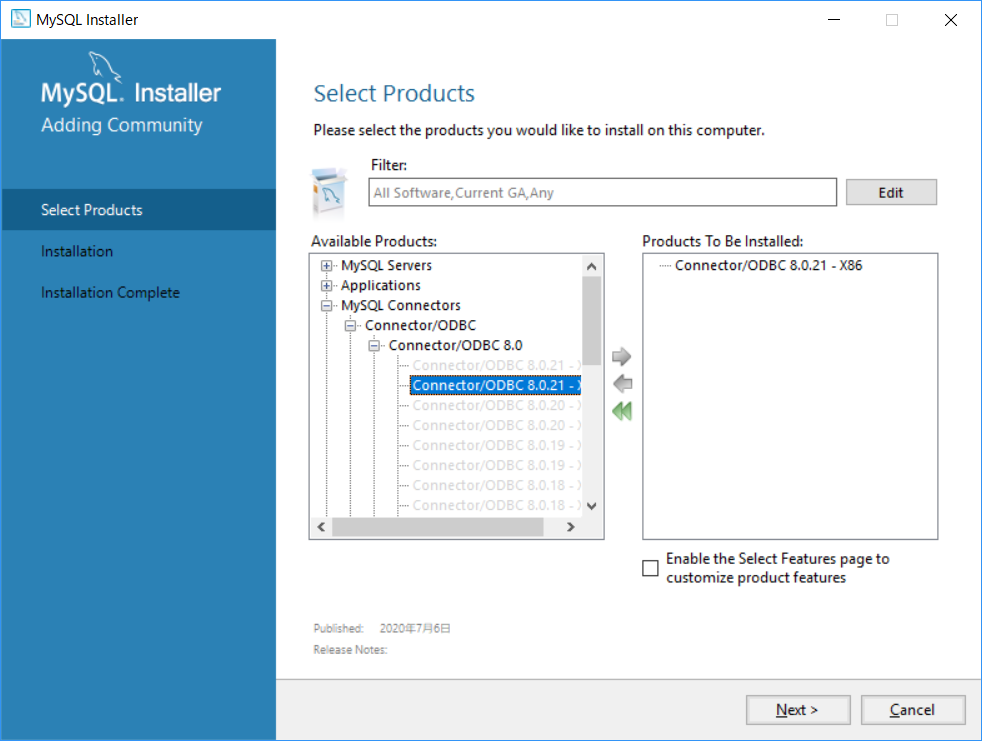
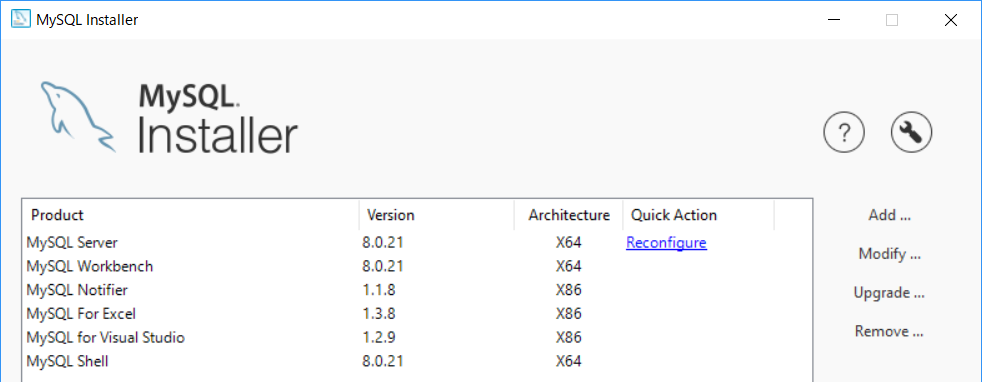
ExcelとMySQLのコネクタの

ビット版が一致している場合は、３、４作業不要

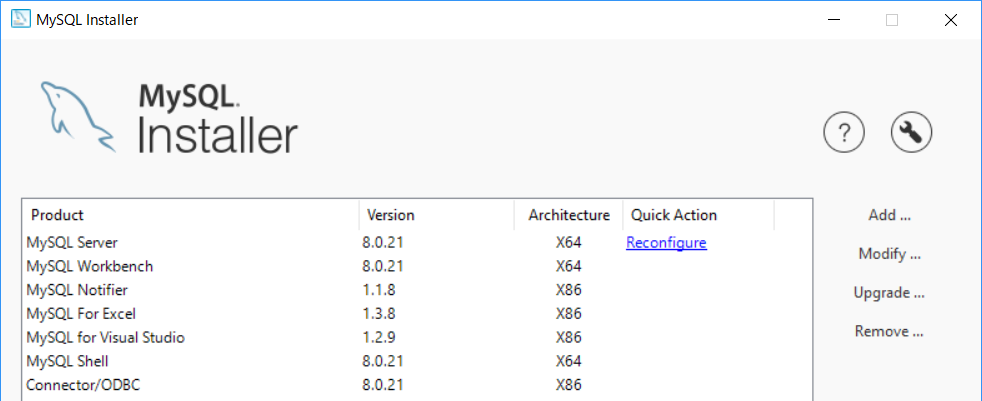
３．「Remove」をクリックしてODBCコネクタを一度削除する。



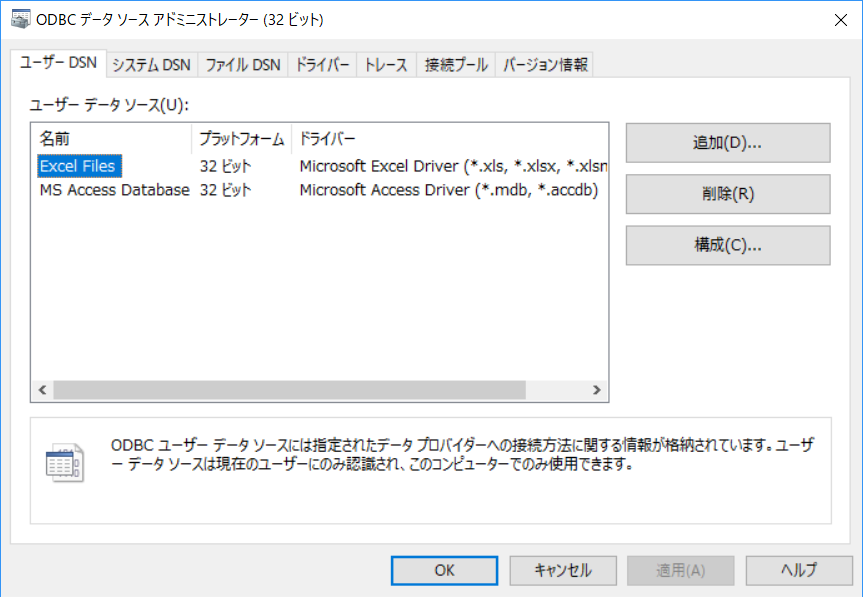
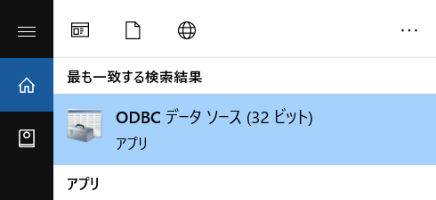
４．「Add」をクリックしてODBCコネクタをインストールする。



X86(32ビット)版コネクタを選択

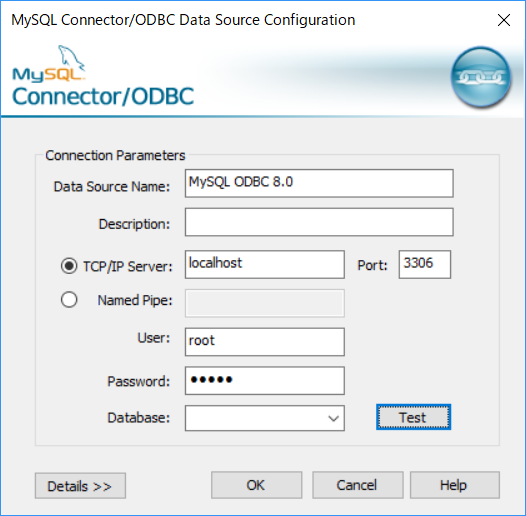
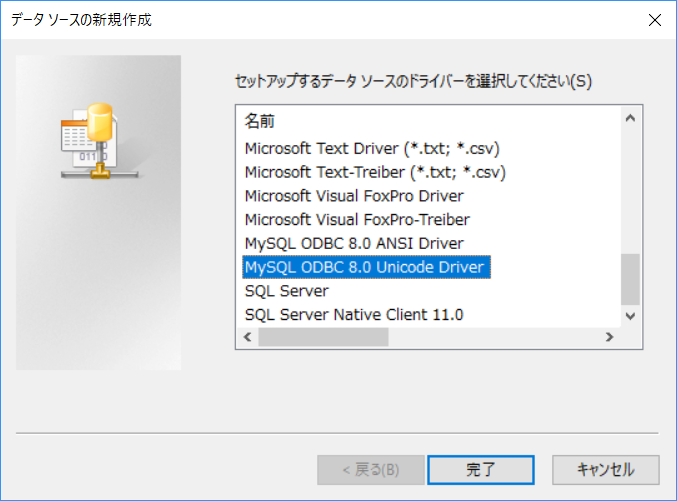
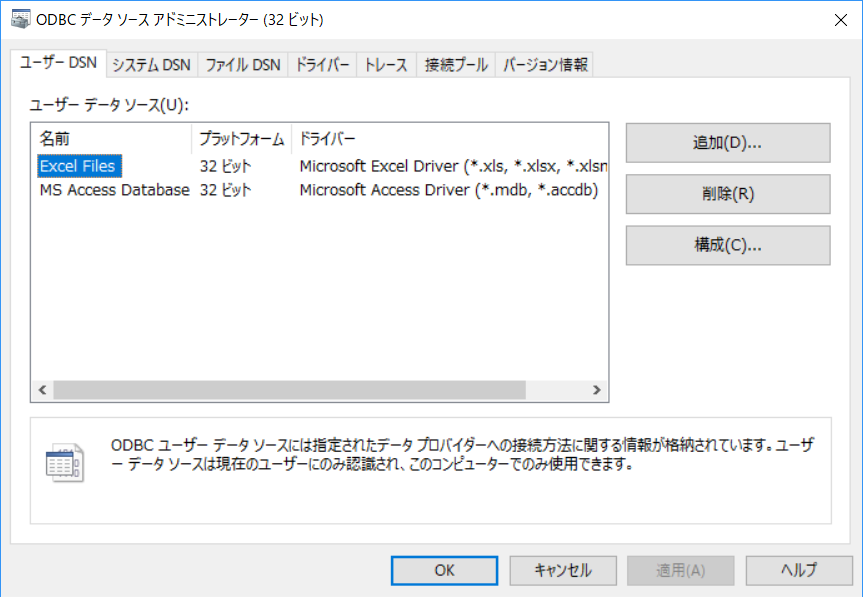


５．ODBC接続設定を行うので、ODBCアドミニストレータを起動する



32ビット版を起動

６．ODBC接続設定を追加する。



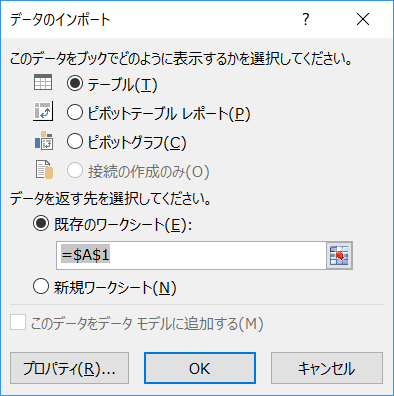
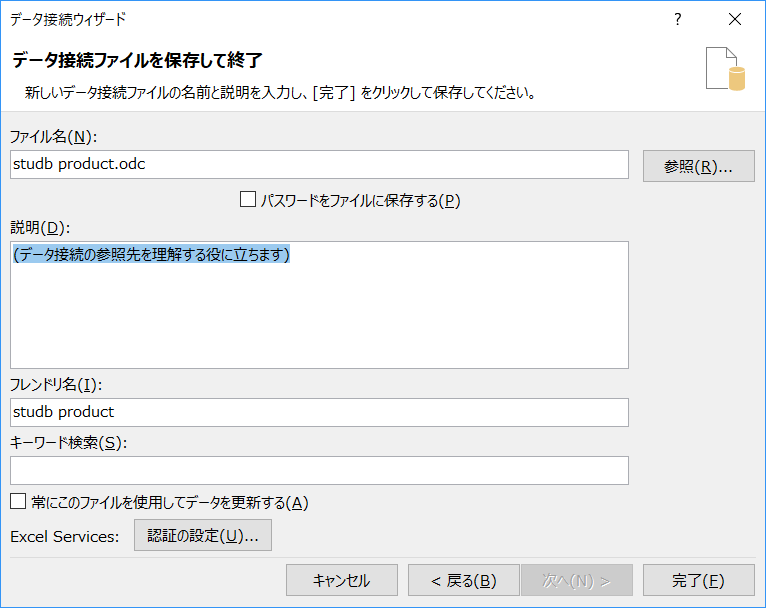
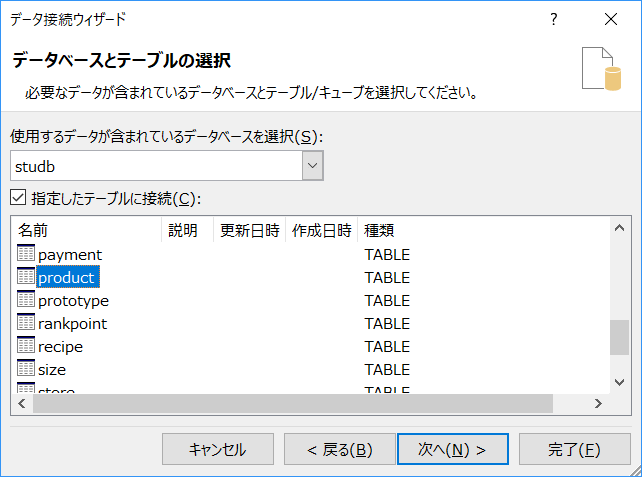
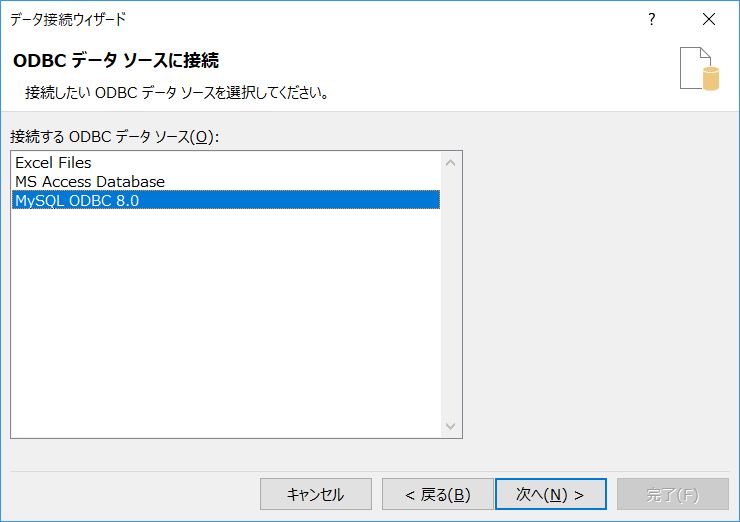
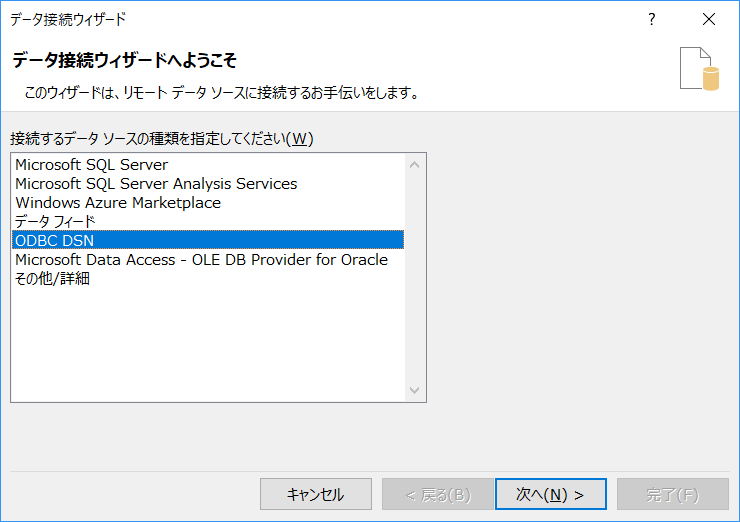
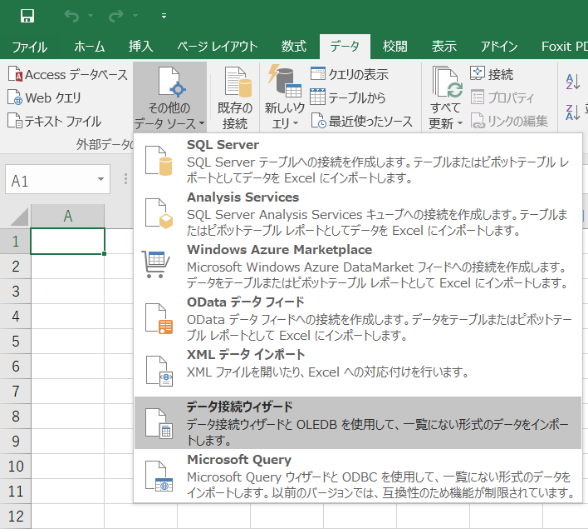
Data Source Name：MySQL ODBC 8.0

TCP/IP Server：localhost

User：root

Password：root

７．Excelからデータベースに接続する



８．Excelにデータベースの情報が表示されていればOK

【まとめ】

**・MySQL Installerで「Connector/ODBC」をインストール**

**・WindowsのODBCアドミニストレータでデータベース情報を登録**

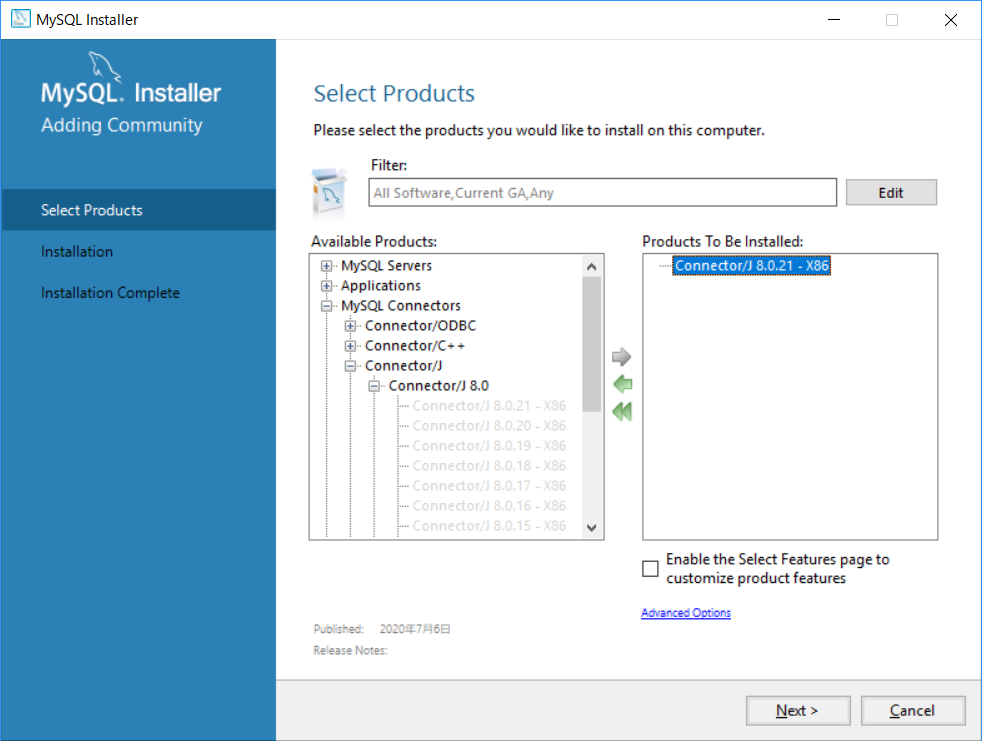
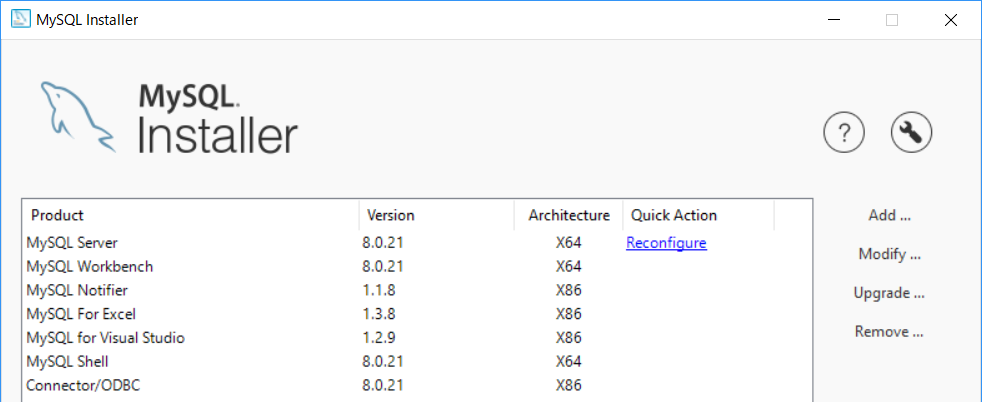
**・ExcelなどからODBCを介してデータベースに接続して、データを取得する**

# Javaからのデータベース接続

JavaからMySQLに接続するには、Java版のMySQLコネクタである「Connector/J」を使用します。また、Java側ではJDBC API（java.sql、javax.sql）が必要ですがこちらはJava SEに含まれているので、基本的にはJavaがインストールされていれば使用できます。

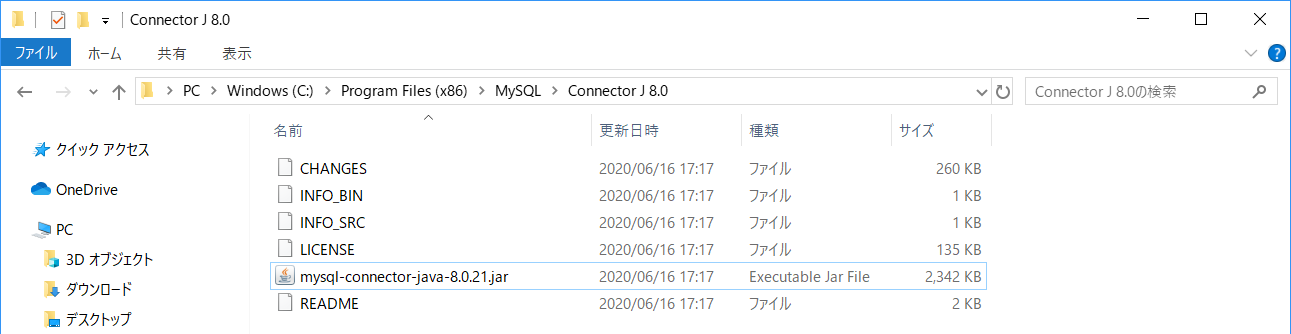
ハンズオン　Javaからデータベースに接続する

１．MySQL InstallerでJava用のMySQLコネクタをインストールする

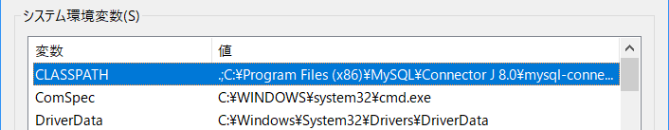
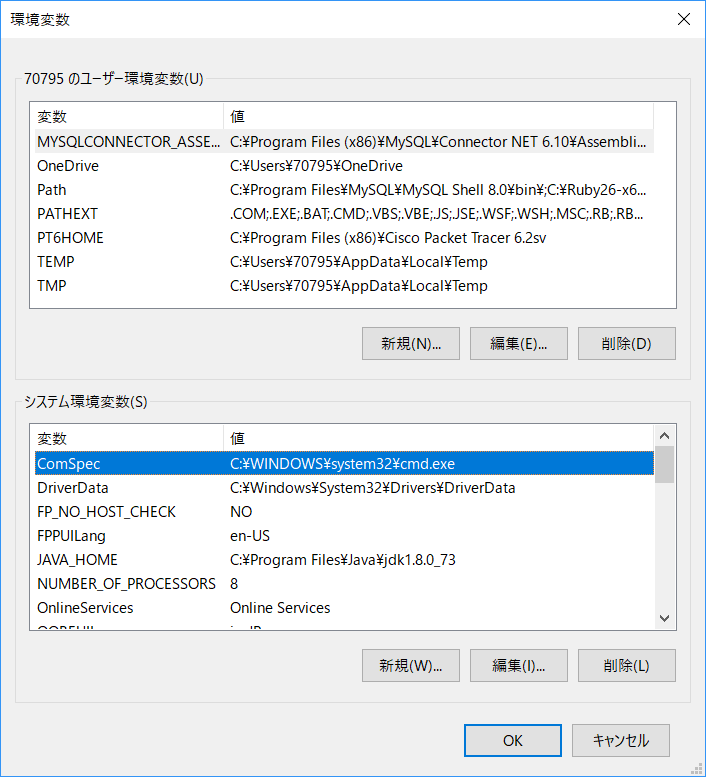
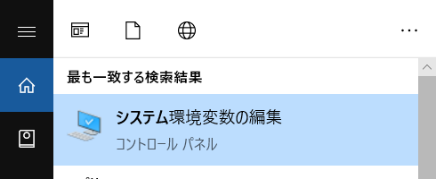


２．Java用MySQLコネクタのjarファイルが存在するか確認する。

C:\Program Files (x86)\MySQL\Connector J 8.0\mysql-connector-java-8.0.21.jar



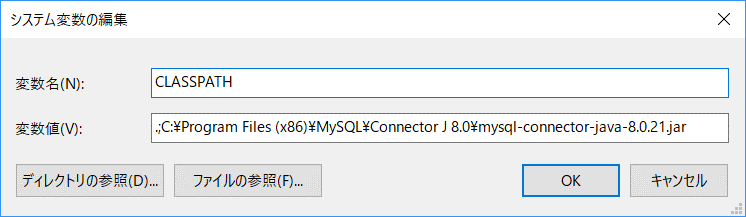
３．jarファイルを認識させるために、環境変数にCLASSPATHを設定する。



変数名：CLASSPATH

変数値：.;C:\Program Files (x86)\MySQL\Connector J 8.0\mysql-connector-java-8.0.21.jar

**※すでにCLASSPATHがある学生は、「新規」では無く「編集」で追加を行うこと！！**



４．Javaプログラムからデータベースに接続する

・MySQLConnect.java

|  |
| --- |
| //JCDB APIをインポートする  import java.sql.\*;  public class MySQLConnect{      public static void main(String[] args) throws Exception{          System.out.println("MySQL接続プログラム開始\n");            //データベース接続に必要な変数を宣言          Connection conn = null;          Statement st = null;          ResultSet rs = null;          try{              //ドライバを読み込む(環境変数でCLASSPAHTの設定が必要)              Class.forName("com.mysql.cj.jdbc.Driver");              //データベース情報              String dbname = "studb";   //接続データベース              String dbuser = "dbuser";  //データベースユーザ              String dbpass = "ecc";     //ログインパスワード              //JDBCドライバ情報設定              String dbinfo =               "jdbc:mysql://localhost/"+dbname+"?characterEncoding=UTF-8&serverTimezone=JST";              //データベースに接続する              conn = DriverManager.getConnection(dbinfo, dbuser, dbpass);              conn.setAutoCommit(false); //オートコミットを無効                st = conn.createStatement();                //INSERT文を作成する              String isql  = "INSERT INTO product (product\_no, pname, category, price ) ";                     isql += "VALUES ('1015', 'バナナアイス', 'サイド', 370 )";                //SQL文を出力しておく              System.out.println(isql);                //executeUpdate関数でUPDATE、INSERTまたは、CREATEなどのDDL文が実行できる。              //戻り値は、データ処理件数              int cnt = st.executeUpdate(isql);              System.out.println(cnt +"件のデータを登録しました\n");                  //SELECT文を送信する              String ssql = "SELECT \* FROM product WHERE pname LIKE '%アイス%'";              //SQL文を出力しておく              System.out.println(ssql);              //SELECT文の場合は、executeQuery関数を使用する。              //こちらは実行結果が戻り値になるので結果をデータセットに格納する              rs = st.executeQuery(ssql);              //データベースの値を受け取る変数宣言              String productNo;              String productName;              String category;              int price;                //データセットにデータが存在する間繰り返す              while(rs.next()){                  productNo   = rs.getString("PRODUCT\_NO");                  productName = rs.getString("PNAME");                  category    = rs.getString("CATEGORY");                  price       = rs.getInt("PRICE");                  // １行ごとに商品情報を画面に出力する                  String wkSt = "商品番号："+ productNo +                                " 商品名："+ productName +                                " カテゴリ："+ category +                                " 価格：" + price ;                  System.out.println(wkSt);              }                //ハンズオンの為、ロールバックを行う。              conn.rollback();              System.out.println("\nロールバックしました");          }catch(Exception e){              //エラーの場合は、ロールバックを行う。              if(conn != null){                  conn.rollback();              }              e.printStackTrace();          }finally{              //MySQLとの接続を切断する              if(rs != null){ rs.close(); }              if(st != null){ st.close(); }              if(conn != null){ conn.close(); }          }      }  } |

５．Javaの実行結果に商品情報が出力されていればOK

・java.sqlパッケージについて

<https://docs.oracle.com/javase/jp/8/docs/api/java/sql/package-summary.html>

JavaでDBに格納されたデータにアクセスして処理するAPI

|  |  |
| --- | --- |
| 主なインタフェース | 説明 |
| Connection | データベースとの接続(セッション)を行う  コミットやロールバックなどトランザクション制御やセッションに関する処理を受け持つ |
| Statement | 静的SQL文を実行して、作成された結果を返すオブジェクト  SQLの命令とその結果に関する処理を受け持つ |
| ResultSet | データベースの結果を表すデータ表。  通常、SELECT文を実行することで生成される |

Java

MySQL

Result

Set

Table

SQL

StatementでSQL文を発行して

データ操作を実行

ConnectionでDBに接続

【まとめ】

**・MySQL Installerで「Connector/J」をインストール**

**・JavaプログラムでMySQLコネクタを認識させるため、CLASSPATH環境変数を設定**

**・JavaプログラムでJDBCドライバにMySQL接続情報をセットしてデータベースに接続**

# PHPからのデータベース接続

PHPの場合は、対応するMySQLコネクタはありません。PHPの拡張機能であるmysqliやPDO\_MySQLなどを使用してMySQLサーバとの接続を行います。

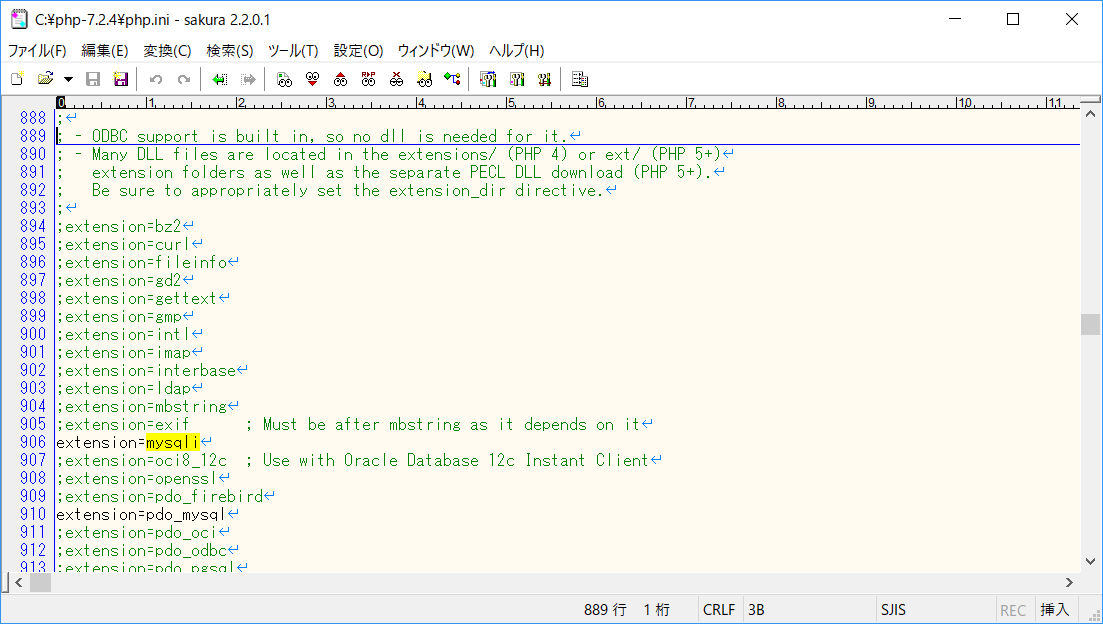
PHPを介してのデータベース接続は、他の授業で学習していますのでこちらでは

ハンズオンは実施せずに簡単な紹介に留めておきます。

Windows環境でのPHPの場合、MySQLドライバはphp\_mysqli.dllです。

環境ファイルを編集することによる有効化することが出来ます。

・php.iniファイル



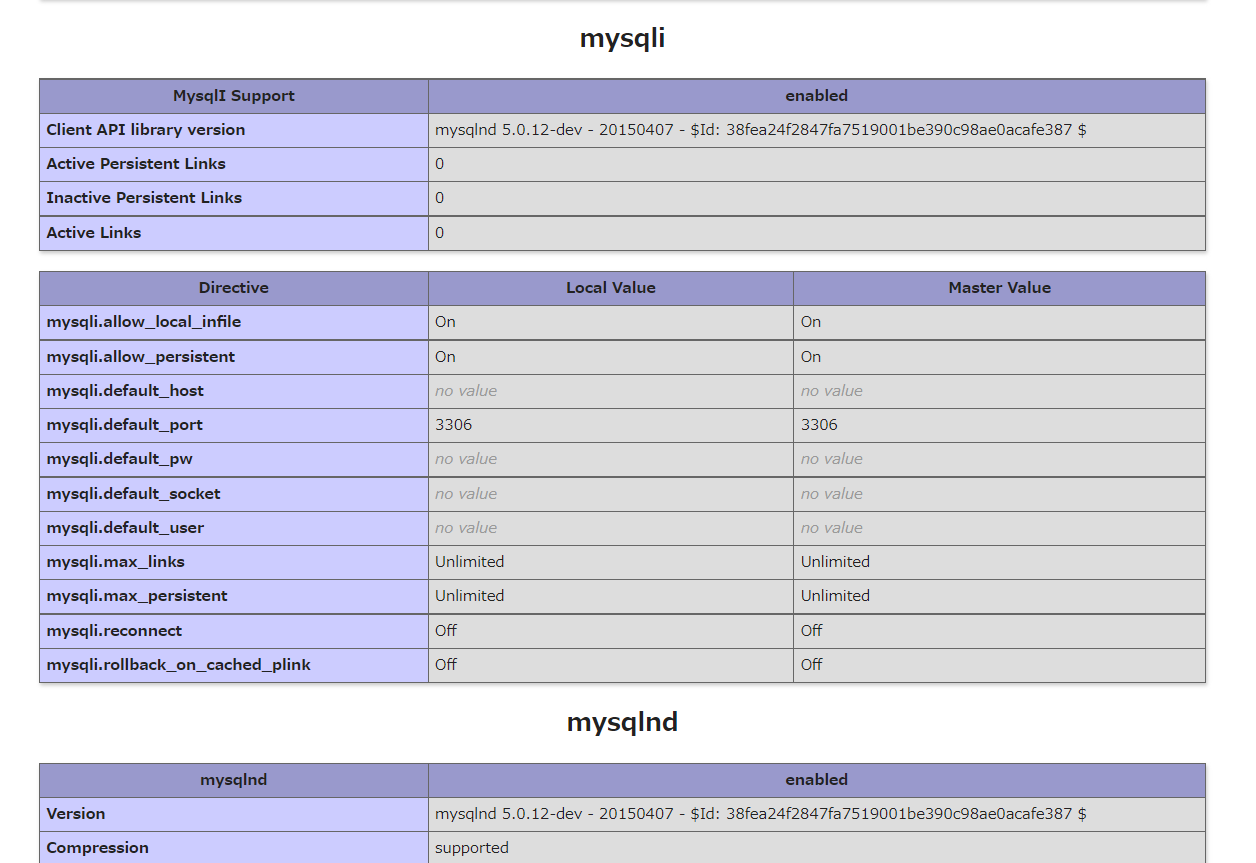
；(コメント)を削除するとmysqliが有効化される。

環境ファイルを編集したらWebサーバの再起動が必要。

※PHP 5.3.0 以降のPHPはデフォルトで有効化になっています

mysqliが有効の場合は、phpinfo()を実行した時に、以下のように情報が表示される。

※もし表示されない場合は、何らかの理由でmysqliが無効化されています。



・PHPからのデータベース接続

