設計・製造支援アプリケーション構築プラットフォーム

MZ Platform

データベースアクセスチュートリアル

National Institute of
Advanced Industrial Science
and Technology
AIST

独立行政法人
産業技術総合研究所

データベースアクセスチュートリアル

このチュートリアルでは、『データベースアクセス』コンポーネントを使用した簡単なデータベースアプリケーションの構築を通して、データベースへのアクセス方法を習得します。

◆目次

◆図表目次

◇図目次

◇表目次

データベースを操作してみよう

ここではデータベースの操作を行うコンポーネントである『データベースアクセス』を使用します。『データベースアクセス』を使って SQL データベースの操作を行う簡単なアプリケーションを構築してみましょう。このアプリケーションの構築には『複合コンポーネント』を使用します。

Step.1 データベース操作の手順を理解する

SQL データベースは、SQL（Structured Query Language）と呼ばれる問い合わせ言語を用いて、表として格納されたデータの操作を行うデータベース管理システムです。ここでは、「表」をテーブル、表の「行」をレコード、表の「列」をフィールドと呼ぶことにします。SQL ではデータの取り出し（検索）、テーブルの作成、レコードの追加、削除、更新といった操作をサポートしています。SQL の詳細に関しましては市販の参考書をご覧下さい。

データベースの操作は、『データベースへ接続する』、『データベースを検索する（SQL 文をデータベースへ送信し結果を受信する）』、『データベースから切断する』、という手順で行います。このレッスンでは、この手順を実現するため、『データベースアクセス』コンポーネントと次の複合コンポーネントを作成します。

- **データベース接続フレーム**
- **データベース検索フレーム**

データベースの接続と切断には、データベース接続フレームと『データベースアクセス』を、データベースの検索には、データベース検索フレームと『データベースアクセス』を使用します。このレッスンで作成するアプリケーションの実行画面を下図に示します。

それでは順に複合コンポーネントを作成しながら、アプリケーションを構築していきましょう。

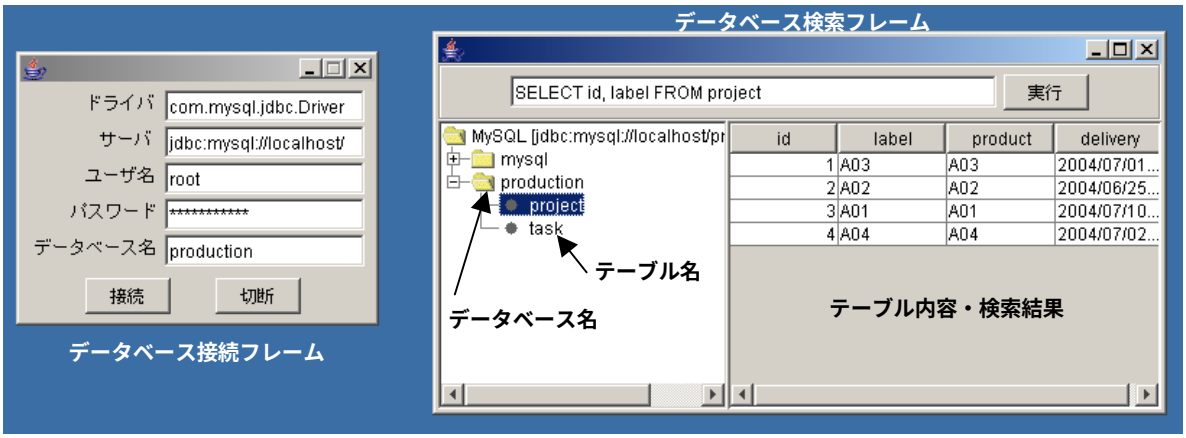

図 1 データベースアクセスアプリケーションの画面イメージ

注意：このアプリケーションの実行には、JDBC（Java DataBase Connectivity）を介してアクセス可能な MySQL 等の SQL データベースが必要です。MySQL のインストールに関しては、「サンプルアプリケーションマニュアル」の「2.5.1 準備 －MySQL 関連ファイルの入手とインストール－」、または、「工程管理システムマニュアル」を参照してください。また、以下では、サンプルのデータベース「production」が既に存在することを仮定しています（他のデータベース、テーブルでも構いません）。サンプルのデータベース「production」に関しては、「工程管理システムマニュアル」を参照してください。

Step.2 データベースに接続する

データベースにアクセスするため、『データベースアクセス』コンポーネントを用います。データベースに接続するためには、『データベースアクセス』のメソッド『データベースに接続する(String, String, String, String) 』を用います。各引数の意味は次のとおりです。

第 1 引数：ドライバ名
第 2 引数：データベースの URL（サーバ名 + データベース名）
第 3 引数：ユーザ名
第 4 引数：パスワード

はじめに、これらの情報を入力する複合コンポーネント「データベース接続フレーム」を作成しましょう。ビルダー画面上でマウスを右クリックし、[複合コンポーネント追加]－[コンポーネント生成]を選択し、複合コンポーネントを生成します。このコンポーネントをダブルクリックし、複合コンポーネントの構築画面に移ります。複合コンポーネントには「データベース接続フレーム」と名前を付けることとします。

複合コンポーネントで使用するコンポーネントは、『フレーム』（１つ)、『ボタン』（２つ)、『ラベル』（５つ)、『テキストフィールド』（４つ)、『パスワード入力フィールド』（１つ)、『文字列格納変数』（１つ）です。コンポーネントのカテゴリは以下のとおりです。

フレーム	**：[画面構成部品] － [ウィンドウ] － [フレーム]**
ボタン	**：[画面構成部品] － [ボタン] － [ボタン]**
ラベル	**：[画面構成部品] － [テキスト] － [ラベル]**
テキストフィールド	**：[画面構成部品] － [テキスト] － [テキストフィールド]**
パスワード入力フィールド	**：[画面構成部品] － [テキスト] － [パスワード入力フィールド]**
文字列格納変数	**：[処理部品] － [変数] － [文字列格納変数]**

[画面編集]ボタンを押して、文字列格納変数(ID:1-14)以外のコンポーネントを以下の画面イメージに従って配置し、属性等を設定しましょう。

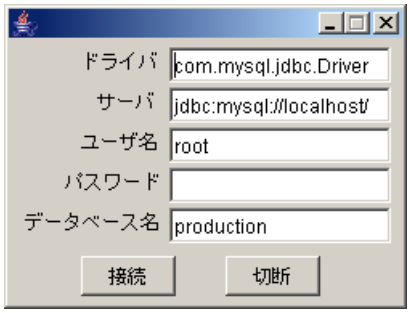

図 2 データベース接続フレームの画面イメージ

[接続]ボタンが押されたときに、対応するテキストフィールド及びパスワード入力フィールドからテキストを取得し、『データベースアクセス』のメソッド『データベースに接続する(String, String, String, String) 』に設定すればよいのですが、注意が必要です。このメソッドの第１引数、第３引数、第４引数は対応するテキストフィールド又はパスワード入力フィールドから取得し設定すればよいのですが、第２引数（データベースの URL）は「サーバ名」+「データベース名」になります。従って、テキストをつなぎ合わせる処理が必要になります。この処理は『文字列格納変数』で行うことができます。[接続]ボタンが押されたときに、この処理が実行されるようにコンポーネント間を接続しましょう。

表 1 コンポーネント接続関係（サーバ名とデータベース名の結合）

項目	内容
イベント発生元コンポーネント	ボタンコンポーネント（ID:1-2）
発生イベント	アクションイベント
イベント番号	定常起動
接続先	①文字列の設定 ◇接続先コンポーネント 文字列格納変数コンポーネント（ID:1-14） ◇起動メソッド 文字列を設定する(String) ＜引数：文字列＞ 取得方法：『メソッド戻り値』 取得先コンポーネント：テキストフィールド（ID:1-10） 取得メソッド：“テキストを取得する()”
	②文字列の追加 ◇接続先コンポーネント 文字列格納変数コンポーネント（ID:1-14） ◇起動メソッド 指定した文字列と連結して置き換える（String） ＜引数：連結する文字列＞ 取得方法：『メソッド戻り値』 取得先コンポーネント：テキストフィールド（ID:1-12） 取得メソッド：“テキストを取得する()”

また、フレームが閉じられたこと、ボタンが押されたことを複合コンポーネントの外に知らせる（イベントを外部に伝播する）ために、次の接続を行います。ただし、発生させるイベントがどれもアクションイベントであり、区別をつけるため、それぞれイベント番号(0～2)を設定します。

表 2 コンポーネント接続関係（複合コンポーネント外へのイベントの伝播）

項目	内容
イベント発生元コンポーネント	フレームコンポーネント（ID:1-1）
発生イベント	アクションイベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース接続フレーム（ID:1） ◇起動メソッド イベント番号を指定してイベントを伝播させる(PFEvent, int) ＜引数：対象イベント＞ 取得方法：『イベント』 ＜引数：指定するイベント番号＞ 取得方法：『固定値』 値：“0”

項目	内容
イベント発生元コンポーネント	ボタンコンポーネント（ID:1-2）
発生イベント	アクションイベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース接続フレーム（ID:1） ◇起動メソッド イベント番号を指定してイベントを伝播させる(PFEvent，int) ＜引数：対象イベント＞ 取得方法：『イベント』 ＜引数：指定するイベント番号＞ 取得方法：『固定値』 値：“１”

項目	内容
イベント発生元コンポーネント	ボタンコンポーネント（ID:1-3）
発生イベント	アクションイベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース接続フレーム（ID:1） ◇起動メソッド イベント番号を指定してイベントを伝播させる(PFEvent，int) ＜引数：対象イベント＞ 取得方法：『イベント』 ＜引数：指定するイベント番号＞ 取得方法：『固定値』 値：“２”

更に、[接続]ボタンが押された際に、外部のコンポーネントである『データベースアクセス』からメソッド『データベースに接続する(String, String, String, String) 』の４つの引数に相当する文字列を取得可能なように、メソッドを公開します。

表 3 複合コンポーネントの公開メソッド（その１）

コンポーネント名	メソッド名	公開メソッド名(変更後)
フレーム(ID:1-1)	フレームを表示する()	同左
フレーム(ID:1-1)	フレームを閉じる()	同左
テキストフィールド(ID:1-9)	テキストを取得する()	ドライバ名を取得する()
テキストフィールド(ID:1-11)	テキストを取得する()	ユーザ名を取得する()
パスワード入力フィールド(ID:1-13)	パスワード文字列を取得する()	同左
文字列格納変数(ID:1-14)	文字列を取得する()	データベースの URL を取得する()

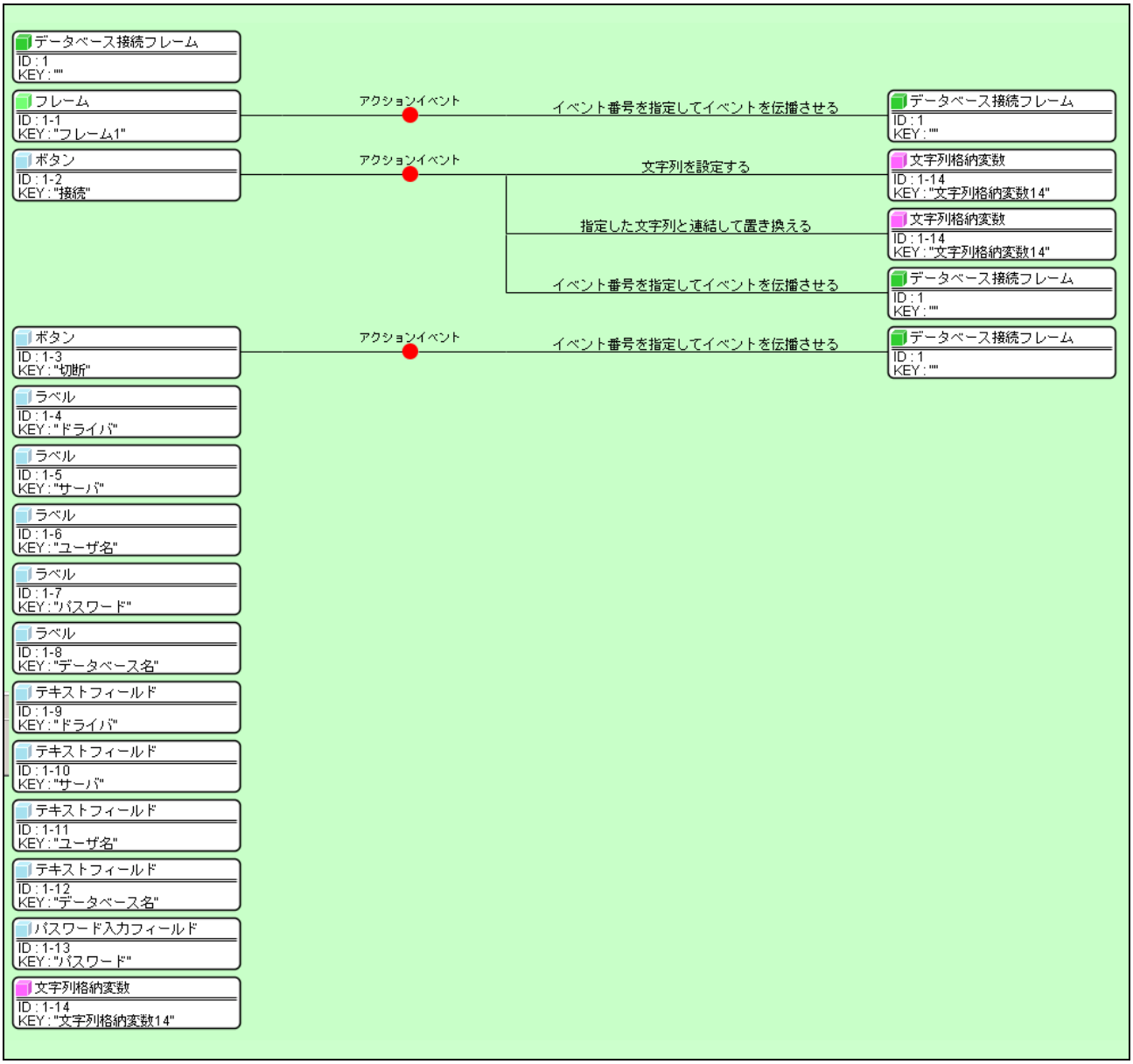

図 3 データベース接続フレームのコンポーネント接続関係

右上の  を押して、アプリケーションのビルダー画面に戻り、『データベースアクセス』コンポーネントと接続しましょう。はじめに、『データベースアクセス』コンポーネントを生成します。カテゴリは次のとおりです。

データベースアクセス　：[入出力] － [データベース] － [データベースアクセス]

データベース接続フレーム(ID:1)の[接続]ボタンが押された際にデータベースに接続するようにコンポーネント間を接続します。[接続]ボタンが押された際に、複合コンポーネントからアクションイベントがイベント番号“1”で発生することにしましたので、次のようにコンポーネント間を接続します。

表 4 コンポーネント接続関係（データベースへの接続）

項目	内容
イベント発生元コンポーネント	**データベース接続フレーム（ID:1）**
発生イベント	**アクションイベント**

イベント番号	**1**
接続先	**◇接続先コンポーネント** **データベースアクセス（ID:2）** **◇起動メソッド** **データベースに接続する(String, String, String, String)** **＜引数：ドライバ名＞** **取得方法：『メソッド戻り値』** **取得先コンポーネント：データベース接続フレーム（ID:1）** **取得メソッド：“ドライバ名を取得する()”** **＜引数：データベースの URL＞** **取得方法：『メソッド戻り値』** **取得先コンポーネント：データベース接続フレーム（ID:1）** **取得メソッド：“データベースの URL を取得する()”** **＜引数：ユーザ名＞** **取得方法：『メソッド戻り値』** **取得先コンポーネント：データベース接続フレーム（ID:1）** **取得メソッド：“ユーザ名を取得する()”** **＜引数：パスワード＞** **取得方法：『メソッド戻り値』** **取得先コンポーネント：データベース接続フレーム（ID:1）** **取得メソッド：“パスワードを取得する()”**

データベース接続フレーム(ID:1)の[切断]ボタンが押された際には、アクションイベントがイベント番号“2”で発生するように設定しました。このときにデータベースとの接続を切断するように、次のようにコンポーネント間を接続します。

表 5 コンポーネント接続関係（データベースとの接続の切断）

項目	内容
イベント発生元コンポーネント	**データベース接続フレーム（ID:1）**
発生イベント	**アクションイベント**
イベント番号	**2**
接続先	**◇接続先コンポーネント** **データベースアクセス（ID:2）** **◇起動メソッド** **データベースとの接続を切断する ()**

このステップの最後に、データベース接続フレーム(ID:1)が閉じたとき（アクションイベント、イベント番号“0”）にアプリケーションを終了する、アプリケーションが開始したときにデータベース接続フレーム(ID:1)を表示する、アプリケーションが終了したときにデータベース接続フレーム(ID:1)を閉じる、ようにコンポーネント間を接続します。

表 6 コンポーネント接続関係（アプリケーションの開始・終了処理）（その１）

項目	内容
イベント発生元コンポーネント	データベース接続フレーム（ID:1）
発生イベント	アクションイベント
イベント番号	0
接続先	◇接続先コンポーネント アプリケーション ◇起動メソッド アプリケーションを終了する()

項目	内容
イベント発生元コンポーネント	アプリケーション
発生イベント	アプリケーション開始イベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース接続フレーム（ID:1） ◇起動メソッド フレームを表示する（）

項目	内容
イベント発生元コンポーネント	アプリケーション
発生イベント	アプリケーション終了イベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース接続フレーム（ID:1） ◇起動メソッド フレームを閉じる()

それでは、[実行（設定可）]ボタンを押して、ここまで作成したアプリケーションを実行させ、データベースに接続してみましょう。はじめに、ドライバ名、サーバ名、ユーザ名、パスワード、データベース名を入力します。例えば、MySQL に JDBC ドライバを用いて接続する場合には、ドライバ名には“com.mysql.jdbc.Driver”、サーバー名には“jdbc:mysql://ホスト名/”と記入します。MZ Platform と同じパソコンに MySQL がインストールされている場合、ホスト名は“localhost”と記入します。ユーザ名、パスワード、データベース名には、それぞれデータベースインストール時に設定したものを入力します。例えば、MZ Platform に添付のデータベース“production”をインストール済みの場合は、データベース名は“production”、ユーザ名とパスワードはデータベースインストール時に設定した値を記入します。この状態で、念のためアプリケーションを保存することをお勧めします。それでは[接続]ボタンを押してみましょう。正しく接続された場合には、何も起こりません。接続に失敗した場合、次のようなエラーダイアログが表示されます（他のダイアログが表示される場合もあります）。このようなダイアログが出た場合には、ドライバ名、サーバ名、ユーザ名、パスワード、データベース名が正しく入力されているかもう一度確認して、再度接続を試みてください。

図 4 データベース接続失敗のエラー

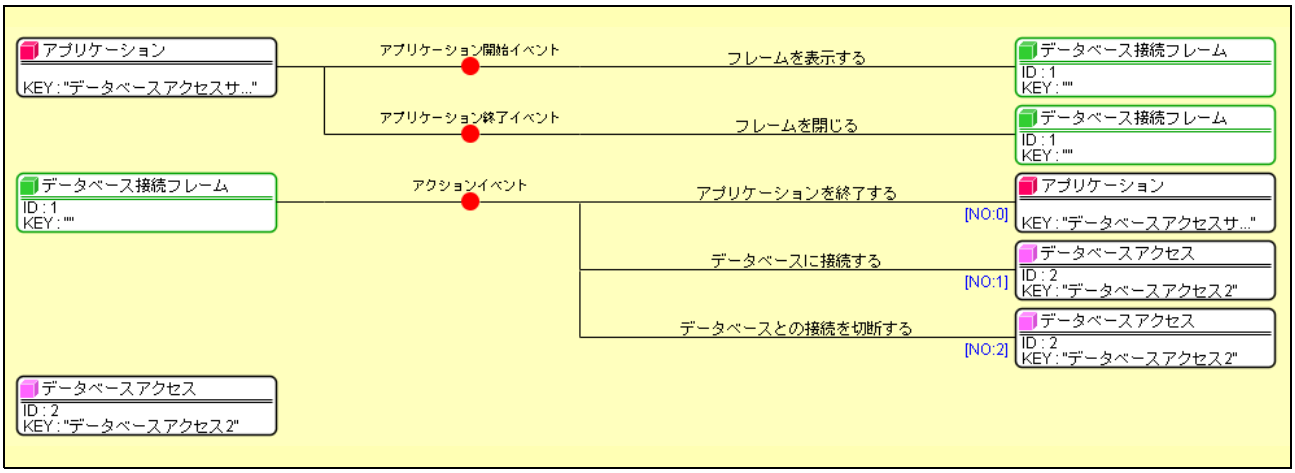

図 5 データベースアクセスアプリケーションのコンポーネント接続関係（その１）

Step.3 データベースを検索する

データベースに接続ができましたので、今度はデータベースを検索してみましょう。『データベースアクセス』コンポーネントでは次の手順で、検索を実行し結果を得ることができます。

検索実行：SQL 文（文字列）を引数としてメソッド『SQL 文を実行する（String)』を実行する

検索結果：テーブルデータがデータ生成イベントの内包データとして返ってくる

はじめに、検索を実行する複合コンポーネント「データベース検索フレーム」を作成しましょう。ビルダー画面上でマウスを右クリックし、[複合コンポーネント追加]－[コンポーネント生成]を選択し、複合コンポーネントを生成します。このコンポーネントをダブルクリックし、複合コンポーネントの構築画面に移ります。複合コンポーネントには「データベース検索フレーム」と名前を付けることとします。

使用するコンポーネントは、『フレーム』、『テーブル』、『パネル』、『テキストフィールド』、『ボタン』、『イベント生成』です。コンポーネントのカテゴリは以下のとおりです。

フレーム	**：[画面構成部品]**	**－ [ウィンドウ]**	**－**	**[フレーム]**
テーブル	**：[画面構成部品]**	**－ [テーブル]**	**－**	**[テーブル]**
パネル	**：[画面構成部品]**	**－ [パネル]**	**－**	**[パネル]**
テキストフィールド	**：[画面構成部品]**	**－ [テキスト]**	**－**	**[テキストフィールド]**
ボタン	**：[画面構成部品]**	**－ [ボタン]**	**－**	**[ボタン]**
イベント生成	**：[処理部品]**	**－ [イベント]**	**－**	**[イベント生成]**

テキストフィールド(ID:3-4)は検索の際に必要となる SQL 文を入力するために使用し、テーブル(ID:3-2)は検索結果を表示するために使用します。[画面編集]ボタンを押して、イベント生成(ID:3-6)以外のコンポーネントを以下の画面イメージに従って配置し、属性等を設定しましょう。ここで、テキストフィールド(ID:3-4)とボタン(ID:3-5)はパネル(ID:3-3)に貼り付け、「領域配置」のモードで、パネル(ID:3-3)をフレーム(ID:3-1)の「North」に、テーブル(ID:3-2)を「Center」に配置します。

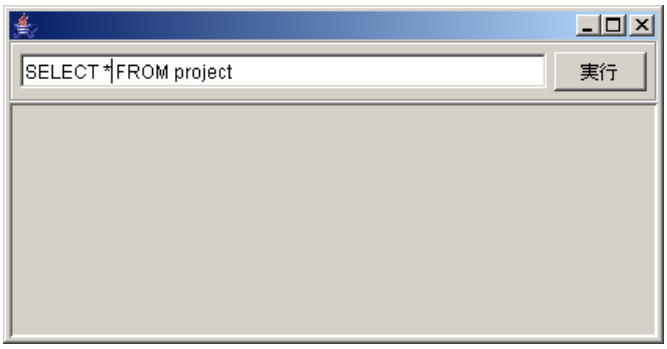

図 6 データベース検索フレームの画面イメージ

ここでは、[実行]ボタンが押されたときに、テキストフィールド(ID:3-4)に記述された SQL 文を、「データ設定イベント」に内包させて、複合コンポーネント外部に伝播することにします。ボタンから発生するイベントは「アクションイベント」ですので、「データ設定イベント」を生成しなければなりません。任意のイベントを生成するコンポーネントが『イベント生成』コンポーネントです。「データ設定イベント」を生成するにはメソッド『データ設定イベントの発生（Object, int)』を用います。第 1 引数と第 2 引数は次のとおりです。

第 1 引数：設定データ（データ設定イベントに内包されるデータ）

第 2 引数：イベント番号

[実行]ボタンが押されたときに、テキストフィールド（ID:3-4）に記述された SQL 文を、「データ設定イベント」に内包し、複合コンポーネント外部に伝播するように、次のようにコンポーネント間を接続します。また、フレームが閉じられたことを複合コンポーネントの外部に伝播するための接続も行っておきます。ただ

し、イベント番号は“0”に設定します。

表 7 コンポーネント接続関係（SQL 文の伝播）

項目	内容
イベント発生元コンポーネント	ボタン（ID:3-5）
発生イベント	アクションイベント
イベント番号	定常起動
接続先	◇接続先コンポーネント イベント生成（ID:3-6） ◇起動メソッド データ設定イベントの発生（Object，int） ＜引数：設定データ＞ 取得方法：『メソッド戻り値』 取得先コンポーネント：テキストフィールド（ID:3-4） 取得メソッド：“テキストを取得する()” ＜引数：イベント番号＞ 取得方法：『固定値』 値：“0”

項目	内容
イベント発生元コンポーネント	イベント生成（ID:3-6）
発生イベント	データ設定イベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース検索フレーム（ID:3） ◇起動メソッド イベントを伝播させる(PFEvent) ＜引数：対象イベント＞ 取得方法：『イベント』

項目	内容
イベント発生元コンポーネント	フレームコンポーネント（ID:3-1）
発生イベント	アクションイベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース検索フレーム（ID:3） ◇起動メソッド イベント番号を指定してイベントを伝播させる(PFEvent，int) ＜引数：対象イベント＞ 取得方法：『イベント』 ＜引数：指定するイベント番号＞ 取得方法：『固定値』 値：“0”

SQL 文による検索結果を外部のデータベースアクセスコンポーネント(ID:2)から受け取れるように、テーブル(ID:3-2)のメソッド『テーブルデータを設定する(PFObjectTable)』を公開メソッドとして公開します。また同時に、フレームを外部から表示又は閉じられるように、メソッドを公開します。

表 8 複合コンポーネントの公開メソッド（その２）

コンポーネント名	メソッド名	公開メソッド名(変更後)
フレーム（ID:3-1）	フレームを表示する()	同左
フレーム（ID:3-1）	フレームを閉じる()	同左
テーブル（ID:3-2）	テーブルデータを設定する(PFObjectTable)	同左

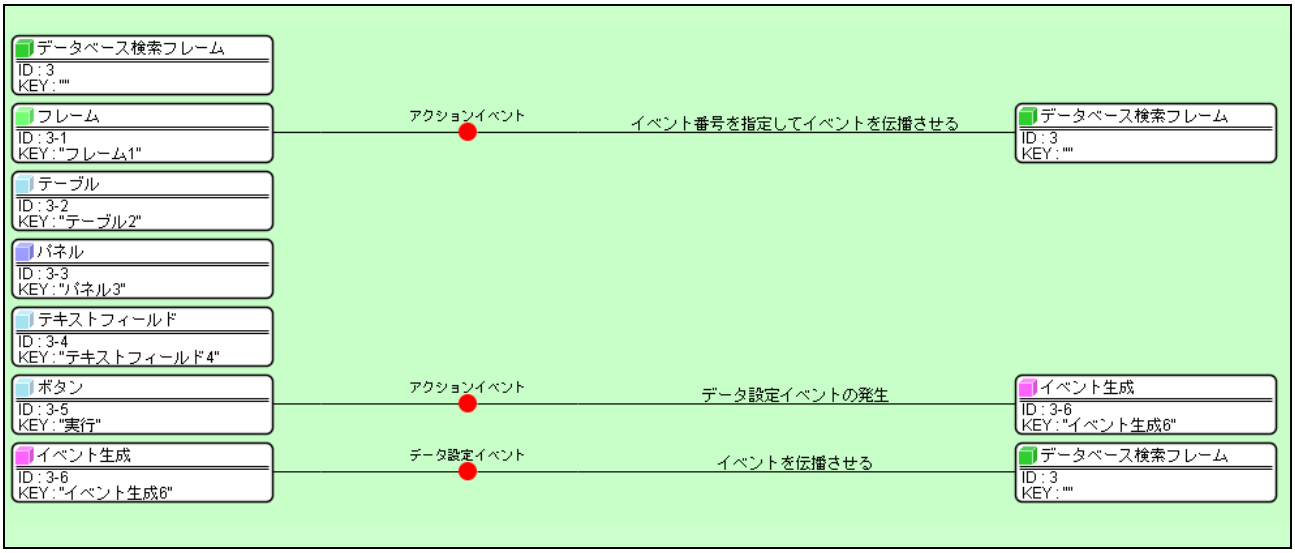

図 7 データベース検索フレームのコンポーネント接続関係（その１）

それでは、右上の  を押して、アプリケーションのビルダー画面に戻り、データベースアクセスコンポーネント(ID:2)と接続しましょう。はじめに、データベース検索フレーム(ID:3)からのデータ設定イベントに内包されている SQL 文を受け取り、この文字列を引数としてデータベースアクセス(ID:2)のメソッド『SQL 文を実行する（String)』を実行します。次に、検索実行時にデータベースアクセス(ID:2)から発生するデータ生成イベントに内包されるテーブルデータをデータベース検索フレーム(ID:3)のテーブルに設定します。コンポーネント間を次のように接続します。

表 9 コンポーネント接続関係（SQL 文の実行と結果の取得）

項目	内容
イベント発生元コンポーネント	データベース検索フレーム（ID:3）
発生イベント	データ設定イベント
イベント番号	0
接続先	◇接続先コンポーネント データベースアクセス（ID:2） ◇起動メソッド

	SQL 文を実行する(String) ＜引数：SQL 文＞ 取得方法：『イベント内包』 取得メソッド：“ イベント対象データ ”

項目	内容
イベント発生元コンポーネント	データベースアクセス（ID:2）
発生イベント	データ生成イベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース検索フレーム（ID:3） ◇起動メソッド テーブルデータを設定する（PFObjectTable） ＜引数：テーブルデータ＞ 取得方法：『イベント内包』 取得メソッド：“ イベント対象データ ”

このステップの最後に、データベース検索フレーム(ID:3)が閉じたとき（アクションイベント、イベント番号“ 0 ”）にアプリケーションを終了する、アプリケーションが開始したときにデータベース検索フレーム(ID:3)を表示する、アプリケーションが終了したときにデータベース検索フレーム(ID:3)を閉じる、ようにコンポーネント間を接続します。

表 10 コンポーネント接続関係（アプリケーションの開始・終了処理）（その２）

項目	内容
イベント発生元コンポーネント	データベース検索フレーム（ID:3）
発生イベント	アクションイベント
イベント番号	0
接続先	◇接続先コンポーネント アプリケーション ◇起動メソッド アプリケーションを終了する()

項目	内容
イベント発生元コンポーネント	アプリケーション
発生イベント	アプリケーション開始イベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース検索フレーム（ID:3） ◇起動メソッド フレームを表示する（）

項目	内容
イベント発生元コンポーネント	アプリケーション
発生イベント	アプリケーション終了イベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース検索フレーム（ID:3） ◇起動メソッド フレームを閉じる()

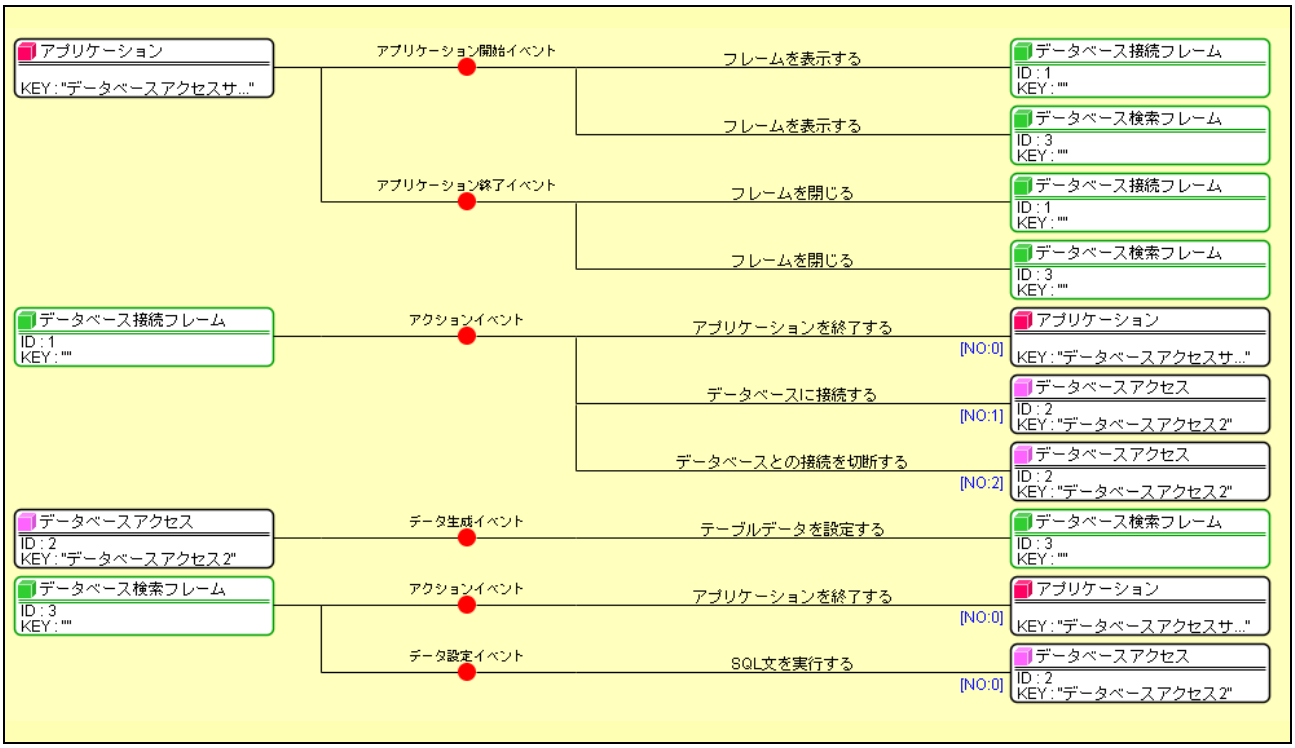

図 8 データベースアクセスアプリケーションのコンポーネント接続関係（その２）

それでは、ここまで作成したアプリケーションを保存後、実行させて、検索を実行してみましょう。データベースに接続後、「SELECT * FROM project」と入力し(project はデータベース production に存在するテーブル)、[実行]ボタンを押して検索を実行してみましょう。次の図のように検索結果が表示されましたか？

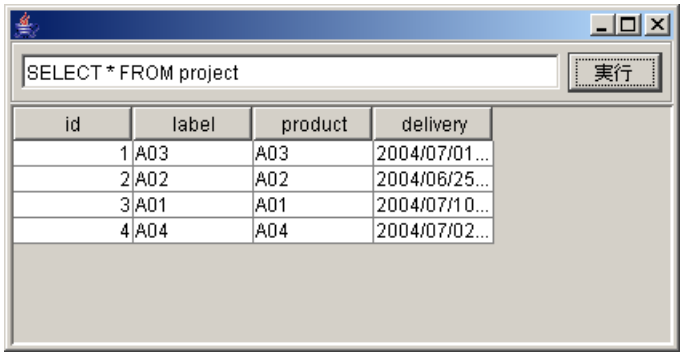

図 9 データベース検索実行の画面イメージ

注意：データベースの種類や設定によっては、検索結果に日本語が含まれる場合、文字化けする場合があります。この場合、使用する文字コードを指定することにより解決する場合があります。MySQL の場合には、データベース接続フレームのデータベース名の欄に、使用する文字コードをデータベース名とともに“ データベース名?useUnicode=true&characterEncoding=SJIS ”のように記述し、再接続します。すなわち、データベースアクセスコンポーネントのメソッド「データベースに接続する(String, String, String, String)」の第２引数（データベースの URL）に“ jdbc:mysql://ホスト名/データベース名?useUnicode=true&characterEncoding=SJIS ”と設定されることになります。詳細は各データベースの解説書をご覧下さい。

Step.4 データベース構造をツリー表示する

これまでの画面では、データベースがどのように構成され、どういうテーブルが存在するかわかりませんでした。そのため、インストール時に覚えていたデータベース名とテーブル名を使用して検索を実行しました。このステップではデータベースの構造がツリーで表示され、ツリー中のテーブル名をクリックしたときに、データが自動的に表示される便利な機能を追加します。『データベースアクセス』コンポーネントにはデータベースのツリー構造を得る便利なメソッドが用意されています。そのメソッドは『getDatabaseTree()』です。

はじめに、データベース検索フレーム(ID:3)に『ツリー』を追加し、上述のメソッドを使用してデータベースのツリーを表示してみましょう。データベース検索フレーム(ID:3)をダブルクリックして、複合コンポーネントの編集画面に移り、『ツリー』を追加します。『ツリー』のカテゴリは次のとおりです。

ツリー　：　[画面構成部品] － [ツリー] － [ツリー]

[画面編集]のモードで、このツリー（ID:3-7）をフレームの「West」に配置します。また、外部からツリーデータを設定できるようにツリー(ID:3-7)のメソッドを公開します。

表 11 複合コンポーネントの公開メソッド（その３）

コンポーネント名	メソッド名	公開メソッド名(変更後)
ツリー（ID:3-7）	ツリーデータを設定する(PFObjectTree)	同左

それでは、右上の  を押して、アプリケーションのビルダー画面に移り、データベース接続時にデータベースアクセス(ID:2)からデータベースのツリーデータを取得し、データベース検索フレーム(ID:3)にツリーデータを設定しましょう。コンポーネント間の接続は次のとおりです。

表 12 コンポーネント接続関係（ツリーデータの設定）

項目	内容
イベント発生元コンポーネント	データベース接続フレーム（ID:1）
発生イベント	アクションイベント
イベント番号	1
接続先	◇接続先コンポーネント データベース検索フレーム（ID:3） ◇起動メソッド ツリーデータを設定する（PFObjectTree） ＜引数：ツリーデータ＞ 取得方法：『メソッド戻り値』 取得先コンポーネント：データベースアクセス（ID:2） 取得メソッド：“ getDatabaseTree() ”

それでは実行してみます。データベース接続フレーム(ID:1)の[接続]ボタンを押してデータベースに接続後、次の図のようにデータベースの構成が表示されるはずです。

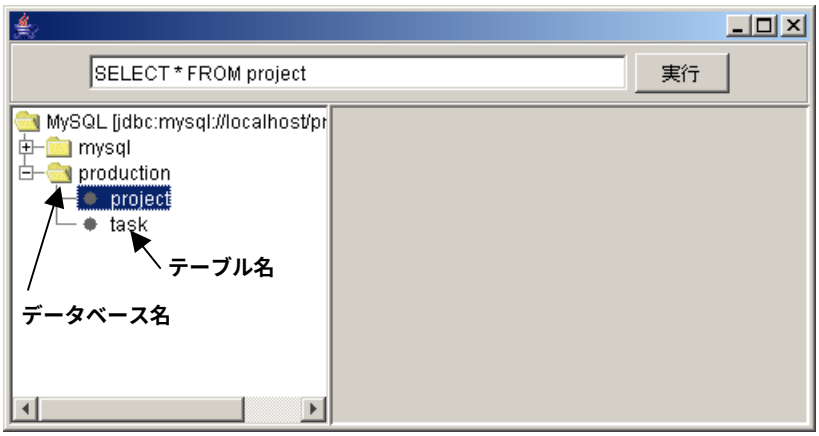

図 10 データベースツリー表示の画面イメージ（その１）

この状態でツリーのノードを開いてテーブル名をクリックしても何も起こりません。このステップの後半では、テーブル名がクリックされた際にテーブルの内容を右側のテーブル（ID:3-2）に表示するようにします。表示までにはいくつかの段階が必要です。はじめに、ツリー（ID:3-7）がクリックされた際に、テーブル名を取得する必要があります。またこの際、マウスクリックで自由にデータベースを変更できることから、データベースを指定しなおす必要があり、データベース名も同時に取得する必要があります。次に、テーブル名から、このテーブルのデータ全体を取得する SQL 文を生成し（テーブル名が「project の場合」、「SELECT * FROM project」）、データベースアクセス(ID:2)で検索を実行する必要があります。これらの処理を実現するため、『ツリーノード格納変数』（１つ）、『リスト格納変数』（１つ）、『比較演算(≧)』（２つ）、『文字格納変数』（３つ）を『データベース検索フレーム』（ID:3）に追加します。コンポーネントのカテゴリは以下のとおりです。

ツリーノード格納変数 ：[処理部品] － [変数] － [ツリーノード格納変数]
リスト格納変数 ：[処理部品] － [変数] － [リスト格納変数]
比較演算(≧) ：[処理部品] － [条件制御] － [比較演算(≧)]
文字格納変数 ：[処理部品] － [変数] － [文字格納変数]

処理の手順は次のとおりです。ツリー(ID:3-7)のノードが選択されたとき、データ選択イベントが発生します。このイベントには選択されたツリーのノードが内包されています。このノードをツリーノード格納変数（ID:3-8）に格納します。ツリーノード格納変数を用いると、ノードのパス（ルートからこのノードに至るまでに通過するノードのリスト）をリストとして取得することができます。このリストの要素数が“２”以上であれば、このリストのインデクス“１”の要素としてデータベース名が取得でき、要素数が“３”以上であればインデクス“２”の要素としてテーブル名も取得できます。これらデータベース名、テーブル名を文字格納変数（ID:3-12,3-13）にそれぞれ格納します。リストの要素数の判定は、２つの比較演算(≧)（ID:3-10,3-11）にリストの要素数を設定して実行します。文字格納変数(ID:3-14)を使用して、取得したテーブル名から SQL 文を作成し、イベント生成（ID:3-6）を介して外部に伝播し、データベースアクセス（ID:2）で検索を実行します。このときのイベント番号は以前に設定したイベントと区別するため“１”とします。このような手順を実行するように、次のようにコンポーネント間を接続します。

表 13 コンポーネント接続関係（ツリーからの SQL 文の生成と伝播）

項目	内容
イベント発生元コンポーネント	ツリー（ID:3-7）
発生イベント	データ選択イベント
イベント番号	定常起動

接続先	◇接続先コンポーネント ツリーノード格納変数（ID:3-8） ◇起動メソッド ツリーノードを設定する（PFObjectTreeNode） ＜引数：ツリーノード＞ 取得方法：『イベント内包』 取得メソッド：“選択データ”

項目	内容
イベント発生元コンポーネント	ツリーノード格納変数（ID:3-8）
発生イベント	データ設定イベント
イベント番号	定常起動
接続先	◇接続先コンポーネント リスト格納変数（ID:3-9） ◇起動メソッド リストを設定する（PFObjectList） ＜引数：リスト＞ 取得方法：『メソッド戻り値』 取得先コンポーネント：ツリーノード格納変数（ID:3-8） 取得メソッド：“パスを取得する（）”

項目	内容
イベント発生元コンポーネント	リスト格納変数（ID:3-9）
発生イベント	データ設定イベント
イベント番号	定常起動
接続先	◇接続先コンポーネント 比較演算（≧）（ID:3-10） ◇起動メソッド 数値に変換して左右オペランドに設定した後で演算を行う（String, String） ＜引数：左オペランド＞ 取得方法：『メソッド戻り値』 取得先コンポーネント：リスト格納変数（ID:3-9） 取得メソッド：“要素数を取得する（）” ＜引数：ツリーノード＞ 取得方法：『固定値』 値：“2”

項目	内容
イベント発生元コンポーネント	比較演算（≧）（ID:3-10）
発生イベント	処理完了イベント
イベント番号	1

接続先	①要素の取得 ◇接続先コンポーネント リスト格納変数（ID:3-9） ◇起動メソッド 要素を指定位置で取得する（int） ＜引数：位置＞ 取得方法：『固定値』 値：“１”
	②文字列設定 ◇接続先コンポーネント 文字列格納変数（ID:3-12） ◇起動メソッド 文字列を設定する（String） ＜引数：文字列＞ 取得方法：『メソッド処理結果』 取得メソッド：“要素を指定位置で取得する（int）”
	③比較演算 ◇接続先コンポーネント 比較演算（≧）（ID:3-11） ◇起動メソッド 数値に変換して左右オペランドに設定した後で演算を行う(String, String) ＜引数：左オペランド＞ 取得方法：『メソッド戻り値』 取得先コンポーネント：リスト格納変数（ID:3-9） 取得メソッド：“要素数を取得する（）” ＜引数：ツリーノード＞ 取得方法：『固定値』 値：“３”

項目	内容
イベント発生元コンポーネント	比較演算（≧）（ID:3-11）
発生イベント	処理完了イベント
イベント番号	1
接続先	①要素の取得 ◇接続先コンポーネント リスト格納変数（ID:3-9） ◇起動メソッド 要素を指定位置で取得する（int） ＜引数：位置＞ 取得方法：『固定値』 値：“２”
	②文字列設定 ◇接続先コンポーネント

	文字列格納変数（ID:3-13） ◇起動メソッド 文字列を設定する（String） ＜引数：文字列＞ 取得方法：『メソッド処理結果』 取得メソッド：“要素を指定位置で取得する（int）”
	③文字列設定（SQL 文生成） ◇接続先コンポーネント 文字列格納変数（ID:3-14） ◇起動メソッド 文字列を設定する（String） ＜引数：文字列＞ 取得方法：『固定値』 値：“ SELECT * FROM ” (注意:最後はスペースです)
	④文字列設定（SQL 文生成） ◇接続先コンポーネント 文字列格納変数（ID:3-14） ◇起動メソッド 指定した文字列と連結して置き換える（String） ＜引数：文字列＞ 取得方法：『メソッド戻り値』 取得先コンポーネント：文字列格納変数（ID:3-13） 取得メソッド：“文字列を取得する（）”
	⑤イベント生成 ◇接続先コンポーネント イベント生成（ID:3-6） ◇起動メソッド データ設定イベントの発生（Object，int） ＜引数：設定データ＞ 取得方法：『メソッド戻り値』 取得先コンポーネント：文字列格納変数（ID:3-14） 取得メソッド：“文字列を取得する（）” ＜引数：イベント番号＞ 取得方法：『固定値』 値：“1”

また、外部から選択されたデータベース名を取得できるように、文字列格納変数(ID:3-12)のメソッドを公開します。

表 14 複合コンポーネントの公開メソッド（その４）

コンポーネント名	メソッド名	公開メソッド名(変更後)
文字列格納変数(ID:3-12)	文字列を取得する（）	データベース名を取得する()

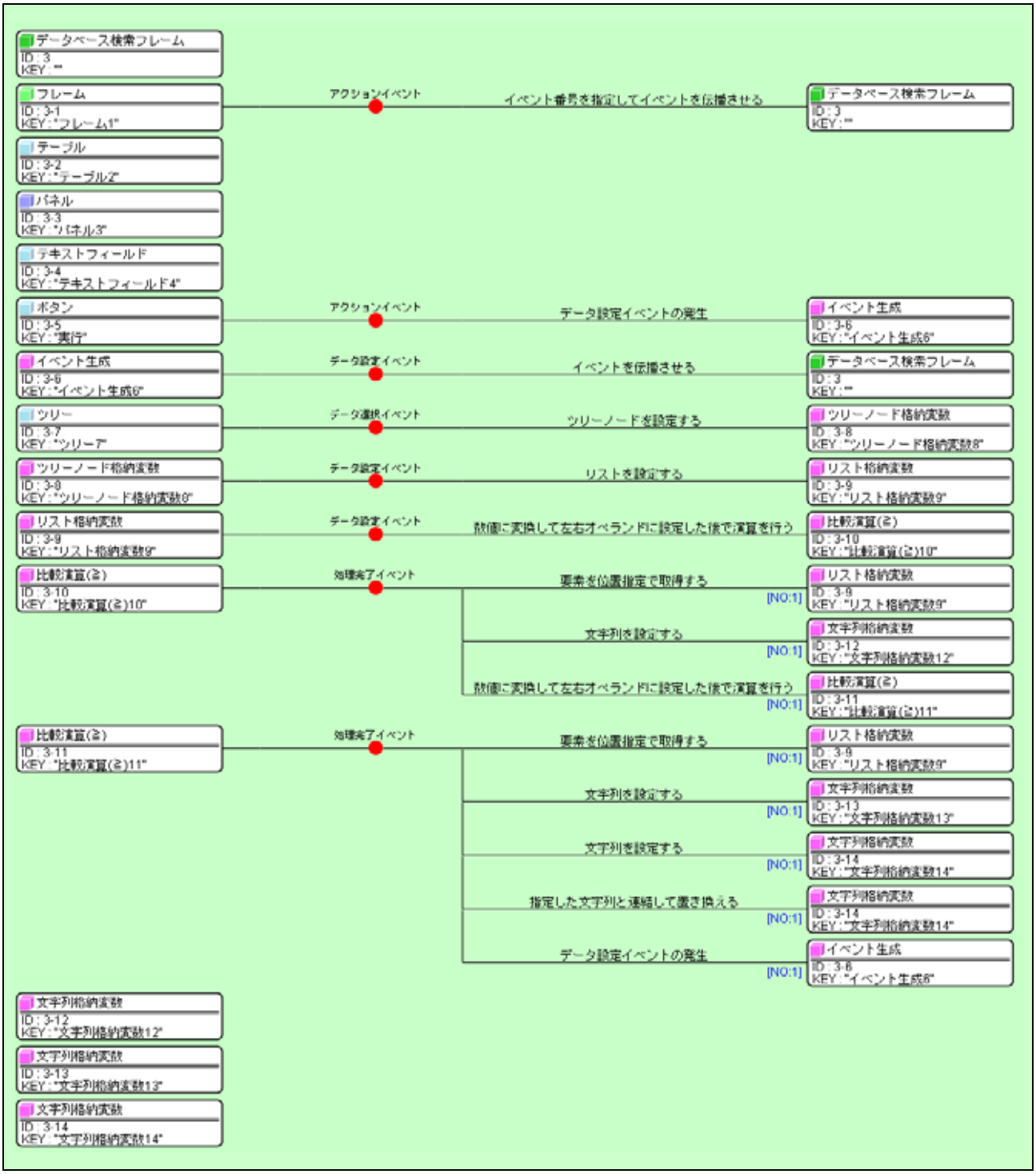

図 11 データベース検索フレームのコンポーネント接続関係（その２）

右上の  を押して、アプリケーションのビルダー画面に移り、データベースアクセス(ID:2)と接続します。はじめに、現在選択されているデータベース名をデータベースアクセス(ID:2)のメソッド『setDatabase(String)』を使用して設定する必要があります。前のステップまではデータベース名が変更することが無かったので設定の必要がありませんでした。コンポーネント間の接続は次のとおりです。

表 15 コンポーネント接続関係（データベースの選択）

項目	内容
イベント発生元コンポーネント	データベース検索フレーム（ID:3）
発生イベント	データ設定イベント
イベント番号	1
接続先	◇接続先コンポーネント データベースアクセス（ID:2） ◇起動メソッド setDatabase(String) ＜引数：データベース名＞ 取得方法：『メソッド戻り値』 取得先コンポーネント：データベース検索フレーム（ID:3） 取得メソッド：“ データベース名を取得する ”

次に、既に設定されているコンポーネント間の接続を変更します。データベース検索フレーム（ID:3）はデータベースアクセス(ID:2)に『SQL 文を実行する』を介して接続されていますが、イベント番号に『1』を追加し、順序を setDatabase（String)による接続の後ろ（下）に移動させます。

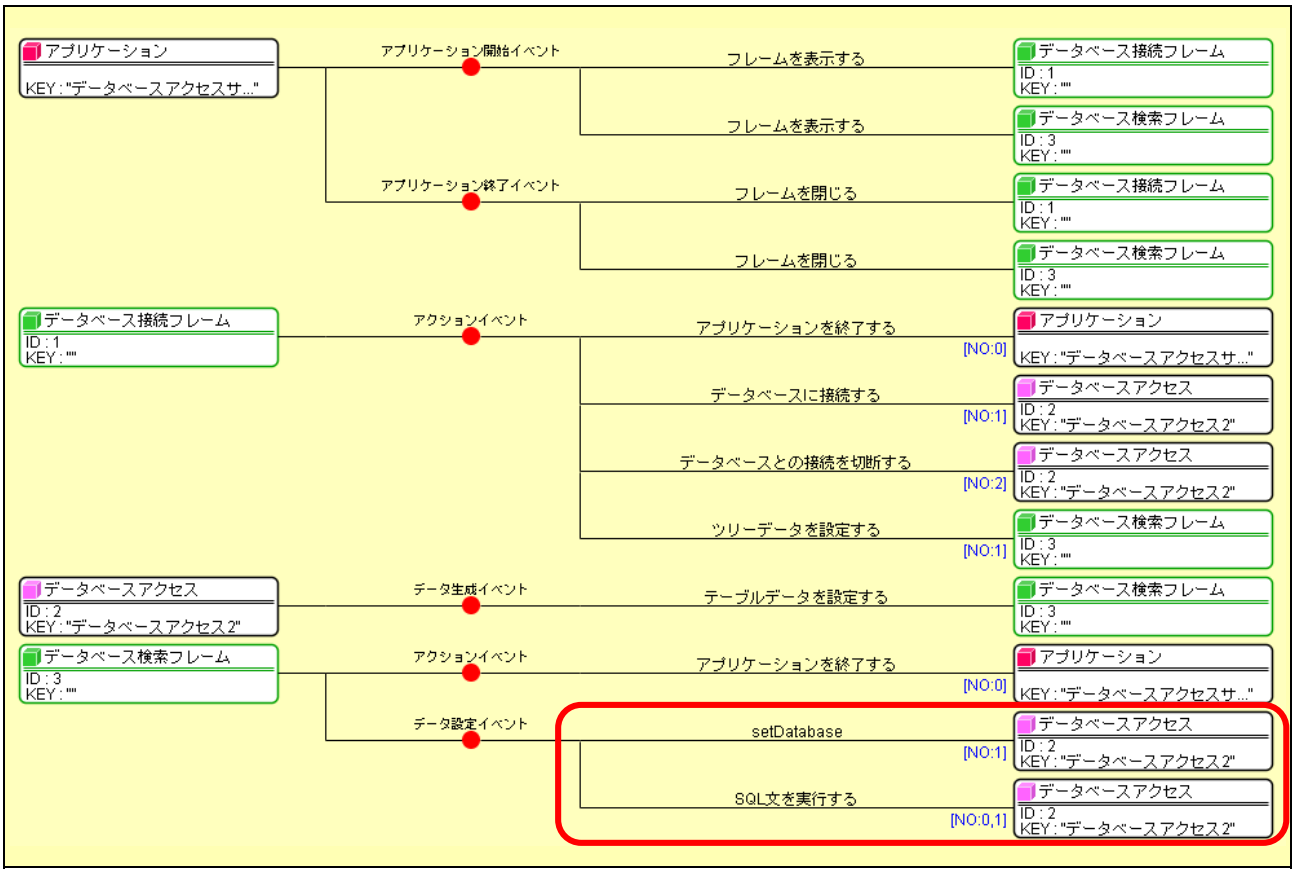

図 12 データベースアクセスアプリケーションのコンポーネント接続関係（その３）

それでは保存し、実行してみましょう。ツリーのノードを選択した際にテーブルの内容が表示されましたか？ご覧のように、データベース検索フレーム(ID:3)のコンポーネント接続はとても複雑になっています。ツリーノード選択から SQL 文生成までを複合コンポーネントにしてみてはいかがでしょうか？

注意：MySQL を用いる場合、ツリーに「mysql」と言うデータベースが表示されます。このデータベース中のテーブルは MySQL が管理しているため、データの操作はお勧めできません。

Step.5 ツリー表示に制限を加える

前のステップまでで機能的にはほぼアプリケーションが完成しました。しかし、このままでは、下図に示すように、データベースのツリー表示において、データベース名やテーブル名等のノードをつかんで、他の場所に移動できてしまいます。

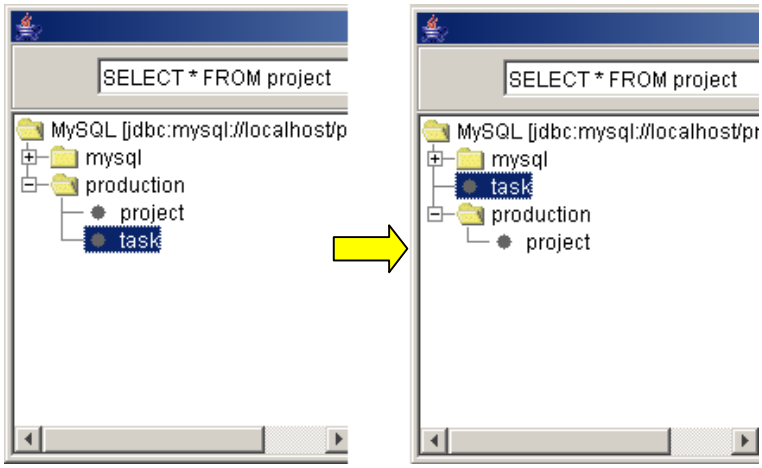

図 13 データベースツリー表示の画面イメージ（その２）

しかし、データベースの実際の構造は変化していないわけですから、このようなツリーの操作に対して制限を設けなければなりません。『ツリー』コンポーネントでは、このようなツリーの情報の変更前に「変更前イベント」として、処理要求イベントが発生する仕様になっており、その接続先に論理値を戻り値として返すメソッドを設定しておけば、操作を制限することができます。すなわち“ true ”が戻り値として返される場合には更新が実行され、“ false ”が返される場合には更新が無効となります。『ツリー』において、制限できる操作項目は以下のとおりで、発生する処理要求イベントに対するイベント番号で区別されます。

表 16 ツリーコンポーネントの制限可能操作とイベント番号

操作	イベント番号
ノードの更新	0
ノードの追加	1
ノードの削除	2
ノードの移動	3
ノードを開く	4
ノードを閉じる	5

それでは実際にツリーの操作を制限する処理を追加してみましょう。データベース検索フレーム（ID:3）の編集画面に移ります。ここで操作の可否を格納するコンポーネントとして『論理値（Boolean）格納変数』を追加します。このコンポーネントのカテゴリは次のとおりです。

論理値(Boolean)格納変数 ：[処理部品] － [変数] － [論理値(Boolean)格納変数]

いま制限したいのは、ツリーノードの移動ですから、次のようにコンポーネント間を接続します。

表 17 コンポーネント接続関係（ツリーのノード操作の制限）（その１）

項目	内容
イベント発生元コンポーネント	ツリー（ID:3-7）
発生イベント	処理要求イベント
イベント番号	3
接続先	◇接続先コンポーネント 論理値(Boolean)格納変数（ID:3-15） ◇起動メソッド 論理値(Boolean)を取得する()

外部からツリーのノード移動の可否を設定可能とするために、次のメソッドを公開します。

表 18 複合コンポーネントの公開メソッド（その５）

コンポーネント名	メソッド名	公開メソッド名(変更後)
論理値(Boolean)格納変数（ID:3-15）	文字列により論理値(Boolean)を設定する(String)	ツリーノードの移動可否を設定する(String)

右上の  を押して、アプリケーションのビルダー画面に移り、アプリケーション起動時にツリーノードの移動不可を指定するため、次のようにコンポーネント間を接続します。

表 19 コンポーネント接続関係（ツリーのノード操作の制限）（その２）

項目	内容
イベント発生元コンポーネント	アプリケーション
発生イベント	アプリケーション開始イベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース検索フレーム（ID:3） ◇起動メソッド ツリーノードの移動可否を設定する(String) ＜引数：論理値＞ 取得方法：『固定値』 値：“ false ”

これでツリー表示の設定が完了しました。それでは保存後に実行してみます。ツリーのノードの移動は制限されましたか？うまくいかない場合にはもう一度、コンポーネント間の接続関係を確かめてください。確認のため、以下に、データベース検索フレーム(ID:3)とアプリケーション全体のコンポーネント接続関係を示します。

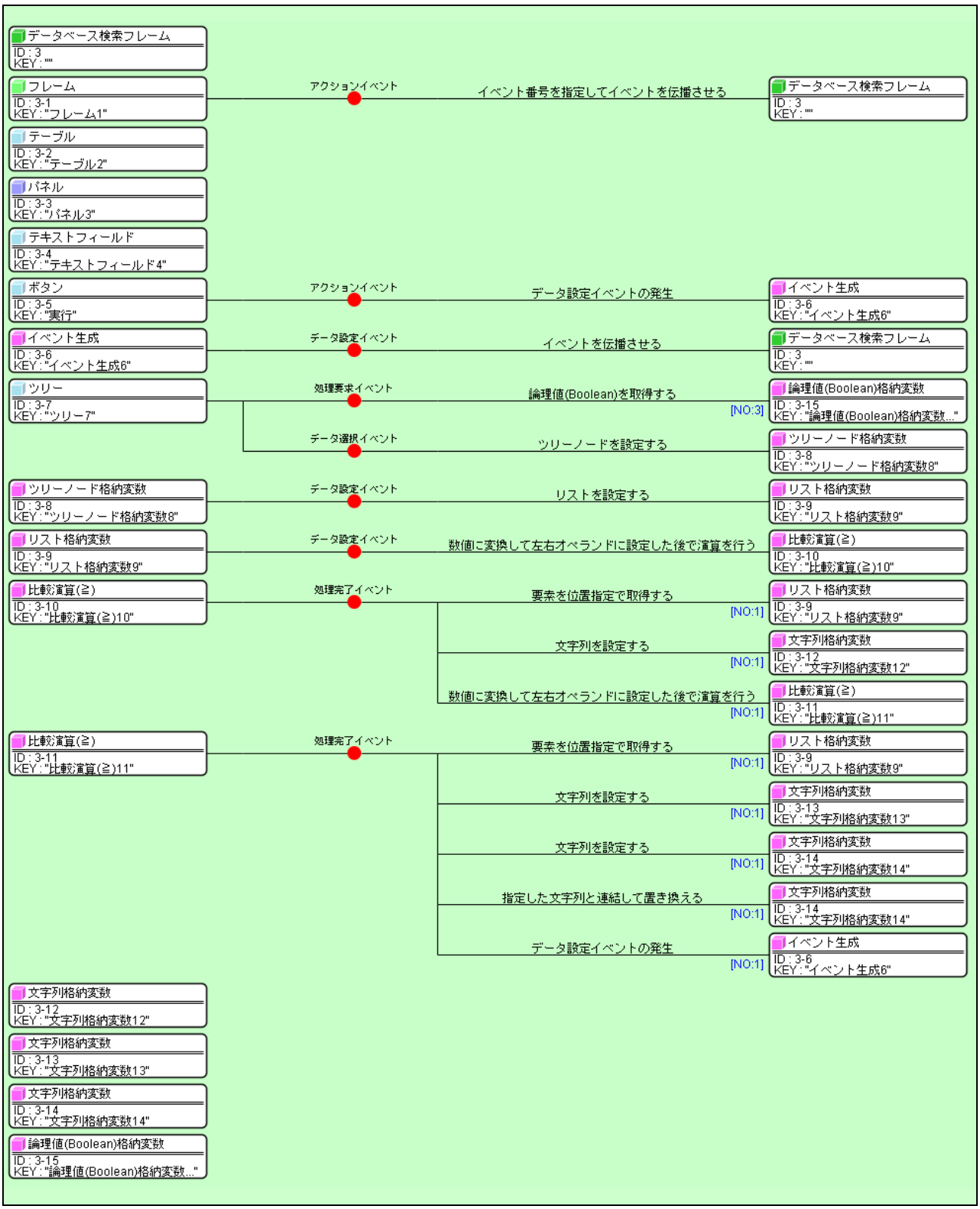

図 14 データベース検索フレームのコンポーネント接続関係（その３）

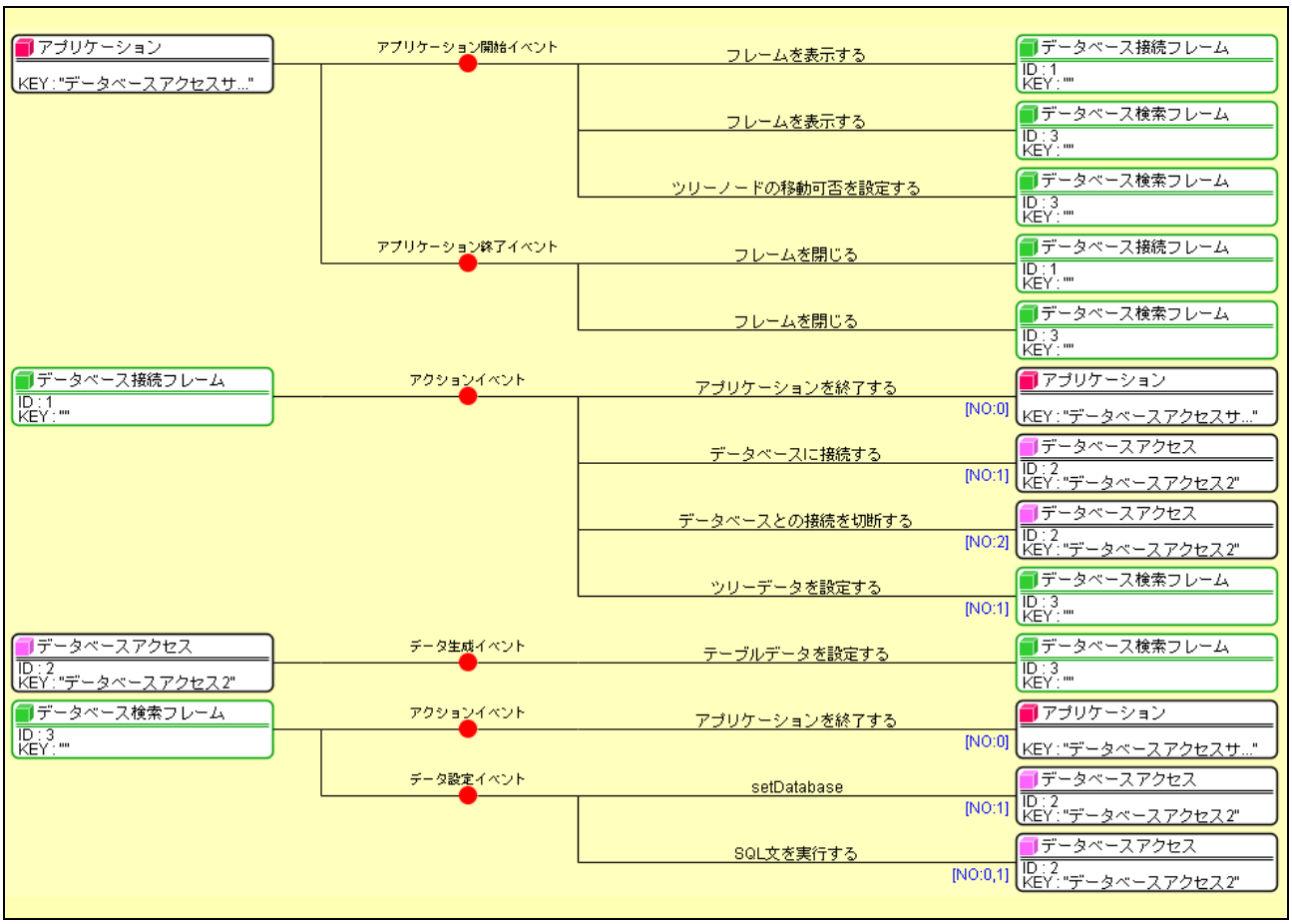

図 15 データベースアクセスアプリケーションのコンポーネント接続関係（その４）

Step.6 テーブル表示に制限を加える

ツリーと同様にテーブルの表示にも制限を加えることが可能です。実際に画面上のテーブルのセルの値を変更しても、データベースのデータには変更が無いわけですから、このようなテーブルのセルの値の更新を制限しましょう。テーブルもツリーと同じように情報の変更前に「変更前イベント」として、処理要求イベントが発生する仕様になっています。その接続先に論理値を戻り値として返すメソッドを設定しておけば、操作を制限することができます。『テーブル』において、制限できる操作項目は以下のとおりで、発生する処理要求イベントに対するイベント番号で区別されます。

表 20 テーブルコンポーネントの制限可能操作とイベント番号

操作	イベント番号
セル更新	0
1 行更新	1
1 列更新	2
単一行追加	10
複数行追加	11
単一列追加	12
複数列追加	13
1 行削除	20
全行削除	21
1 列削除	22
全列削除	23
全行全列削除	24

それでは実際にテーブルの操作を制限する処理を追加します。データベース検索フレーム（ID:3）の編集画面に移ります。ここで操作の可否を格納するコンポーネントとして、ツリーと同様に『論理値（Boolean）格納変数』を追加します。テーブルのセルの更新ですから、次のようにコンポーネント間を接続します。

表 21 コンポーネント接続関係（テーブルのセル更新の制限）（その１）

項目	内容
イベント発生元コンポーネント	テーブル（ID:3-2）
発生イベント	処理要求イベント
イベント番号	0
接続先	◇接続先コンポーネント 論理値(Boolean)格納変数（ID:3-16） ◇起動メソッド 論理値(Boolean)を取得する()

外部からテーブルのセルの更新の可否を設定可能とするために、次のメソッドを公開します。

表 22 複合コンポーネントの公開メソッド（その６）

コンポーネント名	メソッド名	公開メソッド名(変更後)
論理値(Boolean)格納変数（ID:3-16）	文字列により論理値(Boolean)を設定する(String)	テーブルセルの更新可否を設定する(String)

右上の  を押して、アプリケーションのビルダー画面に移り、アプリケーション起動時にテーブルのセルの更新不可を指定するため、次のようにコンポーネント間を接続します。

表 23 コンポーネント接続関係（テーブルのセル更新の制限）（その２）

項目	内容
イベント発生元コンポーネント	アプリケーション
発生イベント	アプリケーション開始イベント
イベント番号	定常起動
接続先	◇接続先コンポーネント データベース検索フレーム（ID:3） ◇起動メソッド テーブルセルの更新可否を設定する(String) ＜引数：論理値＞ 取得方法：『固定値』 値：“ false ”

これでデータベースアクセスアプリケーションが完成しました。それでは保存後、実行してみます。テーブルのセルの更新は制限されましたか？うまくいかない場合には、コンポーネント間の接続関係を確かめてください。確認のため、データベース検索フレーム(ID:3)とアプリケーション全体のコンポーネント接続関係を示します。

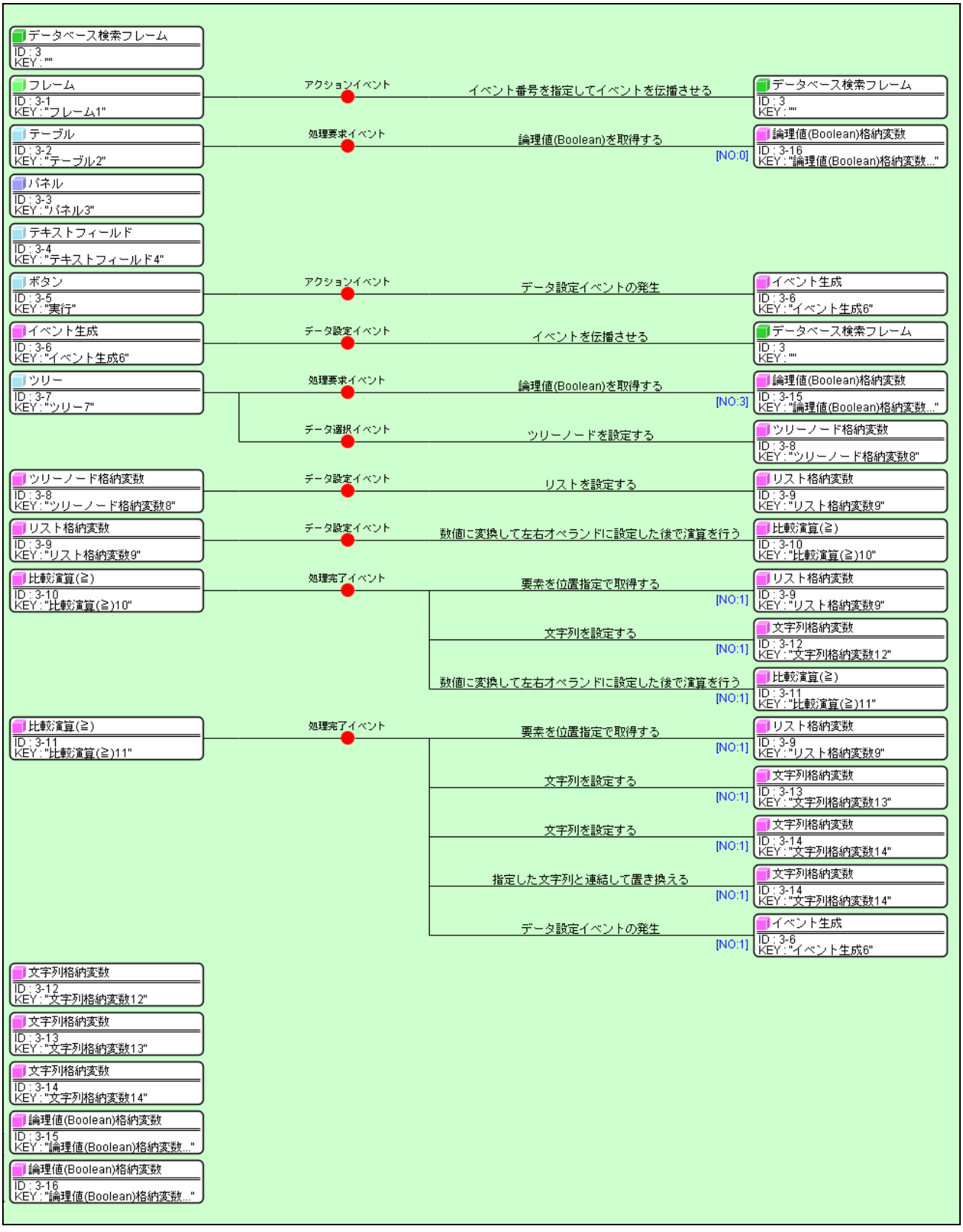

図 16 データベース検索フレームのコンポーネント接続関係（その４）

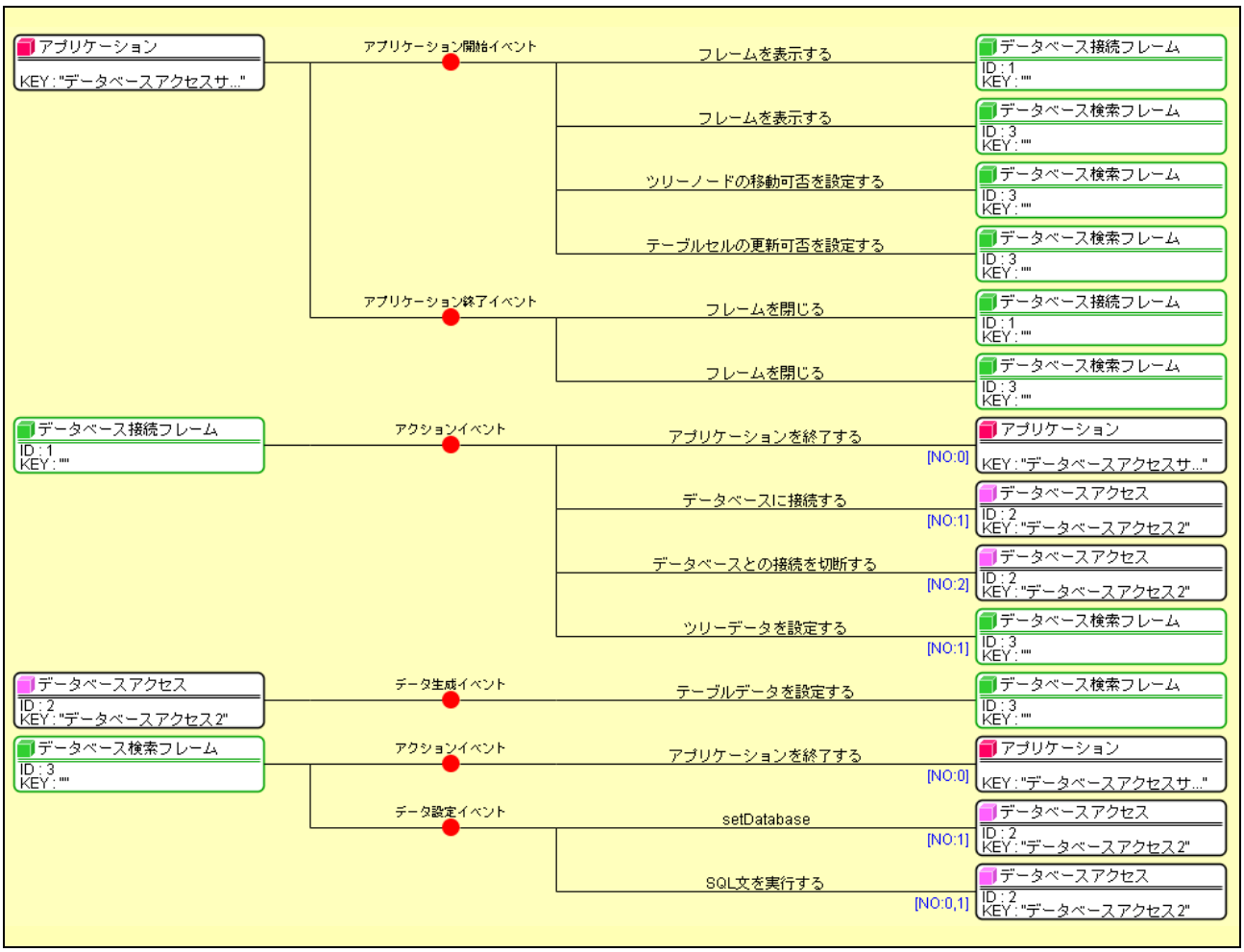

図 17 データベースアクセスアプリケーションのコンポーネント接続関係（その５）