Nowoczesne Systemy Zarządzania
Zeszyt 13 (2018), nr 4 (październik-grudzień)
ISSN 1896-9380, s. 77-88

Instytut Organizacji i Zarządzania
Wydział Cybernetyki
Wojskowa Akademia Techniczna
w Warszawie

Modern Management Systems
Volume 13 (2018), No. 4 (October-December)
ISSN 1896-9380, pp. 77-88

Institute of Organization and Management
Faculty of Cybernetics
Military University of Technology

Aplikacja w języku JAVA do obsługi bazy danych pracowników spółki węglowej

JAVA application to support the coal company employee database

Aurelia Rybak
Politechnika Śląska, Wydział Górnictwa i Geologii,
Katedra Elektrotechniki i Automatyki Przemysłowej

Ewelina Włodarczyk
Politechnika Śląska, Wydział Górnictwa i Geologii,
Katedra Elektrotechniki i Automatyki Przemysłowej

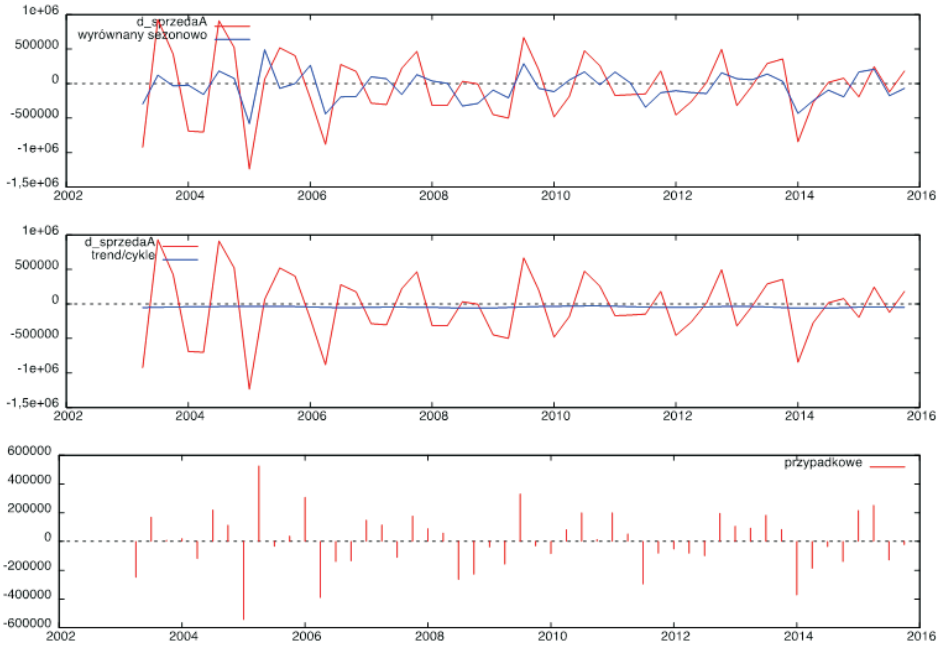
Abstrakt. W artykule zaprezentowano projekt bazy danych, w której przechowywane są informacje na temat pracowników dołowych spółki węglowej, oraz aplikacji w języku Java (JDBC). Aplikacja ma za zadanie ułatwić pracownikom spółki węglowej obsługę bazy danych. Program umożliwia przeglądanie, przeszukiwanie bazy danych, dodawanie i usuwanie rekordów z bazy. Program oraz baza powstały w celu ułatwienia spółkom węglowym elastycznego zarządzania zasobami ludzkimi. To z kolei niezwykle istotne zagadnienie z punktu widzenia kosztów całkowitych produkcji węgla kamiennego. Dodatkowo, biorąc pod uwagę zmiany wydajności pracy związane z sezonowością popytu na węgiel kamienny, rozwiązanie przedstawionego w artykule problemu jest kluczowym aspektem mogącym poprawić kondycję finansową spółek węglowych. Celem pracy było utworzenie narzędzia, którego zastosowanie z całą pewnością umożliwiłoby ograniczenie lub całkowite wyeliminowanie problemów spółek węglowych związanych z przerostem lub niedoborem zatrudnienia w poszczególnych miesiącach roku, a co za tym idzie zbędnych kosztów zatrudnienia. Jest to niezwykle istotne, biorąc pod uwagę, iż koszty te stanowią około 40% kosztu całkowitego produkcji.
Słowa kluczowe: zarządzanie zasobami ludzkimi, baza danych zatrudnienia, elastyczne zatrudnienie.

Abstract. This article presents a database project that stores information about coal company employees and a Java (JDBC) application. The application is intended to support the staff of the coal company during the work with SQL database. The program allows to browse, search the database, add and delete records from the database. The program and the database are designed to facilitate the flexible management of human resources in coal companies. This is a very important issue considering the total cost of hard

coal production and productivity changes associated with the seasonal nature of hard coal demand. The solution presented in the article is a key aspect that could improve the financial condition of coal companies. The purpose of the work was to create a tool, the use of which would certainly limit or eliminate the problems of coal companies related to overgrowth or shortage of employment in particular months of the year. This is extremely important considering that these costs account for approximately 40% of the total production cost.
Keywords: human resource management, employees database, flexible employment.

Wstęp

Spółki węglowe w Polsce borykają się z nadmiarem zgromadzonych czynników produkcji w okresach dekoniunktury oraz niedoborem tych czynników



Rys. 1. Wyniki analizy zjawiska sezonowości szeregu czasowego sprzedaży węgla kamiennego w latach 2003-2016

Źródło: opracowanie własne

w momentach wzmożonego popytu na węgiel kamienny. Przyczyną takiego stanu rzeczy jest zjawisko sezonowości zarówno sprzedaży jak i wydobycia węgla kamiennego (rys. 1).

W związku z tym, iż 40% kosztów całkowitych produkcji węgla kamiennego stanowią koszty związane z pracą żywą, doskonałym rozwiązaniem wyżej wymienionego

problemu było utworzenie bazy danych zawierającej informacje na temat pracowników dołowych na poziomie całej spółki węglowej. Dzięki temu istnieje możliwość skierowania pracowników do kopalni, w której w danym momencie pojawia się konieczność podniesienia wolumenu wydobywanego węgla, co równocześnie umożliwiałoby wyeliminowanie tak zwanego marnotrawstwa w tych kopalniach, gdzie poziom produkcji spada. Wykorzystano w tym celu system zarządzania bazą danych MySQL. Jest to jeden z najpopularniejszych systemów, łatwy w obsłudze i funkcjonalny, stanowiący wolne oprogramowanie chętnie wykorzystywane przez deweloperów (Ullman, 2006, s. 10-11; Sheldon, Moes, 2005, s. 31). Ponieważ baza danych została utworzona w języku SQL, jej obsługa wymagałaby od użytkownika w spółce węglowej znajomości Strukturalnego Języka Zapytań (SQL). Przedmiotem pracy było zatem utworzenie aplikacji w języku Java do obsługi bazy danych. Aplikacja umożliwia nawiązanie połączenia z bazą, a także wyświetlanie jej zasobów, dodawanie i usuwanie rekordów, przeszukiwanie i filtrowanie informacji na temat pracowników (Na G. et al., 2011, s. 2051-2058; Thariqa, Sitanggang, 2015, s. 277-284; Vujic et al. 2000, s. 9-25).

JDBC to standard, który definiuje metody dostępu do baz danych (Grochola, 2000, s. 14). Udostępnia interfejs, który umożliwia nawiązanie połączenia niezależnej aplikacji napisanej w języku Java z relacyjną bazą danych (Iyer et al., 2010, s. 29; Łacheciński, 2015, s. 69-84). Autorzy skorzystali ze standardu tworząc własną aplikację, ponieważ jest ona dzięki temu niezależna od maszyny bazodanowej oraz systemu operacyjnego. Znajomość Javy umożliwia szybkie tworzenie aplikacji bazodanowych o wysokiej funkcjonalności. Baza danych będzie rozwijana i modyfikowana zgodnie z potrzebami spółek węglowych. Oznacza to, iż w przyszłości niezbędne będą dodatkowe funkcje i narzędzia, a JDBC z całą pewnością zapewnia do nich dostęp. Gotowe systemy zarządzania bazami danych, do których autorzy mieli dostęp posiadały wady, które skłoniły autorów do utworzenia aplikacji EMT, np. ograniczenia wielkości bazy danych, autoryzacja dostępu, czy też problemy z wielodostępem.

1. Materiały i metody

Na potrzeby prowadzonych badań utworzono bazę danych w języku SQL oraz aplikację w języku Java. Aplikacja umożliwia obsługę bazy danych zawierającej informacje na temat pracowników spółki węglowej.

Język programowania Java charakteryzuje się:

- prostą, jasną koncepcją,
- obiektowością,
- przenośnością,
- bezpieczeństwem,

- wysoką wydajnością,
- dynamiką,
- wielowątkowością,
- rozproszeniem (Schildt, 2005, s. 33-36; Eckel, 2006, s. 62).

SQL to Strukturalny Język Zapytań, który umożliwia:

- tworzenie bazy danych,
- określanie struktury bazy,
- formułowanie zapytań, a na ich podstawie uzyskiwanie dostępu do zawartych w bazie danych,
- kontrolę bezpieczeństwa (Reese, 2000, s. 44; Czapla, 2015, s. 65-66; Gruber, 2000, s. 19-20).

Zapytanie w języku SQL jest złożone z:

- wyrażenia – to polecenie, które ma zostać wykonane,
- klauzuli – wyznaczającej ograniczenia dla wyrażenia,
- warunków – zdefiniowanych przez ograniczenia (Wilton, Colby, 2005, s.12).

Baza danych zawiera takie informacje (argumenty) jak:

- liczba porządkowa przypisana do pracownika,
- nazwisko pracownika,
- numer identyfikacyjny kopalni, na której jest on zatrudniony,
- status pracownika.

Najistotniejszą informacją zawartą w bazie jest tak zwany status pracownika. Określa on, czy w danym momencie pracownik wykonuje określone zadania związane bezpośrednio z kluczową działalnością danej kopalni, tj. wydobyciem, czy też istnieje możliwość jego oddelegowania do prac związanym z wydobyciem na innej kopalni wchodzącej w skład spółki. Oddelegowanie to przyjmuje formę outsourcingu kapitałowego.

Aplikacja EMT – Employee Managament Tool składa się z okien:

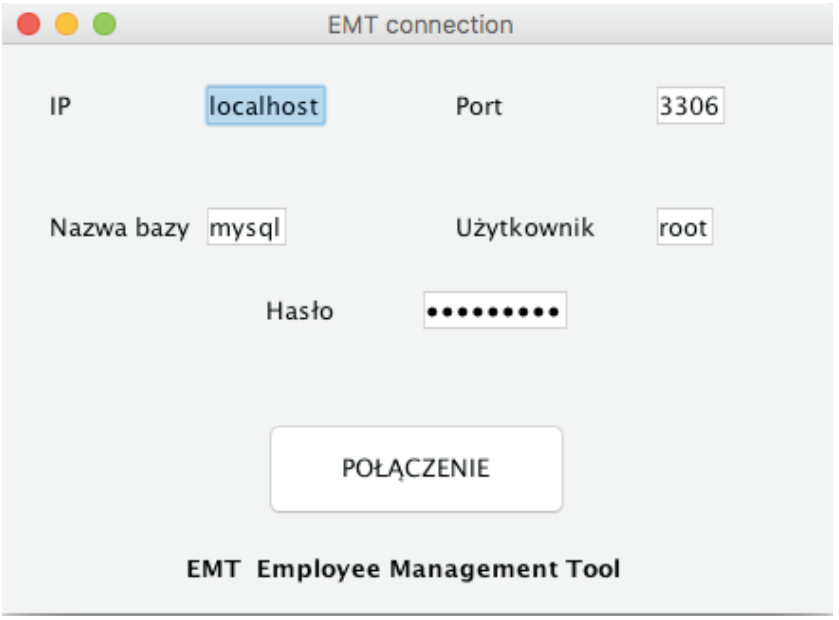
- połącznia,
- głównego,
- błędu połączenia,
- nowy pracownik,
- zapytanie do bazy danych.

Hierarchię okien aplikacji przedstawiono na rysunku 2.

W celu nawiązania połączenia wymagane jest podanie następujących parametrów:

- IP – localhost,
- port domyślny – 3306,
- nazwa bazy danych – mysql,
- użytkownik – root,
- hasło.

Rys. 2. Schemat aplikacji EMT
Źródło: opracowanie własne



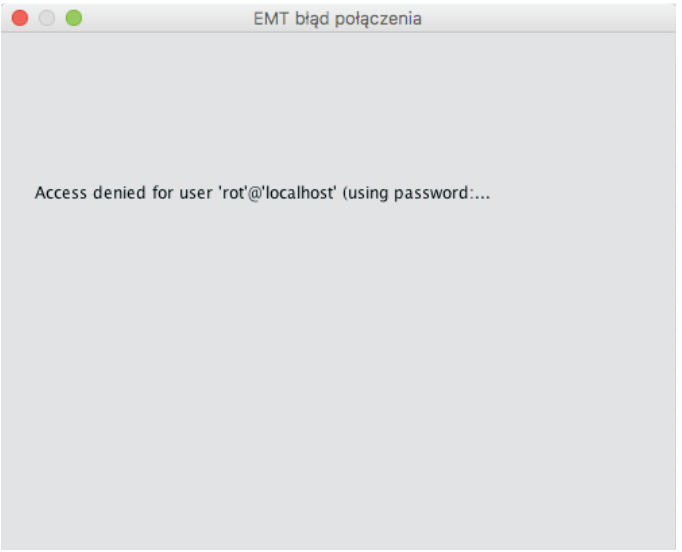
Rys. 3. Okno połączenia z bazą danych aplikacji EMT
Źródło: opracowanie własne

Wartości parametrów należy wprowadzić w polu tekstowym (rysunek 3). Po uruchomieniu zdarzenia ActionEvent poprzez naciśnięcie przycisku *Połączenie* aplikacja łączy się z bazą danych MySQL (rysunek 3).

W przypadku błędnego podania któregoś z parametrów połączenia otwarte zostanie okno dialogowe (JDialog) błędu połączenia. Użytkownik zostaje powiadomiony o charakterze, przyczynie błędu. Jest to okno modalne, które blokuje przetwarzanie zdarzeń przez pozostałe okna aplikacji. Na rysunku 4 zaprezentowano okno błędu z komunikatem oznajmiającym podanie niewłaściwej nazwy użytkownika.

Po nawiązaniu połączenia otwarte zostaje okno główne programu przedstawione na rysunku 5. Użytkownik ma możliwość wykonania następujących operacji:

- pobrania zawartości bazy danych,
- dodania do bazy nowego pracownika,
- usunięcia wybranego pracownika,
- utworzenia zapytania do bazy danych.



Rys. 4. Okno dialogowe błędu połączenia programu EMT

Źródło: opracowanie własne

Rys. 5. Okno główne programu EMT

Źródło: opracowanie własne

2. Pobranie zawartości bazy danych

Pobranie danych z bazy uruchamiane jest naciśnięciem przycisku *Pobierz*. Przyciskowi została przypisana metoda ActionPerformed. Jej wywołanie umożliwia przesłanie do bazy danych następującego zapytania:

zapytanie= „SELECT lp, nazwisko, id, status FROM pracownik". Pętle for z kolei umożliwiają wyświetlenie rekordów dla wszystkich pól bazy. Wyniki zapytania: wyświetlane są w tabeli (JTable) o polach zgodnych z polami tabeli bazy danych MySQL, są to pola:

- zaznaczenie (CheckBox powiązany z przyciskiem *Usuń*),
- lp. (typu int),
- nazwisko (typu String),
- id kopalni, na której pracownik jest zatrudniony (typu int),
- status (typu int) określa dostępność pracownika. Zakłada się, że pracownik jest dostępny (status 1), jeżeli przewidywany okres jego bezczynności na kopalni macierzystej to przynajmniej dwa tygodnie. Jeżeli status pracownika to 0 nie jest możliwe jego przeniesienie na inną kopalnię.

3. Dodawanie do bazy nowego pracownika

W celu dodania do bazy danych nowego rekordu należy wykorzystać obiekt Button o nazwie *Nowy pracownik*. Umożliwia to wywołanie okna *Nowy pracownik* (rysunek 6). Metoda przypisana do przycisku *Dodaj* powoduje wykonanie zapytania:

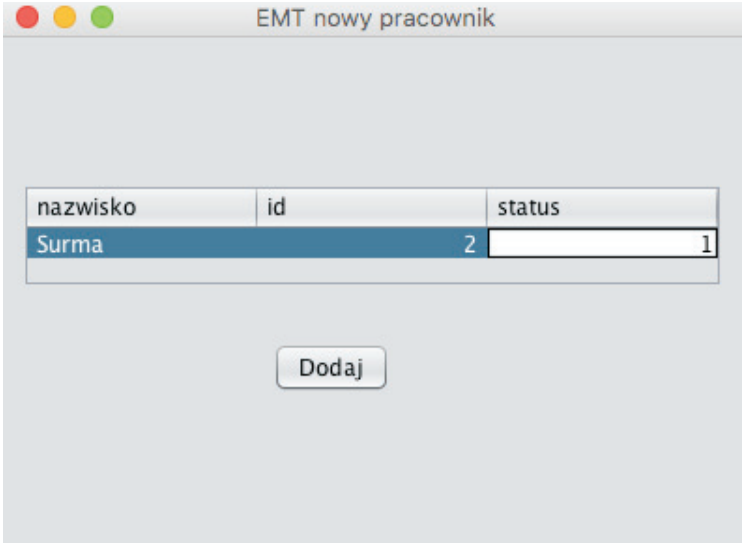
zapytanie= „INSERT INTO pracownik(lp, nazwisko, id, status) values (...)".

Dodatkowo wprowadzono zabezpieczenie przed dodaniem do bazy danych pustego rekordu. W przypadku niewprowadzenia do tabeli (*NowyTabla*) odpowiedniej wartości zmiennej pojawia się komunikat na temat popełnionego błędu, np. „Podaj nazwisko" (rysunek 7). Liczba porządkowa poszczególnych rekordów jest automatycznie dodawana do tabeli.

```
private void BdodajActionPerformed(java.awt.event.ActionEvent evt) {

    Text_bledy_nowyp.setText("Błąd!\n");
    if(NowyTabela.getModel().getValueAt(0, 0)==null){
    sprawdzenie=false;
    Text_bledy_nowyp.append("Podaj nazwisko!\n");
    }
    if(NowyTabela.getModel().getValueAt(0, 1)==null){
   sprawdzenie=false;
    Text_bledy_nowyp.append("Podaj id kopalni!\n");
    }if(NowyTabela.getModel().getValueAt(0, 2)==null){
    sprawdzenie=false;
    Text_bledy_nowyp.append("Podaj status pracownika: 0- niedostępny, 1-dostępny \n");
    }
    if(sprawdzenie){
    try {

   zapytanie= "SELECT MAX(lp) lp FROM pracownik";
    ResultSet rs2 = db.statement.executeQuery(zapytanie);
    rs2.next();
      int pomocnicza=0;
      pomocnicza=Integer.parseInt(rs2.getString("lp"))+1;

 zapytanie= " INSERT INTO pracownik(lp,nazwisko, id, status) values ("
        +pomocnicza+",'"

         + NowyTabela.getModel().getValueAt(0, 0)+"', '"
           + NowyTabela.getModel().getValueAt(0, 1)+"', "
          + NowyTabela.getModel().getValueAt(0, 2)+")";
  db.statement.executeUpdate(zapytanie);

  } catch (SQLException e) {
      e.printStackTrace();
   }
```

Rys. 6. Dodawanie do bazy danych nowego pracownika

Źródło: opracowanie własne



Rys. 7. Okno programu EMT umożliwiające dodanie do bazy nowego pracownika

Źródło: opracowanie własne

4. Usuwanie wybranego pracownika z bazy

Możliwe jest również usunięcie z bazy zbędnego pracownika, bądź błędnie wprowadzonego rekordu. Operacja usuwania pracownika z bazy danych zapoczątkowana zostaje przyciskiem *Usuń zaznaczone*. Aby jednak możliwe było określenie, który z pracowników powinien zostać usunięty w tabeli umieszczono pole zawierające obiekt CheckBox.

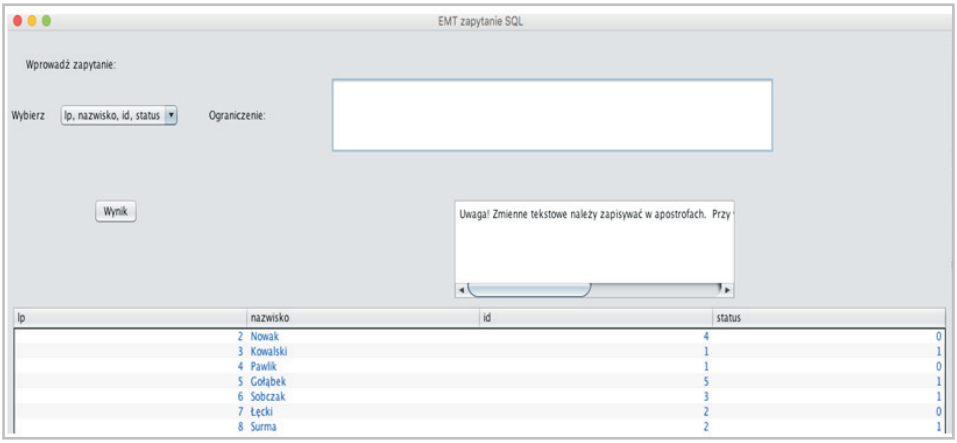
Po wyborze pracownika i kliknięciu przycisku *Usuń* następuje uruchomienie akcji *UsuwanieActionPerformed*. W ciele metody *ActionPerformed* umieszczono kod, którego wykonanie powoduje:

wysłanie zapytania: „DELETE FROM pracownik WHERE lp=”+usuwanie;

5. Zapytanie do bazy danych

W celu ułatwienia przeszukiwania bazy danych pracownikom spółki węglowej utworzono okno Zapytanie SQL. Okno składa się z takich elementów jak:

- pole wyboru kolumn (JComboBox),
- pole tekstowe dla ograniczenia (JTextField),
- tabeli (JTable),
- przycisk Wynik (JButton),
- okno przewijanego (JScrollPane) (rysunek 8).



Rys. 8. Okno zapytanie SQL programu EMT

Źródło: opracowanie własne

Użytkownik ma możliwość zdefiniowania wyniku, czyli wyboru listy kolumn która ma się znaleźć w tabeli zawierającej rezultaty zapytania. Ponadto istnieje możliwość wprowadzenia dodatkowego warunku, który mają spełniać zwrócone rekordy. Stanowi to ułatwienie dla użytkownika niezaznajomionego z językiem SQL. Struktura polecenia SELECT:

SELECT
FROM
WHERE

ukryta została w kodzie instrukcji warunkowej if-else. Jeżeli na przykład użytkownik wskaże, iż z bazy danych mają zostać pobrane dane na temat statusu pracownika oraz liczba porządkowa uruchomiony zostanie wysłane do bazy danych zapytanie:

zapytanie= „Select lp, status from pracownik where"+poletekstowe.getText();

Ograniczenia pobierane są z pola tekstowego i umieszczane w klauzuli WHERE. W przypadku, gdy użytkownik nie poda warunku ograniczającego zostaje powiadomiony o konieczności jego wprowadzenia komunikatem umieszczonym w polu tekstowym warunku.

Dodatkowe instrukcje na temat właściwego zapisu warunku ograniczającego umieszczono w oknie przewijanym.

Podsumowanie

Specyfika przemysłu wydobywczego węgla kamiennego w Polsce, sezonowość sprzedaży i produkcji węgla niekorzystnie wpływają na poziom kosztów całkowitych przedsiębiorstwa. Przerost kosztów wiąże się z ograniczonym zyskiem operacyjnym. Ogromny udział w kosztach całkowitych mają koszty stałe produkcji, niezależne od poziomu wydobycia. Do kosztów stałych zaliczane są w polskim górnictwie koszty zatrudnienia. Produktywność marginalna pracy żywej jest natomiast około trzykrotnie niższa aniżeli pracy uprzedmiotowionej. Aby wyeliminować tak zwane marnotrawstwo w przedsiębiorstwach górniczych konieczne jest uelastycznienie zatrudnienia. To z kolei może zostać osiągnięte dzięki zastosowaniu outsourcingu kapitałowego (wewnętrznego). Narzędziem ułatwiających to niezwykle skomplikowane zadanie jest Aplikacja EMT (Employee Managament Tool). Celem pracy było utworzenie aplikacji do obsługi bazy danych MySQL zawierającej informacje na temat pracowników dołowych spółki węglowej. Kluczową informacją zawartą w bazie jest tak zwany status pracownika. Aplikacja działa w oparciu o interfejs programowania JDBC. Umożliwia połączenie z bazą danych, wprowadzanie zmian w bazie danych, dodawanie nowych pracowników, usuwanie zbędnych rekordów, filtrowanie bazy danych za pomocą odpowiednich zapytań. Graficzny interfejs użytkownika EMT ułatwi pracę z bazą danych pracownikom w spółce węglowej niezaznajomionym z Strukturalnym Językiem Zapytań. Kolejnym krokiem w pracy

nad doskonaleniem aplikacji będzie jej dostosowanie do wymogów Polskiej Grupy Górniczej. Na potrzeby aplikacji pozyskane zostaną dane rzeczywiste ze spółki węglowej. Aplikacja zostanie zmodyfikowana zgodnie z sugestiami pracowników spółki. Dzięki temu możliwe będzie przetestowanie jej funkcjonalności w zderzeniu z danymi empirycznymi.

BIBLIOGRAFIA

[1] Czapla K., 2015, *Bazy danych. Podstawy projektowania i języka SQL*, Wyd. Helion, Gliwice.

[2] Eckel B., 2006, *Thinking in Java*, Wyd. Helion, Gliwice.

[3] Grochola M., 2000, *Java aplikacje bazodanowe*, Wyd. Helion, Gliwice.

[4] Gruber M., 2000, *SQL*, Wydanie II, Wyd. Helion, Gliwice.

[5] Iyer V., Perry E., Wright B., Pfaeffle T., 2010, *Oracle Database JDBC Developer's Guide and Reference*, Oracle Corporation.

[6] Łachociński S., 2015, *Interfejsy dostępu do baz danych – przegląd technologii Borland, Embarcadero, Sun, Oracle*, Informatyka ekonomiczna, Vol. 3(37).

[7] Na G., Longzhe J., Peng W., Ling L., 2011, The Study and Application of Safety Information Management System of the Coal Mines Engineering, Procedia Engineering, Vol. 26.

[8] Reese G., 2000, *Database Programming with JDBC and Java*, O'Reilly Media, Sebastopol.

[9] Schildt H., 2014, *Java kompendium programisty*, Wyd. Helion, Gliwice.

[10] Sheldon R., Moes G., 2005, *Beginning MySQL*, Wiley Publishing, Indianapolis.

[11] Thariqa P., Sitanggang I.S., 2015, *Spatial online analytical processing for hotspots distribution based on socio-economic factors in Riau Province Indonesia,* Procedia Environmental Sciences, Vol. 24.

[12] Ullman L., 2006, *MySQL, Second Edition: Visual QuickStart Guide*, Peachpit Press, Berkeley.

[13] Vujic S., Miljanovic I., Petrovski A., 2000, *A Concept of The Establishing the Computer Supported Information Management System at the"DRMNO" Open Pit Mine, Coal Basin Kostolac*, Journal of Mining Science, Vol. 44.

[14] Wilton P., Colby J., 2005, *SQL od podstaw*, Wyd. Helion, Gliwice.