**MINISTERUL EDUCATIEI AL REPUBLICII MOLDOVA**

**UNIVERSITATEA DE STAT “ALECU RUSSO” DIN BALTI**

**FACULTATEA DE STIINTE REALE, ECONOMICE SI ALE MEDIULUI**

**CATEDRA DE INFORMATICA STIINTE EXACTE**

**Lucrul cu baze de date în Haskell**

**Referat la disciplina: “Programarea funcțională (Python, Scala, etc)”  
Profesor: Skutnitki Olesea**

**Autori:** Cebonas Igor,   
Mitriniuc Eric. **Grupa:** IS21Z.

**CUPRINS**

[**CE ESTE HASKELL** 3](#_Toc26261825)

[**BAZA DE DATE** 4](#_Toc26261826)

[**a)** **Introducere** 4](#_Toc26261827)

[**b)** **Установка** 5](#_Toc26261828)

[**ОБЩИЙ РАБОЧИЙ ПРОЦЕСС** 6](#_Toc26261829)

[**a)** **Подключить и отключить** 6](#_Toc26261830)

[**b)** **Запущенные Запросы** 6](#_Toc26261831)

[**Tranzacție și Rularea declarațiilor SQL** 7](#_Toc26261832)

[**BIBLIOGRAFIE** 8](#_Toc26261833)

# **CE ESTE HASKELL**

**Haskell** este un *limbaj de programare*, un limbaj în care oamenii pot exprima modul în care trebuie să se comporte calculatoarele.

*Haskell*este un limbaj de *[programare funcțional](https://en.wikibooks.org/wiki/Computer_programming/Functional_programming" \o "Programare computer / Programare funcțională)* . Este lămurit în câteva etape:

1. Este pur – funcția dacă este invocată folosind aceleași argumente va întoarce aceleași rezultate de fiecare data.
2. Este leneș – evaluările sunt efectuate doar în momentele când este nevoie.
3. Este de tip modern – tipurile de clasă și tipurile de date algebrice generalizate sunt de ultimă generație.

*Haskell* este ușor de utilizat deoarece folosirea funcțiilor pure face ca codul să fie mult mai ușor de motivat, iar sistemul avansat gaseste ușor greșelele.

*Haskell* se împarte în cîteva piese, cum ar fi:

1. Pistă pentru începători:
   1. Bazele Haskell;
   2. Haskell Elementar;
   3. Haskell Interimar;
   4. Monade.
2. Pistă avansată:
   1. Haskell Avasat;
   2. Tipuri de date;
   3. Teorie;
   4. Haskell Performant.

Noi vom începe cu Pista pentru începători și anume cu Bazele Haskell.

# **BAZA DE DATE**

## **Introducere**

Cel mai popular modul de bază al bazei de date Haskell este [HDBC](https://github.com/hdbc/hdbc/wiki). HDBC oferă un strat de abstractizare între programele Haskell și bazele de date relaționale SQL. Acest lucru vă permite să scrieți codu=l bazei de date o singură dată, în Haskell, și să funcționeze cu o serie de baze de date SQL backend.

* **HDBC** - Haskell Connectness Database.
* **SQL** -Structured Query Language.

HDBC este modelat slab pe interfața DBI a lui Perl, deci a fost influențat și de DB-API v2 Python, JDBC în Java și HSQL în Haskell.

* **DBI -** Database independent interface for Perl.
* **DB-API v2 Python –** Python Database API Specification.
* **JDBC -** Java DataBase Connectivity.
* **HSQL –** Hyper Structured Query Language Database.

Aceste drivere backend Haskell Connectness Database (HDBC) există: PostSQL, SQLite și ODBC și ele sunt disponibile pentru Windows și Unix, Linux și Mac).

* **PostSQL –** este un system [de](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%A1%D0%A3%D0%91%D0%94)[gestionare a bazelor de date](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)[relațional cu obiecte](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%A1%D0%A3%D0%91%D0%94).
* **SQLite –** este un system compact de gestionare a bazelor de date încorporat.
* **ODBC –** este o interfață software pentru accesarea bazelor de date dezvoltate de Microsoft.
* **DBMS** – este un system de gestionare a bazelor de date încorporat.

**MySQL -** este un [open-source](https://en.wikipedia.org/wiki/Open-source_software)[sistem de management al bazelor de date relaționale](https://en.wikipedia.org/wiki/Relational_database_management_system)  și este cea mai populară bază de date open-source și există două drivere pentru MySQL: HDBC-mysql care este native și HDBC-odbc (un backend de bază de date ODBC pentru HDBC).

Avantajul utilizării ODBC este că sintaxa instrucțiunii SQL este izolată de diferite tipuri de motoare de bază de date. Acest lucru duce la creșterea portabilitații aplicației în cazul în care trebuie să vă mutați de la o bază de date la alta. Același argument pentru ODBC se aplică și altor baze de date comerciale.

## **Установка**

1. **PostgreSQL или SQLite –** начинаем с сборкой драйверов SQLite для Windows (c GHC):
2. **Длинная версия**
3. **Укорочнная версия**

По сколько она самая удобная, мы будем с ней работать и начинаем делать следующие шаги: поместите «sqlite3.h» из sqlite-source-3 \* .zip в *`ghc - -print-libdir`/include.* Потом поместите «sqlite3.dll» из sqlitedll-3 \* .zip в ghc - -print –libdir и в %windir%\system32. И в конце продолжите сборку HDBC и HDBC -Sqlite.

1. **MySQL -** cобственная библиотека ODBC-mysql требует наличия клиентской библиотеки C MySQL. Возможно, вам придется [обернуть доступ](http://www.serpentine.com/blog/2010/09/04/dealing-with-fragile-c-libraries-e-g-mysql-from-haskell/) к [вашей базе данных,](http://www.serpentine.com/blog/2010/09/04/dealing-with-fragile-c-libraries-e-g-mysql-from-haskell/) чтобы предотвратить ошибки во время выполнения.
2. **ODBC / MySQL -** заставить HDBC работать с MySQL через ODBC довольно сложно, особенно если у вас нет привилегий root (привилегии суперпользователя).

* Если ваша платформа еще не предоставляет библиотеку, установите Unix-ODBC.
* Установите MySQL-ODBC Connector.
* Установите модуль Database.HDBC.
* Установите модуль Database.HDBC.ODBC.
* Добавьте драйвер mysql в файл odbcinst.ini (в папке $ ODBC\_HOME / etc /) и свой источник данных в $ HOME / .odbc.ini.
* Создать тестовую программу.

Так как драйвер ODBC устанавливается с использованием разделяемой библиотеки по умолчанию, вам потребуется env:

export LD\_LIBRARY\_PATH=$ODBC\_HOME/lib

Если вам не нравится добавлять дополнительные переменные env, попробуйте скомпилировать ODBC с включенной опцией статической библиотеки. Потом напишите простую тестовую программу, которая подключается к базе данных и печатать имена всех ваших таблиц, например:

module Main where

import Database.HDBC.ODBC  
import Database.HDBC  
main =

do c < - connectODBC “DSN = PSPDSN”

XS <- getTables c  
putStr $ “tables” ++ (foldr jn “.” xs) ++ “\n”

Whiere jn a b = a++ “ ”+

# **ОБЩИЙ РАБОЧИЙ ПРОЦЕСС**

## **Подключить и отключить**

Первым шагом любой операции с базой данных является подключение к целевой базе данных. Это делается через специфичный для драйвера API подключения, который имеет тип:

Строка **->**  IO Connection

При наличии строки подключения API подключения вернется Connectionи поместит вас в монаду ввода-вывода.

Несмотря на то, что большинство программ собирает мусор в ваших соединениях, когда они находятся вне области действия или когда программа заканчивается, рекомендуется явно отключиться от базы данных.

conn-> Disconnect

## **Запущенные Запросы**

Выполнение запроса обычно включает в себя следующие шаги:

* Подготовить заявление
* Выполнить оператор с переменными связывания
* Получить набор результатов (если есть)
* Завершите утверждение

HDBC (Haskell Connectness Database) предоставляет два способа связывания переменных и возврата набора результатов: [ SqlValue ]и [ Maybe String ].Вам нужно использовать функции с префиксом “**s”** при использовании [ Maybe String ]вместо [ SqlValue ]. [ SqlValue ]позволяет использовать строго типизированные данные, если безопасность типа очень важна в вашем приложении; в противном случае [ Maybe String ]это более удобно при работе с большим количеством запросов к базе данных. При использовании [ Maybe String ]вы предполагаете, что драйвер базы данных будет выполнять автоматическое преобразование данных. Имейте в виду, что для этого удобства есть цена за производительность.

# **Tranzacție și Rularea declarațiilor SQL**

Tranzacția cu baza de date este controlată de commitși rollback. Cu toate acestea, fiți conștienți că unele baze de date (cum ar fi mysql) nu acceptă tranzacția. Prin urmare, fiecare interogare se află în tranzacția sa atomică.

HDBC oferă withTransactionpentru a vă permite să automatizați controlul tranzacțiilor pe un grup de interogări.

**Rularea declarațiilor:**

* **Select -** selectați
* **Insert -** inserați
* **Update -** actualizați
* **Delete –** ștergeți

**BIBLIOGRAFIE**

1. *Despre lucru cu baza de date in Haskell*. **[online]** Disponibil pe Internet la adresa: <https://en.wikibooks.org/wiki/Haskell/Database>. (accesat pe 30.11.2019).
2. *Despre HDBC (Haskell Connectness Database).* **[online]** Disponibil pe Internet la adresa: <https://github.com/hdbc/hdbc/wiki>. (accesat pe 30.11.2019).
3. *Despre DBI (Database independent interface to Perl)*. **[online]** Disponibil pe Internet la adresa: <https://metacpan.org/pod/distribution/DBI/DBI.pm>. (accesat pe 30.11.2019).
4. *Despre ODBC driver for HDBC*. **[online]** Disponibil pe Internet la adresa: <http://hackage.haskell.org/package/HDBC-odbc>. (accesat pe 30.11.2019).
5. *Despre MySQL driver for HDBC*. **[online]** Disponibil pe Internet la adresa: <http://hackage.haskell.org/package/HDBC-mysql>. (accesat pe 30.11.2019).
6. *Despre HDBC FAQs*. **[online]** Disponibil pe Internet la adresa: <https://github.com/hdbc/hdbc/wiki/FrequentlyAskedQuestions>. (accesat pe 30.11.2019).
7. *Despre bibliotecile MySQL de la Haskell*. **[online]** Disponibil pe Internet la adresa: <http://www.serpentine.com/blog/2010/09/04/dealing-with-fragile-c-libraries-e-g-mysql-from-haskell/>. (accesat pe 30.11.2019).