

Semestrální práce z předmětu KIV/DB2 Hledání min

Jakub Heidtke - [heidtke@students.zcu.cz](mailto:heidtke@students.zcu.cz)

7.6.2017

# 

[**Zadání**](#_la3rc79r8jg7) **2**

[**Datová analýza**](#_x6q73yur2fga) **5**

[**Funkční analýza**](#_rbwtdcndla8a) **6**

[Vybrané pohledy](#_a1r4x6mwdlu1) 6

[Poražení](#_5r7dovd1vikc) 6

[Vítězové](#_3xtjtq4iiyok) 6

[Vybraná uložená funkce](#_9gd8nusyajc4) 7

[Funkce Mnoho\_Min](#_bzl7xdi6emvb) 7

[Vybraná uložená procedura](#_qq9vnnc8pyyi) 8

[Procedura SPOCITEJ\_OBLAST](#_mxy0va8ieefc) 8

[Triggery volající výše uvedenou proceduru a funkci](#_s7vgshhifjug) 10

[Trigger after insert oblast](#_m7vt2w2ard5z) 10

[Trigger before insert mina](#_43k09jrufq4z) 10

[**Scénáře**](#_2aiso17jz74r) **11**

[Zahájení hry](#_loc3xakp9duq) 11

[Vypsání oblasti](#_tljxc9h4udlz) 11

[Odkrytí pole](#_gg0seio4mtlv) 12

[**Závěr**](#_sxyydzzgjs8j) **12**

# 

# Zadání

Cílem této práce je navrhnout a vytvořit relační databázi pro hraní známé počítačové hry Hledání min. Protože bude řešena pouze databázová vrstva aplikace, snažte se co nejvíce programových rutin uložit do databáze a také zajistěte jejich automatickou aktivaci při nastalé události.

Herní oblast má zpravidla tvar obdélníku, ve kterém se nachází několik min. Velikost oblasti a počet min v ní definuje obtížnost hry. Hráč si může vybrat jednu ze tří základních obtížností nebo si může definovat obtížnost vlastní. Úkolem hráče je odkrýt všechna pole oblasti, která nejsou zaminována. Hráči se bude od začátku hry měřit čas, aby bylo možné dosažené výsledky porovnávat. Po odkrytí libovolného pole (hráčem nebo databází) může nastat jedna z těchto událostí:

* hráč šlápl na minu. Hra končí neúspěchem a výsledek se zaznamená do databáze
* odkryto poslední pole, na kterém není mina. Hra končí úspěchem, protože zbylá neodkrytá pole obsahují miny. Také v tomto případě se výsledek uloží do databáze.
* odkryto pole, které je volné a nesousedí s žádným zaminovaným polem. V tomto případě databáze automaticky odkryje všechna sousední (mají společný min. jeden vrchol) pole.

Pro snazší hraní si může hráč označovat ta pole, o kterých si myslí, že jsou zaminovaná. K tomuto rozhodnutí mu pomohou čísla již odkrytých polí, která určují, s kolika zaminovanými poli toto pole sousedí. Takto označené pole nelze odkrýt, ale toto označení lze kdykoliv zrušit.

**V relační databázi budou evidována data v těchto tabulkách:**

|  |  |
| --- | --- |
| OBTIZNOST | Každá hra musí mít definovanou obtížnost. Buď bude vybrána základní, nebo si hráč definuje vlastní obtížnost. Hodnoty parametrů vlastní obtížnosti se ukládají do jiné tabulky. Podle originální hry jsou základní obtížnosti nastaveny takto:   * Začátečník: 9 řádků x 9 sloupců, 10 min * Pokročilý: 16 řádků x 16 sloupců, 40 min * Expert: 16 řádků x 30 sloupců, 99 min |
| OMEZENI | Každá vlastní obtížnost musí splňovat jistá omezení. Např. počet řádků či sloupců nesmí být menší než 9 a větší než 100. Také je vhodné pohlídat, aby počet rozmístěných min v zaminované oblasti nebyl příliš velký, např. nepřekročil 40 procent její velikosti. |
| OBLAST | Elementární část zaminované oblasti definovaná svými souřadnicemi, která může nést minu nebo informaci, s kolika zaminovanými poli sousedí. |
| POLE | Elementární část zaminované oblasti definovaná svými souřadnicemi, která může nést minu nebo informaci, s kolika zaminovanými poli sousedí. |
| MINA | Hráčem označovaná pole, o kterých si myslí, že jsou zaminovaná. |
| TAH | Hráčem odkrývaná pole v zaminované oblasti. Ke každému tahu se bude automaticky ukládat časová značka, kdy byl tah vykonán. |
| STAV | Číselník obsahující, v jakých stavech se hra může vyskytovat. Stavy mohou být tyto: rozehraná, úspěšně ukončená, neúspěšně ukončená. |
| HRA | Průběžně aktualizované informace o probíhající hře. Obsahuje časové značky prvního a naposledy provedeného tahu, počet označených min a stav hry. |

**Z uložených dat v databázi vytvořte databázové pohledy, které nabídnou tato data:**

|  |  |
| --- | --- |
| CHYBNE\_MINY | Seznam polí v zaminovaných oblastech, které byly chybně označené jako zaminované. |
| VITEZOVE | Výsledková tabulka her, které byly úspěšně dokončené. Měla by ukazovat parametry obtížnosti dané hry. |
| PORAZENI | Výsledková tabulka her, které byly neúspěšně dokončené. Měla by navíc ukazovat, kolik min bylo správně odhaleno. |
| OBLAST\_TISK | Zobrazení zaminované oblasti včetně odkrytých polí a (hráčem) označených min. Každý řádek oblasti bude zobrazen voláním funkce RADEK\_OBLASTI. |

**V databázi budou uloženy a používány tyto procedury:**

|  |  |
| --- | --- |
| ZAMINUJ\_OBLAST | Položení min na (náhodná) místa v definované oblasti podle její obtížnosti. |
| SPOCITEJ\_OBLAST | Pro každé nezaminované pole v oblasti spočítá, s kolika zaminovanými poli sousedí. |
| ODKRYJ\_POLE | Rekurzivní procedura, která pro právě odkryté pole, které nesousedí s žádným zaminovaným polem, odkryje všechna jeho sousední pole |
| OZNAC\_MINY | Po úspěšném dohrání hry budou dosud neodkrytá pole označená jako zaminovaná. |

**V databázi budou uloženy a používány tyto funkce:**

|  |  |
| --- | --- |
| SPATNY\_PARAMETR | Hodnota parametru vlastní obtížnosti (ne)překračuje definovaná omezení |
| RADEK\_OBLASTI | Vrátí řetězec znaků ukazující aktuální podobu daného řádku oblasti |
| ODKRYTA\_MINA | Právě odkryté pole skrývá minu, což znamená neúspěšný konec hry |
| MNOHO\_MIN | Nelze označit více zaminovaných polí, než kolik min je v oblasti. |
| VYHRA | Počet neodkrytých polí se rovná počtu min, které se v oblasti nachází. Pokud ano, hra končí úspěchem. |

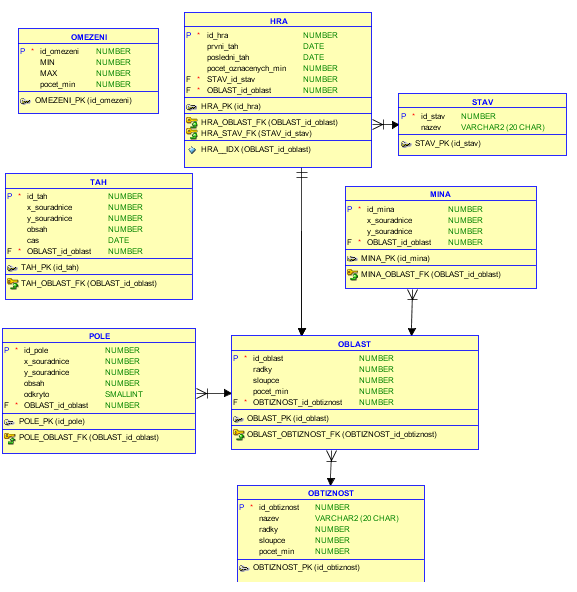
**O automatické činnosti v databázi se postarají triggery o:**

* hlídání hodnot parametrů vlastní obtížnosti (volání funkce SPATNY\_PARAMETR).
* přenesení hodnot parametrů obtížnosti pro aktuální oblast.
* zaminování nastavené oblasti (volání procedury ZAMINUJ\_OBLAST) a očíslování polí této oblasti (volání procedury SPOCITEJ\_OBLAST).
* zabránění odkrytí pole, pokud hra skončila.
* zabránění odkrytí pole, které je hráčem označené jako zaminované.
* hlídání počtu polí označených jako zaminované (volání funkce MNOHO\_MIN).
* volání automatických kontrol (volání funkcí ODKRYTA\_MINA a VYHRA) a případně akcí (volání procedury ODKRYJ\_POLE) při odkrytí pole.
* průběžnou aktualizaci hry po každém jejím tahu.
* označení min, pokud hra skončila úspěšně (volání procedury OZNAC\_MINY).

**Konfigurace, spuštění a průběh hry**

1. Jednorázová konfigurace databáze:
   1. Naplnění tabulky OBTIZNOST daty reprezentující obtížnosti hry
   2. Naplnění tabulky OMEZENI daty definující omezení pro vlastní obtížnost hry.
   3. Naplnění tabulky STAV daty odpovídající různým stavům hry.
2. Zahájení hry vložením nového záznamu do tabulky OBLAST. Automaticky se spustí plnění tabulky POLE odpovídající daty v podobě definované zaminované oblasti a také se vytvoří nový záznam v tabulce HRA.
3. Dále má hráč na výběr jednu z možností:
   1. Zobrazit si aktuální podobu zaminované oblasti volání pohledu OBLAST\_TISK.
   2. Odkrytí libovolného pole vložením záznamu do tabulky TAH. Dále dochází k aktualizaci příslušného záznamu v tabulce HRA.
   3. Označení libovolného pole jako mina vložením záznamu do tabulky MINA.
   4. Zrušení označení zaminovaného pole smazáním odpovídajícího záznamu
4. Pokud hra pokračuje, pokračuj bodem 3, jinak bodem 5 (úspěch) nebo 6 (neúspěch).
5. Hra skončila úspěchem. Je vhodné si zobrazit:
   1. Výsledkovou listinu vítězů voláním pohledu VITEZOVE.
   2. Zobrazení odminované oblasti voláním pohledu OBLAST\_TISK
6. Hra skončila neúspěchem. Je vhodné si zobrazit:
   1. Výsledkovou listinu poražených voláním pohledu PORAZENI.
   2. Seznam chybně označených min voláním pohledu CHYBNE\_MINY.
7. Novou hru zahájíme bodem 2.

# Datová analýza



# Funkční analýza

## Vybrané pohledy

### Poražení

Pohled zobrazuje výsledkovou tabulka her, které byly neúspěšně dokončené a kolik min bylo správně odhaleno.

CREATE OR REPLACE FORCE VIEW "JH"."PORAZENI" ("ID\_HRA", "PRVNI\_TAH", "POSLEDNI\_TAH", "OBTIZNOST", "SPRAVNE\_MINY") AS

select hra.id\_hra,HRA.prvni\_tah,HRA.POSLEDNI\_TAH,nvl(obtiznost.nazev,'Vlastni') obtiznost,nvl(spravne\_miny,0) spravne\_miny

from hra inner join oblast on hra.OBLAST\_ID\_OBLAST =oblast.ID\_OBLAST

left join obtiznost on oblast.obtiznost\_id\_obtiznost = obtiznost.id\_obtiznost

left join

(select HRA.ID\_HRA,POCET\_OZNACENYCH\_MIN - count(id\_hra) spravne\_miny

from hra

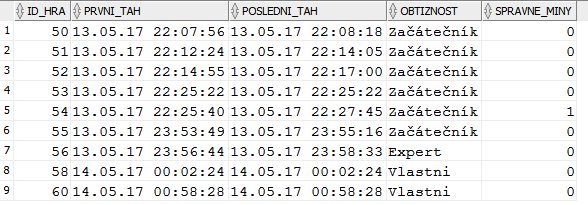
inner join CHYBNE\_MINY on hra.OBLAST\_ID\_OBLAST = CHYBNE\_MINY.OBLAST\_ID\_OBLAST

where stav\_id\_stav = 3

group by HRA.ID\_HRA,POCET\_OZNACENYCH\_MIN) temp on temp.ID\_HRA = HRA.ID\_HRA

where stav\_id\_stav = 3

order by HRA.id\_hra;



### Vítězové

Pohled ukazuje výsledkovou tabulka her, které byly úspěšně dokončené. Rovněž ukazuje i parametry obtížnosti dané hry.

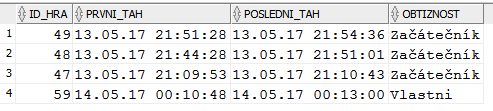
CREATE OR REPLACE FORCE VIEW "JH"."VITEZOVE" ("PRVNI\_TAH", "POSLEDNI\_TAH", "OBTIZNOST") AS

select prvni\_tah,POSLEDNI\_TAH,nvl(obtiznost.nazev,'Vlastni') obtiznost

from hra inner join oblast on hra.OBLAST\_ID\_OBLAST =oblast.ID\_OBLAST

left join obtiznost on oblast.obtiznost\_id\_obtiznost = obtiznost.id\_obtiznost

where stav\_id\_stav = 2;



## Vybraná uložená funkce

### Funkce Mnoho\_Min

Funkce kontroluje zda se hráč pokouší označit více min než v oblasti je.

create or replace FUNCTION MNOHO\_MIN   
(  
 OBLAST\_ID IN NUMBER   
) RETURN NUMBER AS   
ln\_pocet\_min number;  
ln\_pocet\_ozn\_min number;  
BEGIN  
 begin  
 select count(\*)into ln\_pocet\_ozn\_min from pole where OBLAST\_ID\_OBLAST = OBLAST\_ID  
 and oznaceno = 1;  
 exception when no\_data\_found then  
 ln\_pocet\_ozn\_min := 0;  
 end;  
   
 select POCET\_MIN into ln\_pocet\_min from OBLAST where ID\_OBLAST = OBLAST\_ID;  
   
 if ln\_pocet\_min = ln\_pocet\_ozn\_min then   
 return 1;  
 else  
 return 0;  
 end if;  
END;

## Vybraná uložená procedura

### Procedura SPOCITEJ\_OBLAST

Tato procedura počítá kolik se v okolí odkrytého pole nachází min.

create or replace PROCEDURE SPOCITEJ\_OBLAST   
(  
 OBLAST\_ID NUMBER,  
 RADKY NUMBER,  
 SLOUPCE NUMBER  
  
) AS  
bln\_mina number;  
pocet\_min number;  
  
BEGIN  
 FOR y IN 1..SLOUPCE LOOP  
 FOR x IN 1..RADKY LOOP  
 pocet\_min :=0;  
 begin  
 SELECT case obsah when '!' then 1 else 0 end into bln\_mina  
 FROM POLE WHERE OBLAST\_ID\_OBLAST = OBLAST\_ID  
 AND X\_SOURADNICE = x-1  
 and y\_souradnice = y;  
 exception when NO\_DATA\_FOUND then  
 bln\_mina := 0;  
 end;  
 pocet\_min := pocet\_min+ bln\_mina;   
 begin  
 SELECT case obsah when '!' then 1 else 0 end into bln\_mina  
 FROM POLE WHERE OBLAST\_ID\_OBLAST = OBLAST\_ID  
 AND X\_SOURADNICE = x-1  
 and y\_souradnice = y-1;  
 exception when NO\_DATA\_FOUND then  
 bln\_mina := 0;  
 end;  
 pocet\_min := pocet\_min+ bln\_mina;   
 begin  
 SELECT case obsah when '!' then 1 else 0 end into bln\_mina  
 FROM POLE WHERE OBLAST\_ID\_OBLAST = OBLAST\_ID  
 AND X\_SOURADNICE = x  
 and y\_souradnice = y-1;  
 exception when NO\_DATA\_FOUND then  
 bln\_mina := 0;  
 end;  
 pocet\_min := pocet\_min+ bln\_mina;  
 begin  
 SELECT case obsah when '!' then 1 else 0 end into bln\_mina  
 FROM POLE WHERE OBLAST\_ID\_OBLAST = OBLAST\_ID  
 AND X\_SOURADNICE = x+1  
 and y\_souradnice = y-1;  
 exception when NO\_DATA\_FOUND then  
 bln\_mina := 0;  
 end;  
 pocet\_min := pocet\_min+ bln\_mina;  
 begin  
 SELECT case obsah when '!' then 1 else 0 end into bln\_mina  
 FROM POLE WHERE OBLAST\_ID\_OBLAST = OBLAST\_ID  
 AND X\_SOURADNICE = x+1  
 and y\_souradnice = y;  
 exception when NO\_DATA\_FOUND then  
 bln\_mina := 0;  
 end;  
 pocet\_min := pocet\_min+ bln\_mina;   
 begin  
 SELECT case obsah when '!' then 1 else 0 end into bln\_mina  
 FROM POLE WHERE OBLAST\_ID\_OBLAST = OBLAST\_ID  
 AND X\_SOURADNICE = x+1  
 and y\_souradnice = y+1;  
 exception when NO\_DATA\_FOUND then  
 bln\_mina := 0;  
 end;  
 pocet\_min := pocet\_min+ bln\_mina;   
 begin  
 SELECT case obsah when '!' then 1 else 0 end into bln\_mina  
 FROM POLE WHERE OBLAST\_ID\_OBLAST = OBLAST\_ID  
 AND X\_SOURADNICE = x  
 and y\_souradnice = y+1;  
 exception when NO\_DATA\_FOUND then  
 bln\_mina := 0;  
 end;  
 pocet\_min := pocet\_min+ bln\_mina;  
 begin  
 SELECT case obsah when '!' then 1 else 0 end into bln\_mina  
 FROM POLE WHERE OBLAST\_ID\_OBLAST = OBLAST\_ID  
 AND X\_SOURADNICE = x-1  
 and y\_souradnice = y+1;  
 exception when NO\_DATA\_FOUND then  
 bln\_mina := 0;  
 end;  
 pocet\_min := pocet\_min+ bln\_mina;   
   
 update pole   
 set obsah = pocet\_min  
 WHERE OBLAST\_ID\_OBLAST = OBLAST\_ID  
 AND X\_SOURADNICE = x  
 and y\_souradnice = y  
 and OBSAH != '!';  
 END LOOP;   
 END LOOP;   
END SPOCITEJ\_OBLAST;

## Triggery volající výše uvedenou proceduru a funkci

### Trigger after insert oblast

Po vložení záznamu do tabulky oblast se oblast automaticky zaminuje a vypočítají se počty min.

create or replace TRIGGER TRAI\_OBLAST

AFTER INSERT ON OBLAST

FOR EACH ROW

BEGIN

--vytvoreni zaznamů v tabulce pole

FOR x IN 1..:NEW.SLOUPCE LOOP

FOR y IN 1..:NEW.RADKY LOOP

Insert into Pole values (seq\_pole.nextval,y,x,0,0,:NEW.ID\_OBLAST,0);

END LOOP;

END LOOP;

ZAMINUJ\_OBLAST(:NEW.ID\_OBLAST,:NEW.RADKY,:NEW.SLOUPCE,:NEW.POCET\_MIN);

SPOCITEJ\_OBLAST (:NEW.ID\_OBLAST,:NEW.RADKY,:NEW.SLOUPCE);

--vytvoreni zaznamu v tabulce hra

insert into hra values(seq\_hra.nextval,null,null,0,1,:NEW.ID\_OBLAST);

--zaminování oblasti

END;

### Trigger before insert mina

Při pokusu o označení miny vložením záznamu do tabulky mina se zkontroluje zda se hráč nepokouší označit více min než je ve hře, pokud se tak stane, je zde použita custom exception. Dále se inkrementuje počet označených min v tabulce hra

create or replace TRIGGER TRBI\_mina

BEFORE INSERT ON MINA

FOR EACH ROW

BEGIN

if MNOHO\_MIN(:new.OBLAST\_ID\_OBLAST) = 0 then

UPDATE POLE

SET OZNACENO = 1

where OBLAST\_ID\_OBLAST = :new.OBLAST\_ID\_OBLAST

and X\_SOURADNICE = :new.X\_SOURADNICE

and Y\_SOURADNICE = :new.Y\_SOURADNICE;

update HRA

set POCET\_OZNACENYCH\_MIN = (select count(\*) from pole where OBLAST\_ID\_OBLAST = :new.OBLAST\_ID\_OBLAST AND OZNACENO = 1)

where OBLAST\_ID\_OBLAST = :new.OBLAST\_ID\_OBLAST;

else

raise\_application\_error (-20099,'Nelze označit více zaminovaných polí, než kolik min je v oblasti. ');

--DBMS\_OUTPUT.PUT\_LINE('Nelze označit více zaminovaných polí, než kolik min je v oblasti. ');

end if;

END;

# Scénáře

## Zahájení hry

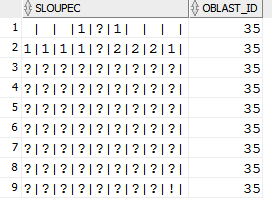
Hra bych doporučil zahájit dvěma způsoby:

1. voláním procedury VYTVOR\_ZAKLADNI\_HRU (obtiznost\_id) -procedura vybere z tabulky obtížnost parametry potřebné pro vytvoření záznamu v tabulce oblast a záznam vytvoří.
2. voláním procedury VYTVOR\_VLASTNI\_HRU (radky NUMBER, sloupce NUMBER, pocet\_min) - procedura pres funkci SPATNY\_PARAMETR ověří že zadané parametry odpovídají omezení v tabulce OMEZENI a následně vytvoří záznam v tabulce oblast.

## Vypsání oblasti

Dalším krokem je vypsání oblasti to provedeme selectem z pohledu OBLAST\_TISK.

select \* from OBLAST\_TISK where OBLAST\_ID = 35;



Vysvětlení políček:

? - neodkryté pole

1-9 - počet sousedících min

‘ ’ - odkryté pole

! - mina

## Odkrytí pole

Dále je možné odkrývat pole, dokud hráč neodkryje všechny, nebo dokud nenarazí na minu. Odkrytí pole je možno zavoláním procedury ODKRYJ\_POLE(oblast\_id, x, y)

# Závěr

Při hře jsem si osvojil nové dovednosti při práci s triggery. Samotný vývoj hry v SQL Developeru mi až tolik nového nepřinesl, protože je to něco čím se zabývám při své profesi. Semestrální práce pro mě byla z velké části zopakováním mých dovedností. Myslím si že zadání se mi podařilo splnit.