FAKULTA RIADENIA A INFORMATIKY

ŽILINSKÁ UNIVERZITA V ŽILINE

ANALÝZA DÁT DOPRAVNÝCH NEHÔD

SEMESTRÁLNA PRÁCA Z PREDMETU DATABÁZOVÁ ANALYTIKA A VÝKONNOSŤ

Autor: **Bc. Matej Poljak**

Učiteľ: **doc. Ing. Michal Kvet, PhD.**

Akademický rok: **2024/2025**

Obsah

[Cieľ semestrálnej práce 3](#_Toc191857530)

[Popis dát 3](#_Toc191857531)

[Transformácia dát do databázy 4](#_Toc191857532)

[Databázová analytika 6](#_Toc191857533)

[Prehľad o počtoch dát 6](#_Toc191857534)

[TOP mesiac/e pre každý kraj podľa počtu umrtí za rok 2024 6](#_Toc191857535)

[N-tý mesiac s najväščím počtom nehôd 9](#_Toc191857536)

[Alkohol a dopravné nehody 11](#_Toc191857537)

[Mesačne najčastejšie zrazené zvieratá v roku 2024 17](#_Toc191857538)

[2,5 mesačný kĺzavý medián počtu zranených pre rok 2024 (s frekvenciou polmesiac) 17](#_Toc191857539)

[Optimalizácia výkonnosti 20](#_Toc191857540)

[Index pre skript získania percentuálneho podielu škôd nehôd s prítomnosťou alkoholu v TOP 1000 najvyššie ocenených škodách 20](#_Toc191857541)

[Index pre výpis TOP 3 najčastejšie zrazených zvierat za každý mesiac 23](#_Toc191857542)

[Optimalizácia spojenia tabuliek nehôd a vozidiel 25](#_Toc191857543)

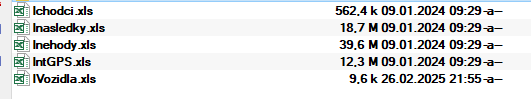
[Záver 26](#_Toc191857544)

# Cieľ semestrálnej práce

Cieľom práce bolo získať dáta, nad ktorými následne vykonáme analýzu. Touto analýzou zistíme zaujímavé informácie vyplývajúce z dát a tak preukážeme schopnosť analytického pohľadu na ne. Na to sme mali využiť analytické a agregačné funkcie, ktoré platforma Oracle poskytuje. Okrem toho sme sa mali pokúsiť optimalizovať rýchlosť dopytu dát s využitím indexov.

# Popis dát

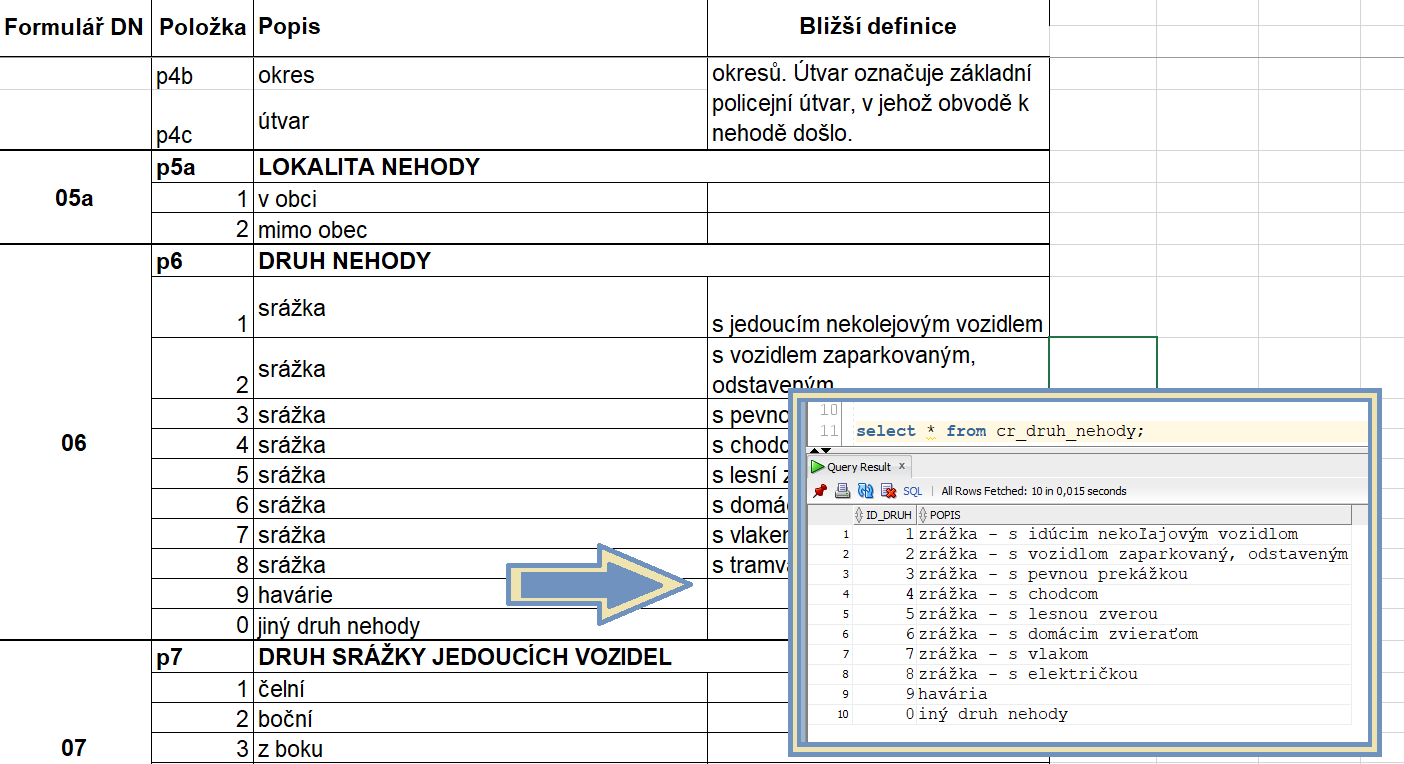
Na prácu sme si zvolili údaje o dopravných nehodách v Českej republike, ktoré poskytuje česká polícia na mesačnej báze. Dáta môžeme získať zo stránky [*https://policie.gov.cz/clanek/statistika-nehodovosti-900835.aspx*](https://policie.gov.cz/clanek/statistika-nehodovosti-900835.aspx), kde si zvolíme rok, z ktorého dáta požadujeme. Stiahneme RAR zložku, ktorá obsahuje súbory s príponou *xls*. Súbory sú rozdelené na záznamy o nehodách, o vozidlách spojených s nehodou, o chodcoch spojených s nehodou a záznamy o následkoch dopravných nehôd (obrázok 1). My si vyberieme pre ďalšie spracovanie dáta o nehodách všeobecne a dáta o vozidlách, čiže dva typy *xls* súborov, pričom použijeme dáta z 3 rôznych rokov: 2023, 2024, 2025.



Obrázok 1 – obsah stiahnutej RAR zložky

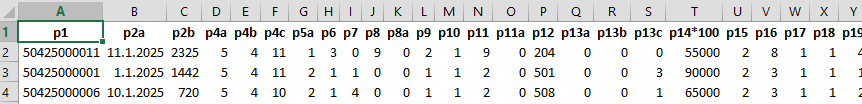
# Transformácia dát do databázy

Na stránke, uvedenej v predchádzajúcej kapitole, sme si stiahli aj dokument *Položky\_formuláře\_WEB.xlsx* (príklad zobrazený na obrázku 2), ktorý obsahuje význam a hodnoty pre jednotlivé stĺpce, ktorými sú naplnené záznamy o nehodách a vozidlách zúčastnených pri nehode. Na základe týchto popisov si pre každý referencujúci stĺpec vytvoríme databázovú tabuľku, ktorá obsahuje významy jednotlivých ID hodnôt, na ktoré sa záznamy o nehodách a vozidlách odkazujú.



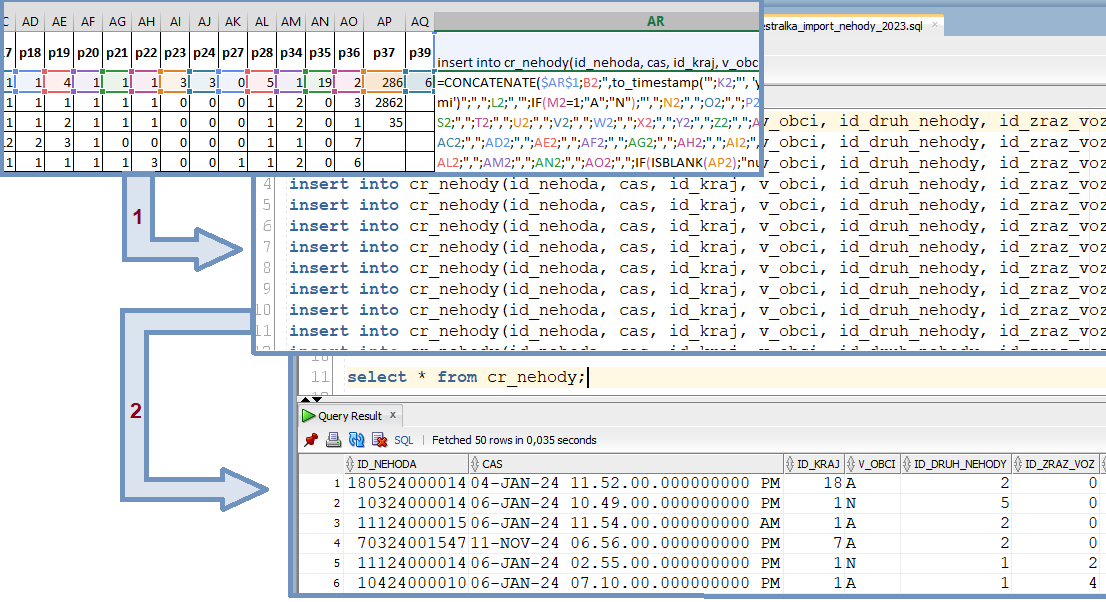
Obrázok 2 – popis hodnôt stĺpcov, na ktoré sa záznamy referencujú

Potom som si vytvoril databázovú tabuľky *cr\_nehody* a *cr\_vozidla* podľa štruktúry jedného riadku súboru *lnehody.xls*, resp. *lvozidla.xls*. Takýto záznam je ukázaný na obrázku 3.



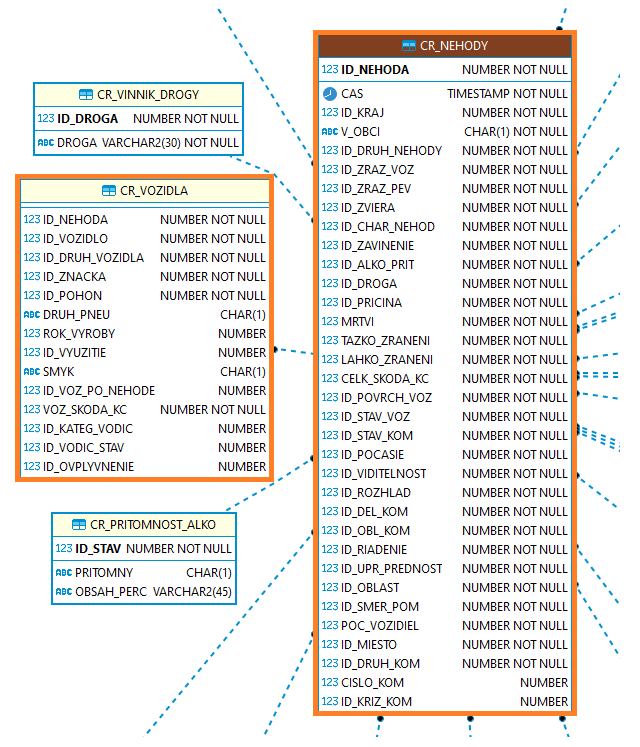
Obrázok 3 – ukážka záznamov o nehodách v xls súbore

Následne som mohol naplniť tabuľky o nehodách a vozidlách hodnotami zo súborov. Najprv som si napísal príkaz, ktorý mi v *xls* súbore vygeneroval sql príkaz *insert*. Všetky vytvorené sql príkazy som si uložil do súboru a potom som spustil vykonanie skriptu v nástroji *SqlDeveloper*. Tento postup je znázornený na obrázku 4.



Obrázok 4 – transformácia záznamov do databázy

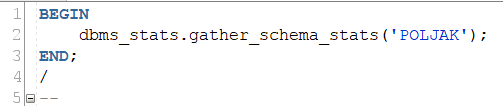
Pre prácu teda budeme využívať (okrem všetkých tabuliek s popismi) databázovú reprezentáciu záznamov zodpovedajúcu obrázku 5.



Obrázok 5 – dátový model najpodstatnejších tabuliek zvýraznených oranžovým rámikom

# Databázová analytika

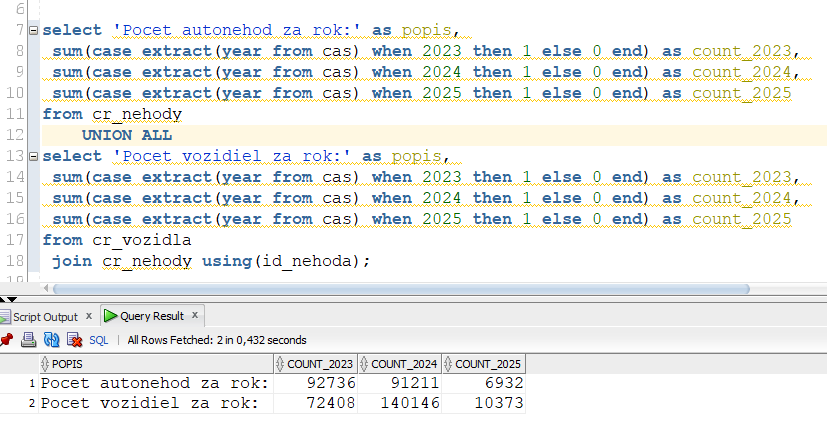
V tejto časti si ukážeme zopár zaujímavých informácií, ktoré sme získali databázovým príkazom *select* spolu s využitím analytických a agregačných funkcií. Pred analýzou výkonnosti si najprv nad všetkými tabuľkami, ktoré prislúchajú k schéme „*POLJAK*“ pregenerujeme štatistiky, pretože sme vložili veľké množstvo nových dát (obrázok 6).



Obrázok 6 – pregenerovanie štatistík pre aktualizáciu kvôli novým tabuľkám

## Prehľad o počtoch dát

Na úvod si pre lepšiu predstavu zistíme najzákladnejšiu štatistiku a to počty záznamov o nehodách a vozidlách za jednotlivé roky. Túto informáciu vieme pekne zobraziť pomocou pivotovej transformácie dosiahnutej použitím agregačnej funkcie *sum*, ktorá pripočíta do stĺpca záznam prave vtedy, keď spĺňa kritérium príslušnosti roku, pre ktorý je daný stĺpec určený. Stĺpec *cas* je *NOT NULL*, preto nemusíme ošetrovať podmienku s chýbajúcou hodnotou. Takéto dopyty sme vytvorili nad oboma tabuľkami nehôd aj vozidiel a následne sme ich zjednotili pre ucelenú informáciu v jednom výpise. Skript aj výpis je zobrazený na obrázku 7.



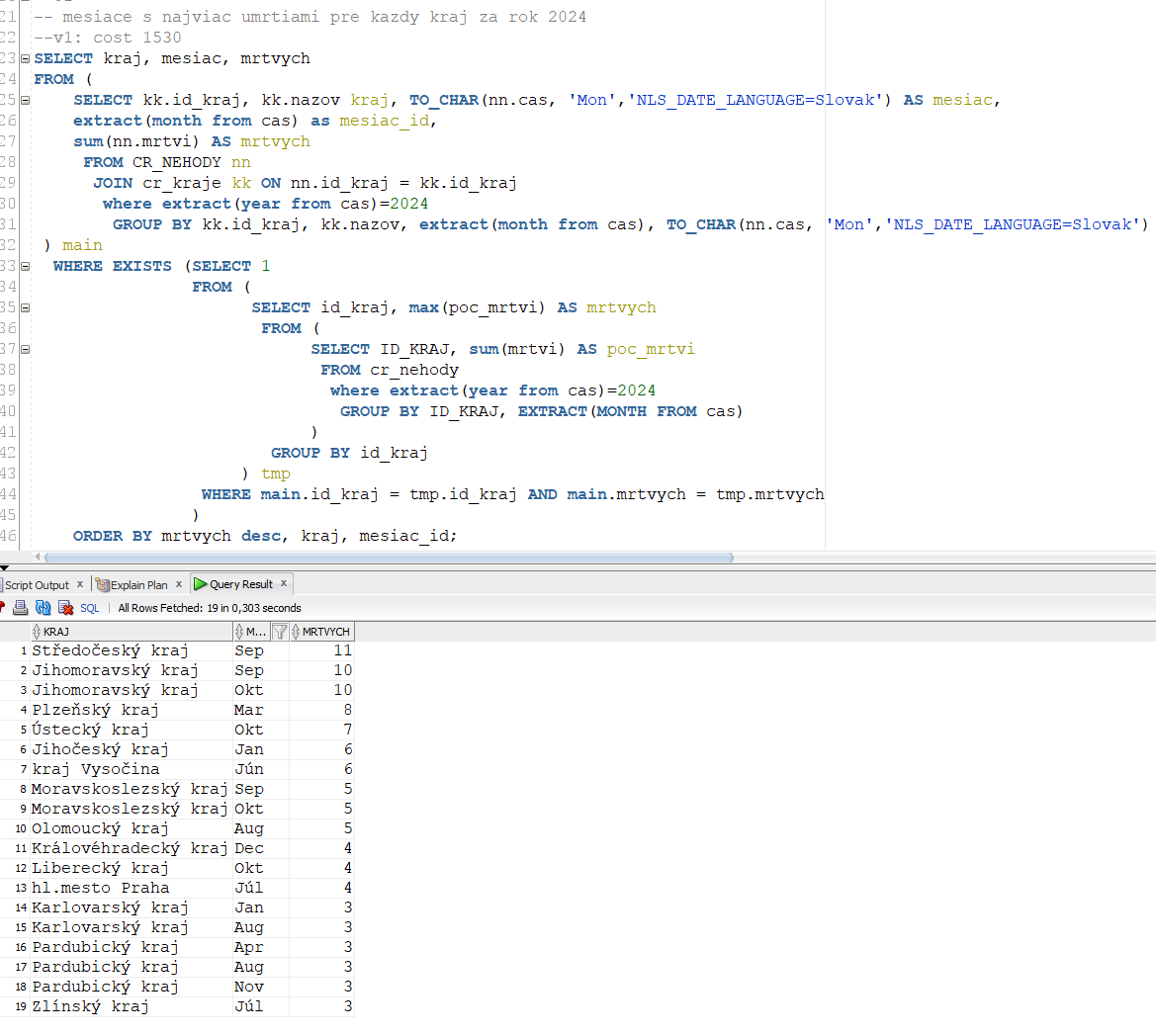
Obrázok 7 – počty nehôd a vozidel za jednotlivé evidované roky

Rok 2025 obsahuje pomerovo iba zlomok dát, pretože sme doteraz mali prístup len k záznamom za mesiac Január. Za rok 2023 je počet záznamov o vozidlách tiež zredukovaný – vybrali sme len necelú polovicu záznamov pre ukážku dát, ktoré sú vzorkou z tohto roku.

## TOP mesiac/e pre každý kraj podľa počtu umrtí za rok 2024

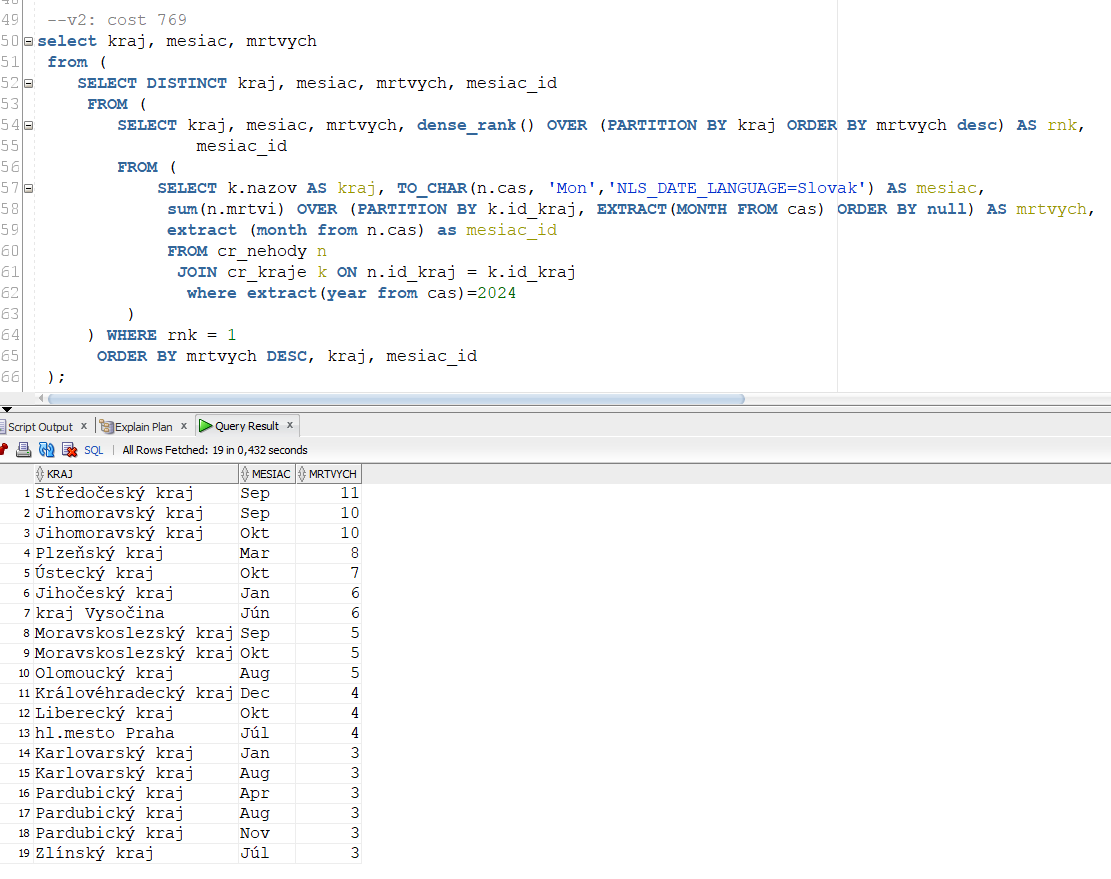
Prvou informáciou, ktorá nás môže zaujímať je koľko a v akých mesiacoch pre konkrétny kraj Českej republiky (ďalej ČR) bol zaznamenaný najvyšší počet úmrtí spôsobených dopravnou nehodou za vymedzené obdobie - rok 2024. Ukážeme si tri varianty ako napísať takýto dopyt na dáta a porovnáme ich náklady.

**Variant 1 – využitie vnoreného príkazu *select* (obrázok 8)**



Obrázok 8 – najviac mŕtvych pre kraj a mesiac za rok 2024 (variant 1)

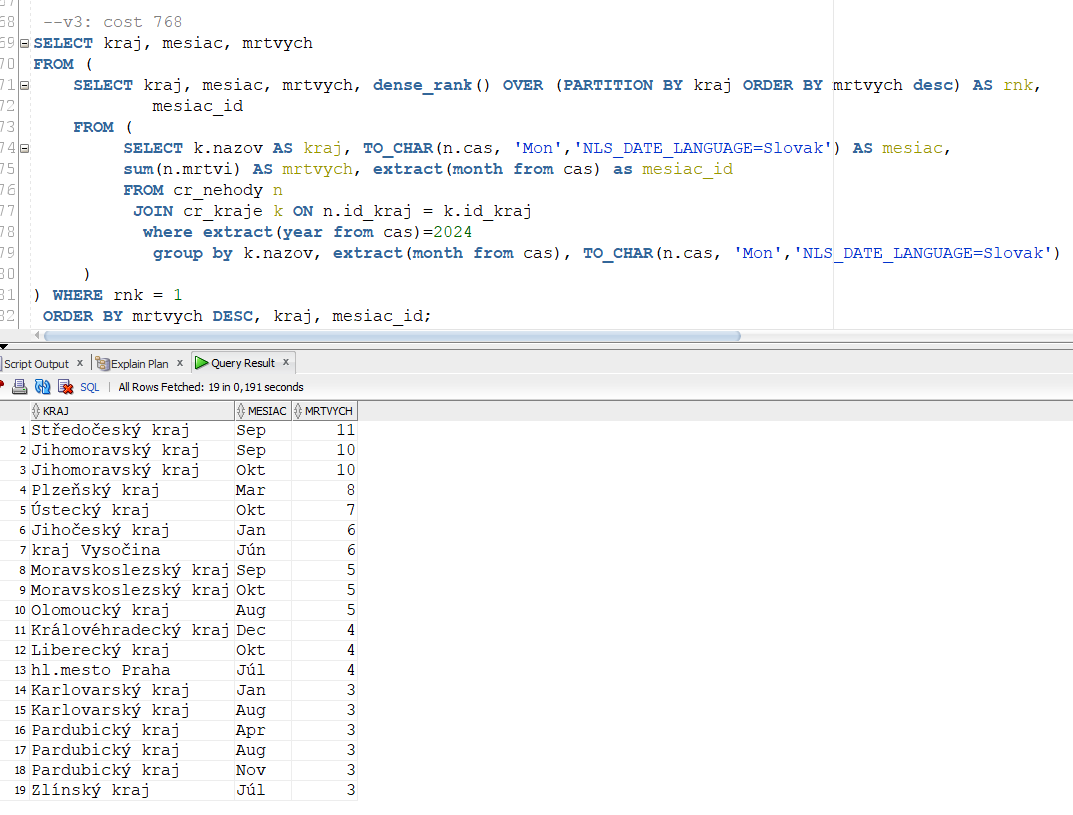
Vnorený *select*, ktorého výsledky označíme aliasom *tmp* vytvorí pre záznamy z tabuľky *cr\_nehody* skupiny podľa mesiaca a kraja, ku ktorým záznamy prislúchajú. Samozrejme, pred samotným zatriedením odfiltrujeme záznamy, ktoré sú mimo roku 2024. Pre skupiny potom získame počty záznamov, ku ktorým si zapamätáme identifikátor ich skupiny – stačí kraj. Následne pre každý kraj vyberieme iba záznam s maximálnym počtom nehôd. Výsledky tohto dopytu budeme vyhľadávať v druhom selecte označenom aliasom *main*, kde si budeme pamätať kraj, jeho názov, mesiac a počet mŕtvych dôsledkom dopravnej nehody. Ak sa takýto záznam zhoduje so záznamom z vnoreného príkazu *select tmp*, ktorý obsahuje maximálne počty, potom tento záznam budeme vypisovať ako výsledok nášho hľadania. Nevýhodou tohto riešenia je, že vnorený príkaz *select tmp* sa musel vykonať nad každým záznamom pre *select* *main*.

**Variant 2 – využitie analytických funkcií (obrázok 9)**

Obrázok 9 - najviac mŕtvych pre kraj a mesiac za rok 2024 (variant 2)

V druhom variante využijeme namiesto vnoreného príkazu *select* analytickú funkciu *sum*, v ktorej si spočítame všetkých mŕtvych pre daný kraj a mesiac, čo definujeme v klauzule *PARTITION BY*. Keďže ide o analytickú funkciu, počet záznamov sa neredukuje, iba každému záznamu podľa jeho skupiny priradí vypočítané číslo. Každý záznam v skupine dostane rovnakú hodnotu sumy, pretože nerobíme tzv. rolling sum, lebo podľa ničoho netriedime (*order by null*). Takto získané hodnoty ohodnotíme ďalšou analytickou funkciou *dense\_rank,* ktorá priradí každému záznamu v rámci skupiny (definovanej krajom) ohodnotenie podľa počtu mŕtvych s TOP ohodnotením pre záznamy s najvyšším počtom mŕtvych. Následne len vyberiem tie záznamy, ktoré dostali ohodnotenie 1 – „najlepšie“, to znamená, najvyšší počet mŕtvych pre každý kraj. Náklady v porovnaní s variantom 1 sú polovičné, pretože sme nemuseli zakaždým vyhodnocovať vnorený príkaz *select*, ale len raz sme si ku každému záznamu priradili jeho ohodnotenie v rámci skupiny na základe počtu.

**Variant 3 – využitie kombinácie agregačných a analytických funkcií (obrázok 10)**



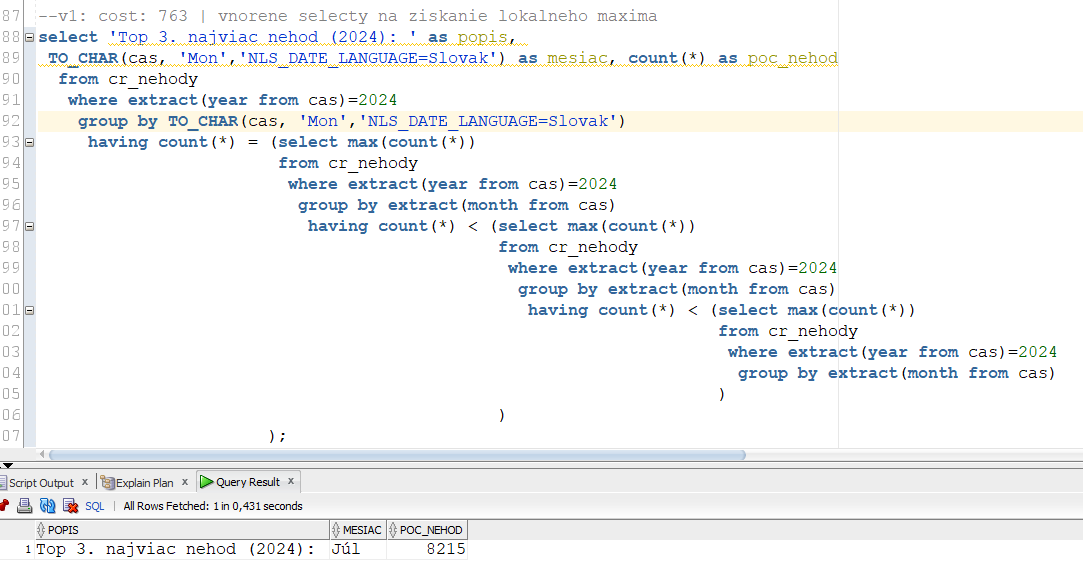
Obrázok 10 - najviac mŕtvych pre kraj a mesiac za rok 2024 (variant 3)

Tretí variant sa od druhého líši iba v spôsobe spočítania počtu mŕtvych pre skupiny definované krajom a mesiacom. Na to sme použili namiesto analytickej funkcie agregačnú funkciu *sum* tak, že sme použili klauzulu *GROUP BY*. Vďaka tomu sme spočítali sumy pre skupiny a tak sme aj, na rozdiel od analytickej funkcie, zredukovali počet záznamov tak, že sme dostali práve jeden záznam pre každú skupinu. Následne sme, podobne ako vo variante 2, iba ohodnotili počty a vybrali pre každý kraj mesiace s najvyšším počtom úmrtí. Porovnanie nákladov na vykonanie variantu 2 a variantu 3 sa javia byť skoro rovnaké (769 verzus 768, t.j. pri variante 3 sme ušetrili 1 jednotku nákladov).

## N-tý mesiac s najväščím počtom nehôd

Ďalej by nás mohlo zaujímať, ktoré 3 mesiace bolo zaznamenaných najviac nehôd a koľko ich presne bolo. Pre prvých *n* mesiacov by bol obtiažnejší výpis bez použitia analytickej funkcie. Síce si taký príklad neskôr ukážeme, ale zamerajme sa teraz len na výpis tretieho mesiaca v poradí s počtom najviac nehôd za rok 2024.

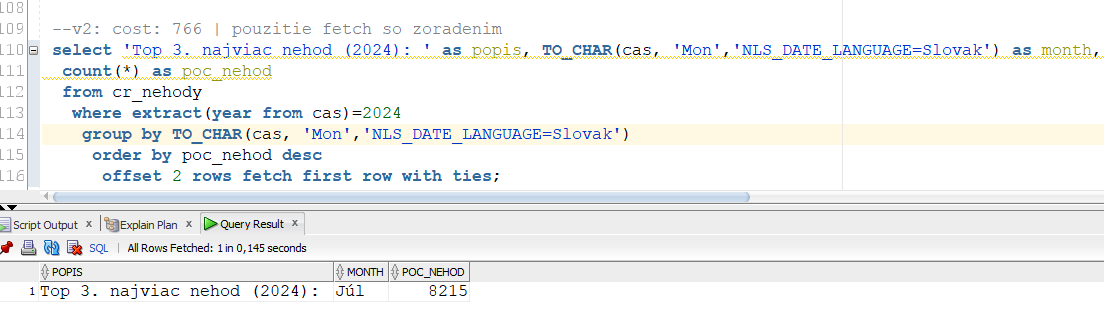
**Variant 1 – využitie vnoreného príkazu *select* (obrázok 11)**



Obrázok 11 - TOP 3. mesiac s najviac nehodami v 2024 (variant 1)

Prvý variant je postavený na vnorených príkazoch *select*, ktoré získajú maximum pri splnení podmienky roku 2024 a zároveň maximum nesmie prekročiť hranicu definovanú vnorenejším príkazom *select*. Takto si vytvoríme toľko vnorení, ktorú n-tú najväčšiu hodnotu v poradí potrebujeme, pričom najvnorenejší *select* predstavuje globálne maximum. Najmenej vnorený *select* obsahuje našu hľadanú n-tú hodnotu, na ktorú sa pozeráme z vonkajšieho príkazu *select*. Toto porovnávanie počtov na najvrchnejšej úrovni nerobíme kvôli získaniu daného počtu, pretože ten sme získali už prvým vnoreným príkazom *select*. Robíme to, aby sme zistili, pre ktorý mesiac je tento počet priradený a tak ho mohli spolu s počtom vypísať.

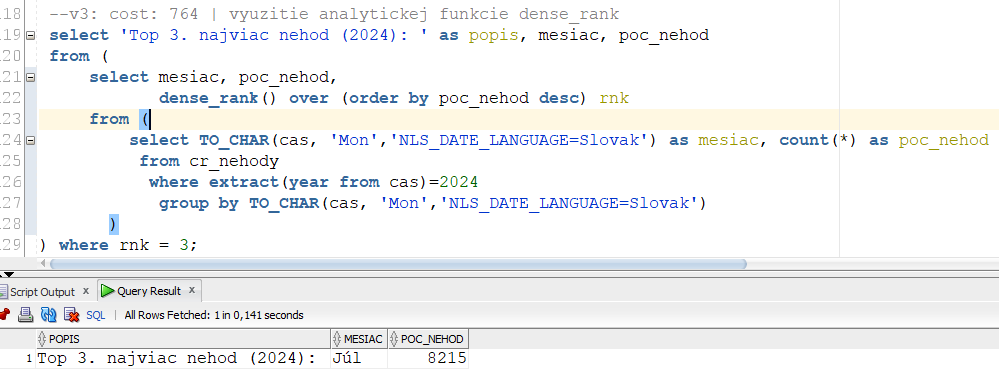
**Variant 2 – využitie klauzuly *fetch* (obrázok 12)**



Obrázok 12 - TOP 3. mesiac s najviac nehodami v 2024 (variant 2)

V druhom variante si pre každý mesiac roku 2024 zistíme počet nehôd a následne je dôležité zoradiť výsledky príkazu *select* vzostupne podľa počtu nehôd. Potom využijeme klauzulu *offset*, pomocou ktorej preskočíme prvé dva najlepšie mesiace a zoberiem s využitím *fetch first row with ties* iba riadky, ktoré podľa počtu predstavujú náš hľadaný top 3. mesiac. Toto riešenie ale nemusí byť správne, pokiaľ napríklad na druhom mieste sme mali dva mesiace. Vybrali by sme teda jeden záznam z dvoch, ktoré prislúchajú k druhému mesiacu a nedozvedeli by sme sa, že sme sa dopustili chyby. Náklady sa zhoršili voči prvému variantu iba o 3 jednotky, takže variant 1 a 2 sú podobne výkonné.

**Variant 3 – využitie analytickej funkcie *dense\_rank* (obrázok 13)**



Obrázok 13 - TOP 3. mesiac s najviac nehodami v 2024 (variant 3)

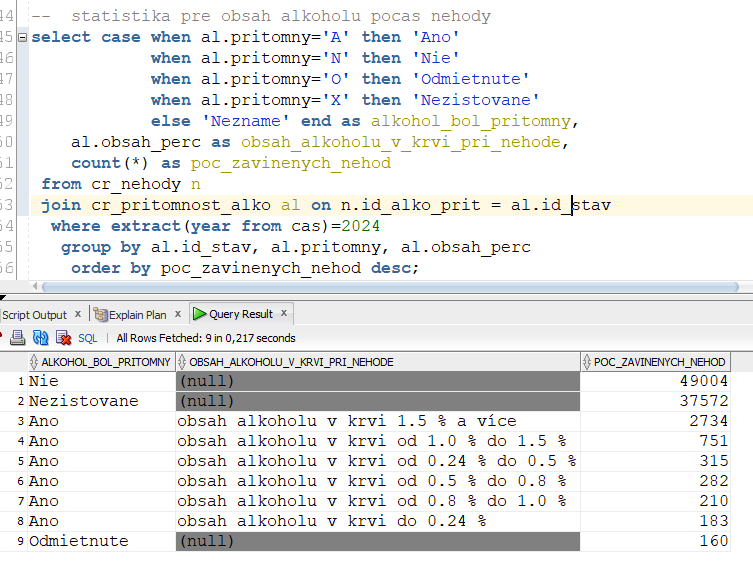
V poslednom variante si pomocou agregačnej funkcie *count* pre každý mesiac spočítame počet záznamov a následne tieto počty zostupne ohodnotíme pomocou analytickej funkcie *dense\_rank* a vyberieme záznamy s ohodnotením *3*. Výhodou tohoto variantu oproti prvým dvom je, že malou zmenou môžeme získať ľubovoľné *n*-té umiestnenie, resp. ľubovoľný interval za sebou idúcich umiestnení. Zmenili by sme iba podmienku, ktoré ohodnotenia chceme získať (t.j. aké čísla môže obsahovať stĺpec *rnk*). Čo sa týka nákladov, tie sa nezmenili (náklady sa zhoršili len o 1 jednotku oproti variantu 1), ale silno sa nám zvýšila miera možnej modifikácie výberov na základe umiestnenia, ktoré je výstupom analytickej funkcie.

## Alkohol a dopravné nehody

Niektoré dopravné nehody bývajú zbytočne zapríčinené požitím alkoholu pred jazdou, v dôsledku čoho sa schopnosti účastníkov cestnej komunikácie byť pozorný výrazne znižujú.

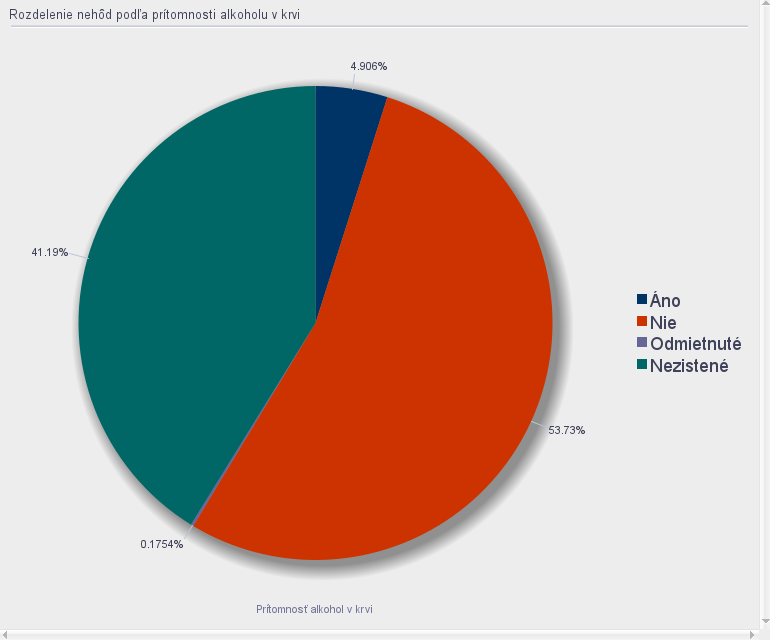
**Ročné početnosti podľa percentuálnej prítomnosti alkoholu v krvi pre rok 2024**

Ako prvé by nás mohlo zaujímať pri koľkých nehodách bola zistená prítomnosť alkoholu a v akom veľkom množstve. Pre zistenie sme vypočítali agregačnou funkciou *count* počty podľa druhov prítomnosti alkoholu v krvi pri nehodách, ako je to zobrazené na obrázku 14 spolu s výsledkami. Z výsledkov môžeme vyčítať, že okolo 50% prípadov alkohol nebol prítomný pri nehode. Zaujímavé je, že pri vyše 40% nehôd sa nezisťovala prítomnosť alkoholu v krvi. V približne 5% nehôd bola potvrdená prítomnosť alkoholu, pričom ak bola prítomnosť potvrdená, potom je vyše 60% pravdepodobnosť, že obsah alkoholu v krvi je viac ako 1.5%.



Obrázok 14 – ročné počty nehôd podľa prítomnosti alkoholu v krvi

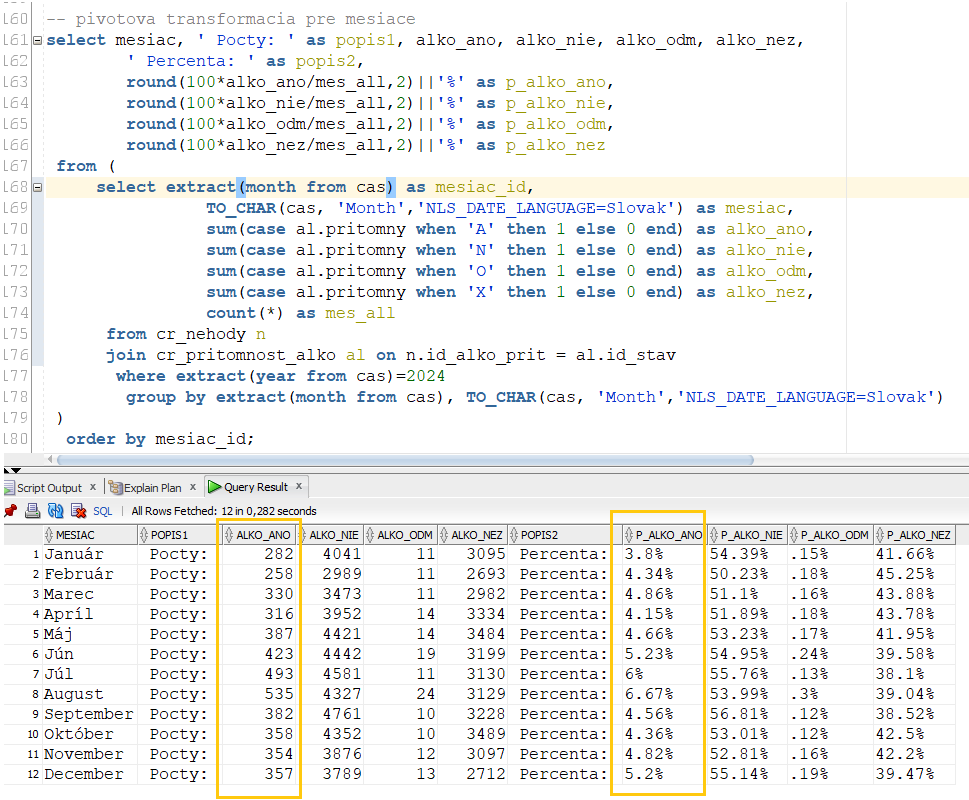
Na vizuálnu ukážku sme si v programe *SqlDeveloper* vytvorili report (obrázok 15), ktorý zobrazuje percentuálne podiely nehôd kde alkohol bol prítomný v krvi (modrá), kde nebol (červená), kde to nebolo zistené (zelená) a kde to bolo odmietnuté (jemne fialová). Vidíme, že skoro 5% nehôd bolo s nejakým obsahom alkoholu v krvi (tu abstrahujeme mieru obsahu). Zaujímavé je, že vyše 41% prípadov to nebolo zistené z rôznych dôvodov, ktoré nepoznáme, lebo dáta v *xls* súboroch túto informáciu neobsahovali.



Obrázok 15 – podiely počtov nehôd podľa zistenia obsahu alkoholu v krvi pre rok 2024

**Mesačné početnosti podľa percentuálnej prítomnosti alkoholu v krvi pre rok 2024**

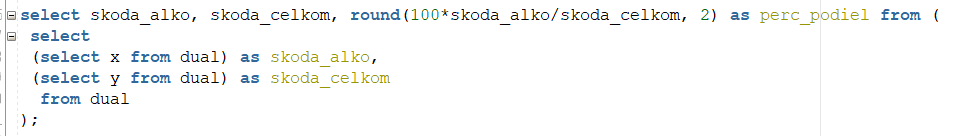
Okrem početností percentuálnej prítomnosti alkoholu v krvi ročnej nás môžu zaujímať aj mesačné početnosti alkoholu v krvi, ako sa celkovo počas roka vyvíjala. Pre lepšie pochopenie uvedieme aj relatívne percentuálne početnosti podielu zisteného alkoholu v krvi pre nehody v rámci mesiaca. Teraz už budeme abstrahovať od miery alkoholu v krvi a bude nás len zaujímať či alkohol bol prítomný, alebo nie. Výsledky spolu so skriptom sú uvedené na obrázku 16. Je vidno, že najviac nehôd so zisteným alkoholom v krvi sa zaznamenal v letných mesiacoch. Môže to byť spojené s tým, že v lete chodia ľudia väčšinou na dovolenky a viacej ľudí berie zodpovednosť vtedy menej vážne. Na konci roka v decembri môžeme tiež vidieť nárast prípadov, čo môže byť opäť istým spôsobom spojené s časom sviatkov.



Obrázok 16 – mesačné počty nehôd podľa prítomnosti alkoholu v krvi

**Percentuálny podiel škôd nehôd s prítomnosťou alkoholu v TOP 1000 najvyššie ocenených škodách**

Môžeme predpokladať, že nehody s prítomnosťou alkoholu môžu mať veľmi katastrofálne následky, preto nás bude zaujímať všeobecné ohodnotenie TOP 1000 najnákladnejších nehôd, čo je približne 1% všetkých nehôd za rok 2024. Z nich identifikujeme všetky, ktoré boli s prítomnosťou alkoholu a ich ohodnotenie dáme do pomeru s celkovými škodami TOP 1000 nehôd, ako to vysvetľuje šablóna príkazu *select* na obrázku 17.

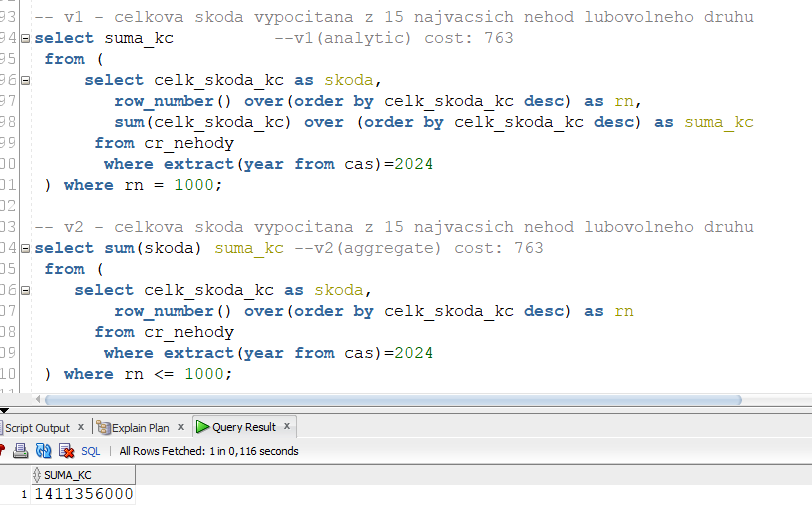


Obrázok 17 – abstrakcia výpočtu percentuálneho podielu pre hľadanú skutočnosť v jazyku SQL

Rozoberme si postupne príkaz *select* pre čitateľ a menovateľ zvlášť.

1. Menovateľ (obrázok 18) – celková škoda TOP 1000 nehôd s najväčšími vyčíslenými škodam v peňažnej mene Kč. Potrebujeme si zoradiť nehody za rok 2024 zostupne podľa veľkosti škody v Kč a následne vybrať prvých 1000 záznamov. Keďže nezohľadňujeme koľko záznamov môže mať teoreticky rovnakú hodnotu škody (nehodnotíme princípom hodnotenia olympijských hier), ale stačí nám jednoducho 1000 ľubovoľných najnákladnejších nehôd, môžeme použiť analytickú funkciu *row\_number* na získanie ohodnotenia pre každý záznam podľa jeho vyčíslenej škody. Tu máme 2 spôsoby sčítania prvých 1000 záznamov s najlepším ohodnotením.
   1. Prvý spôsob je pomocou analytickej funkcie *sum* s využitím tzv. rolling sum, to znamená postupné nasčítanie hodnôt ako suma hodnôt stĺpca predchádzajúcich riadkov s vyšším ohodnotením (príp. rovnakým pri variantoch funkcie *rank*) a aktuálnym riadkom. Aby nasčítavanie fungovalo správne, musíme uviesť, že sa majú hodnoty nasčítavať zoradené podľa hodnoty škody (*order by celk\_skoda\_kc desc*). Potom stačí, keď si vyberieme riadok s ohodnotnením *rn=1000*, kde je nasčítaných TOP 1000 škôd.
   2. Druhý variant je po vytvorení ohodnotení škôd využitie agregačnej funkcie *sum*, ktorou zrátame po odfiltrovaní nepotrebných záznamov (cez *where* podmienku) sumu všetkých zostávajúcich hodnôt stĺpca so s hodnotou škody.

Náklady oboch variantov sú rovnaké a ich skript je zobrazený na obrázku 18.

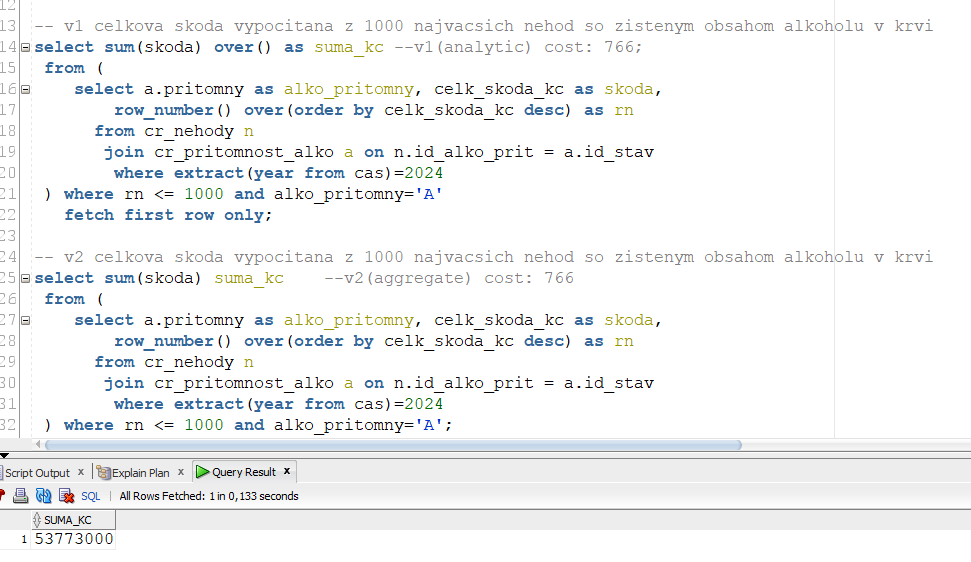


Obrázok 18 – dve možnosti spočítania sumy TOP 1000 najvyšších škôd roku 2024

1. Čitateľ (obrázok 19) – celková škoda s prítomnosťou alkoholu spomedzi TOP 1000 nehôd

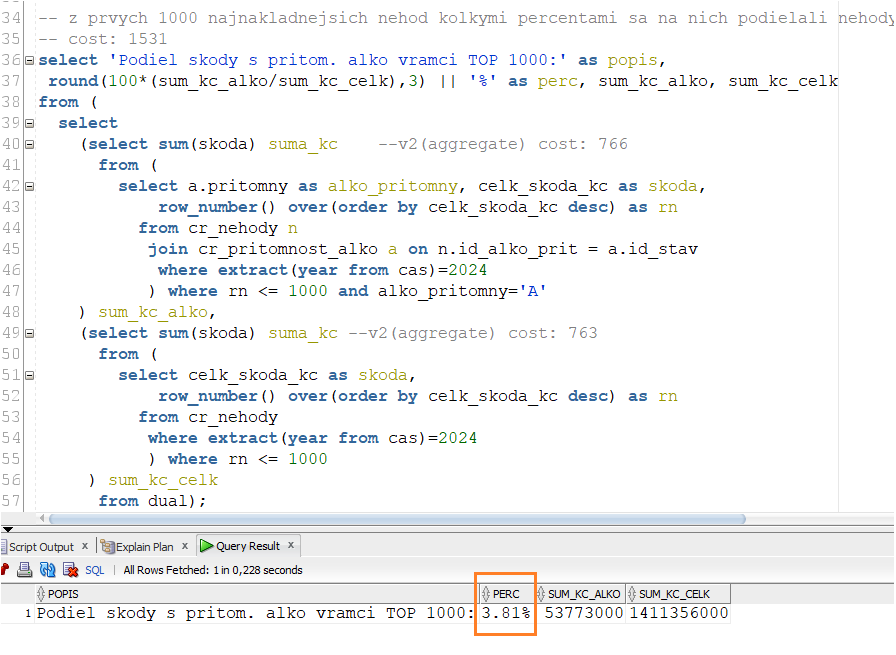
Najprv si zoradíme podľa veľkosti škody v Kč všetky nehody a priradíme im ohodnotenie pomocou analytickej funkcie *row\_number*. Okrem toho si musíme zapamätať či bol alkohol prítomný v krvi počas nehody, alebo nie. Následne, keď máme záznamy zoradené podľa ohodnotenia, vyberieme v podmienke *where* všetky záznamy, ktoré sú medzi prvými TOP 1000 a zároveň bol pri nehode alkohol prítomný v krvi. Potom už stačí len spočítať celkovú sumu škôd z vybraných záznamov a to môžeme dosiahnúť dvoma spôsobmi:

* 1. Využitie analytickej funkcie sum, v ktorej neuvedieme podmienku triedenia, čím zabránime využitiu rolling sum, ale namiesto toho sa sčíta suma zo všetkých záznamov a priradí sa ku každému záznamu. Potom nám stačí pomocou klauzuly *fetch* zobrať ľubovoľný, napr. prvý riadok, v ktorom získame hľadanú sumu.
  2. Druhý(jednoduchší) spôsob je využitie agregačnej funkcie *sum* bez definovania skupiny pre všetky vybrané záznamy.



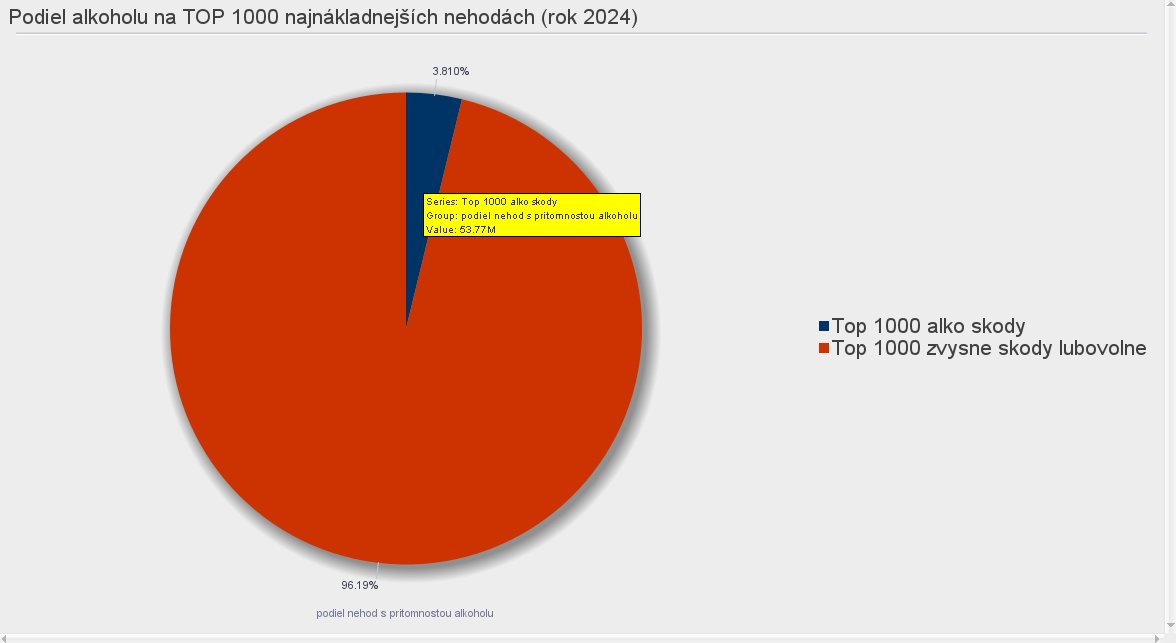
Obrázok 19 - dve možnosti spočítania sumy škôd s obsahom alkoholu spomedzi TOP 1000 roku 2024

Po vytvorení príkazov *select* pre čitateľ aj menovateľ stačí napísať *select* podľa vzoru na obrázku 17. Príklad výsledného skriptu je na obrázku 20. Môžeme vidieť, že na TOP 1000 nehodách podľa veľkosti vyčíslených škôd mali škody s prítomnosťou alkoholu v krvi 3,81% podiel na sumárnej škode.



Obrázok 20 – výsledný skript pre získanie perc. podielu škôd s prít. alkoholu na TOP 1000 škodách

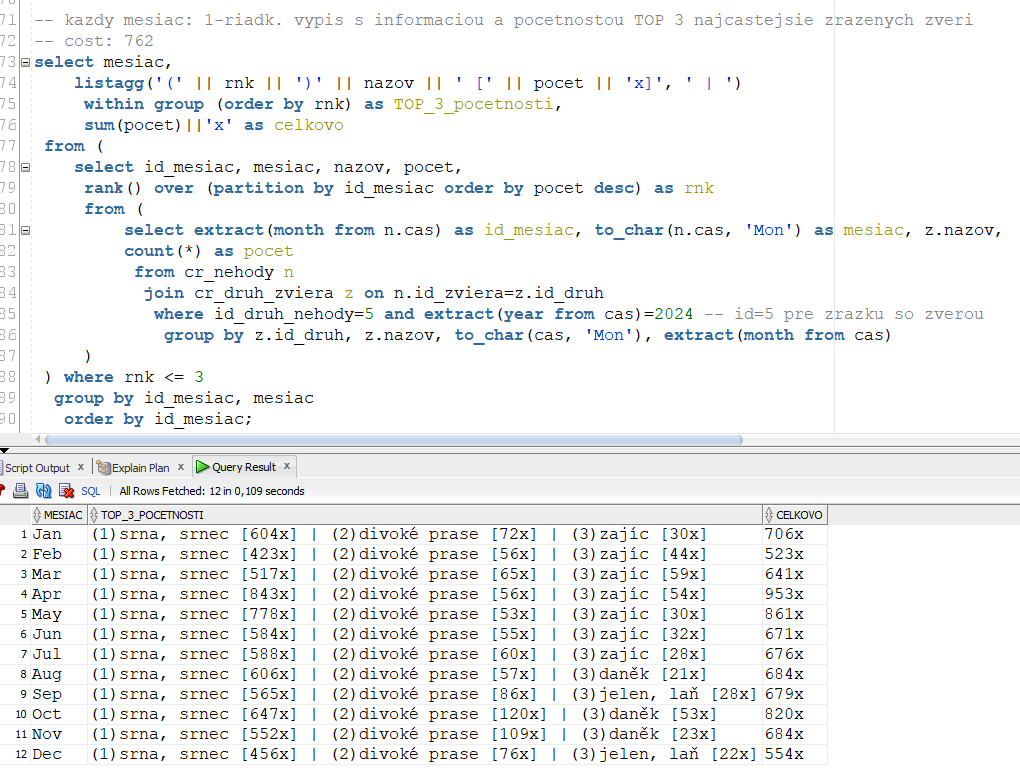
Pre lepšiu predstavu sme vytvorili aj report (obrázok 21) podielu alkoholu na TOP 1000 najnákladnejších nehodách 2024 v programe *SqlDeveloper*. Výsek vyfarbený na modro ukazuje nehody spomedzi TOP 1000, kde bol prítomný alkohol v krvi.



Obrázok 21 – report s podielom nehôd s prítomnosťou alkoholu k ostatným spomedzi TOP 1000 podľa najvyšších nákladov v Kč z roku 2024

## Mesačne najčastejšie zrazené zvieratá v roku 2024

Často dochádza k zrážkam so zverou. Vypíšeme si do jedného riadku pre každý mesiac informáciu o troch najviac zrazených zvieratách v danom mesiaci a k tomu uvedemie informáciu celkového počtu zrážok v mesiaci TOP 3 zvierat. Podľa obrázku 22 je zrejmé, že všeobecne v roku 2024 dochádzalo k najviac zrážkam so srnami, diviakmi a zajacmi. Taktiež, podľa obrázku 22, najviac TOP 3 zrazených zvierat bolo v apríli a to skoro až 1000, čo je o vyše 80% viac než vo februári. Využívame tu agregačnú funkciu *listagg*, ktorou agregujeme do jedného riadku všetky vybrané záznamy pre konkrétny mesiac a vypisujeme pomocou nej nami zvolené údaje, t.j. umiestnenie podľa počtu incidentov so zverou, meno zvieraťa a počty zrazení.



Obrázok 22 – výpis troch najčastejšie zrazených zvierat pre každý mesiac v roku 2024

## 2,5 mesačný kĺzavý medián počtu zranených pre rok 2024 (s frekvenciou polmesiac)

Poslednou informáciou, ktorú budeme zisťovať, je priebeh kĺzavého mediánu počtu zranených (ľahko zranených + ťažko zranených) pre rok 2024 v časovom rade s frekvenciou polmesiac. Medián budeme počítať z piatich hodnôt, t.j. 5 počtov zranených zodpovedajúcich im prislúchajúcim polmesiacom. Skript rozdelíme do 4 krokov tak, ako to ukazuje obrázok 23.



Obrázok 23 – skript pre 2,5 mesačný kĺzavý median počtu zranených pre rok 2024

* **Krok 1** – V prvom kroku si vytvoríme mesiace pre rok 2024 tak, že mesiac bude reprezentovaný jeho prvým dňom. Budeme využívať klauzulu *with*, aby sme sa mohli nasledovne odkazovať na vytvorenú štruktúru a mať pritom prehľadnejší príkaz *select*.
* **Krok 2** – V druhom kroku si vytvoríme polmesiace, ktoré získame tak, že k začiatku mesiaca pripočítame polovicu dní toho mesiaca s prípadným zaokruhlením dní nadol.
* **Krok 3** – V tretiom kroku každému záznamu o nehode musíme priradiť polmesiac, ku ktorému prislúcha a potom pre každý polmesiac spočítať celkovú sumu počtu zranených pri nehodách, ktoré k danému polmesiacu prislúchajú.

Príslušnosť nehody k polmesiacu dosiahneme nasledovne:

a) spojíme záznamy tabuľky nehôd so záznamami štruktúry obsahujúcej polmesiace na základe mesiaca. Z toho vyplýva, že každému záznamu o nehode vzniknú 2 spojenia: s polmesiacom začínajúcim nový mesiac a polmesiacom, ktorý je uprostred mesiaca.

b) teraz je potrebné vybrať 1 z 2 spojení pre každý záznam. Budeme vyberať záznam s tým polmesiacom, po ktorého začiatku nehoda buď:

1) bezprostredne nasleduje (nemôže predchádzať, aby bolo jasne vymedzené, kam patrí)

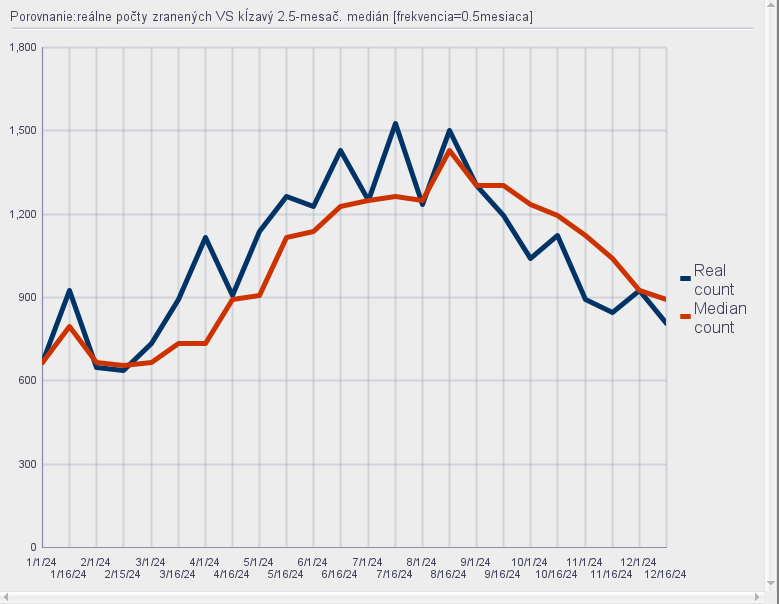
2) alebo sa deň nehody zhoduje so začiatkom polmesiaca

Na priradenie polmesiaca k nehode využijeme analytickú funkciu *row\_number*, ktorá ohodnotí vhodnosť priradenia daného polmesiaca pre nehodu. Hodnotenie bude len v rámci jednej nehody, t.j. ohodnotenie bude nadobúdať hodnoty *1* a *2*. My priradíme ten polmesiac, ktorý dostane ohodnotenie *1*. Vhodnosť priradenia polmesiaca určujeme nasledovne:

1. Ak je deň nehody väčší alebo rovný začiatku polmesiaca, potom vhodnosť priradenia predstavuje rozdiel počtu dní dňa nehody a dňa začiatku polmesiaca
2. Pokiaľ je deň nehody pred dňom začiatku polmesiaca, priradíme mu hodnotu 10000, čo je dostatočne veľká hodnota na to, aby dostala menšiu prioritu, pretože taký veľký počet dní určite nikdy nebude medzi dňom nehody a dňom začiatku polmesiaca

* **Krok 4** – Na začiatku posledného kroku už máme štruktúru, ktorá obsahuje záznamy s polmesiacom a príslúchajúcim počtom zranení pre tento polmesiac. Teraz potrebujeme ku každému polmesiacu nájsť (pokiaľ existuje v záznamoch) chronologicky prvý, druhý, tretí a štvrtý predchádzajúci polmesiac a z neho vytiahnuť informáciu o počte zranených v tom polmesiaci. Na získanie týchto počtov využijeme analytickú funkciu *lag*. Počty zranených z predchádzajúcich polmesiacov pripojíme k základnej množine zjednotením s povolením duplicít a každému záznamu priradíme polmesiac, pre ktorý je hľadaný počet viazaný. Nakoniec spočítame hodnoty mediánov cez agregačnú funkciu *median* pre skupiny definované polmesiacom.

Výsledky vypočítaných mediánov si môžeme zobraziť v programe *SqlDeveloper* definovaním vlastného reportu (obrázok 24). Modrá krivka ukazuje skutočné počty zranených pre daný polmesiac a červená zobrazuje výpočítaný kĺzavý medián, ktorý vyhladzuje priebeh modrej krivky. Môžeme vidieť, že počet zranených počas letných mesiacov je minimálne dvakrát väčší než počty zranených na začiatku roka.



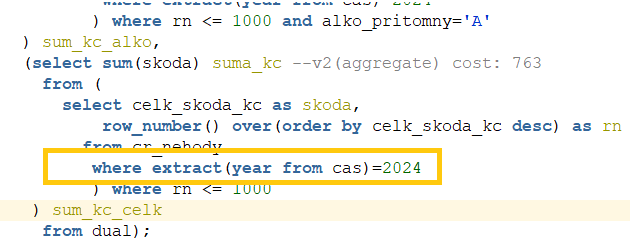
Obrázok 24 – report na porovnanie počtov vyjadrených kĺzavým mediánom a skutočným počtom zranených v priebehu roka 2024

# Optimalizácia výkonnosti

Skúsme sa teraz pozrieť či by sme mohli niektorý z predošlých dotazov rýchlostne zlepšiť vytvorením vhodného indexu.

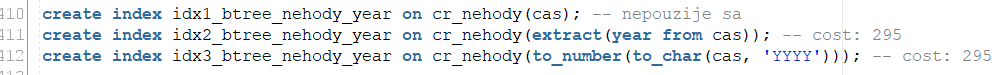
## Index pre skript získania percentuálneho podielu škôd nehôd s prítomnosťou alkoholu v TOP 1000 najvyššie ocenených škodách

Skript z obrázku 18 obsahuje podmienku *where*, v ktorej vyberáme záznamy podľa hľadaného roku.



Obrázok 25 – ukážka filtrovacej podmienky príkazu select pre získanie percentuálneho podielu škôd z obrázku 18

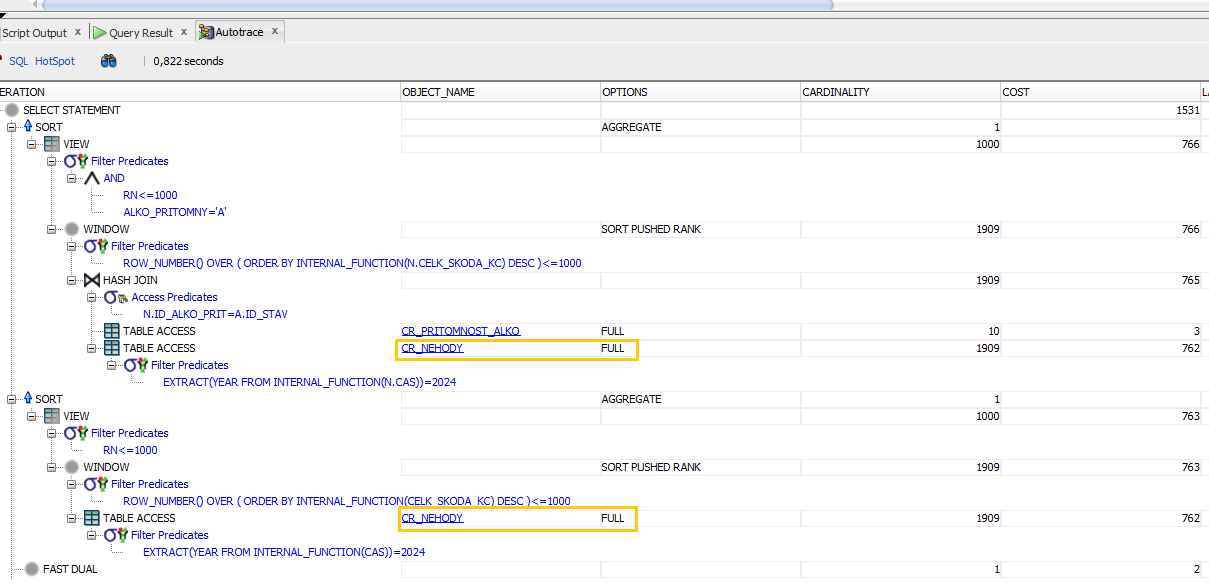
Vyskúšame tri varianty:



Obrázok 26 - varianty pre index na získanie roku

1. Index nad stĺpcom *cas* (index *idx1\_btree\_nehody\_year* z obrázku 26)

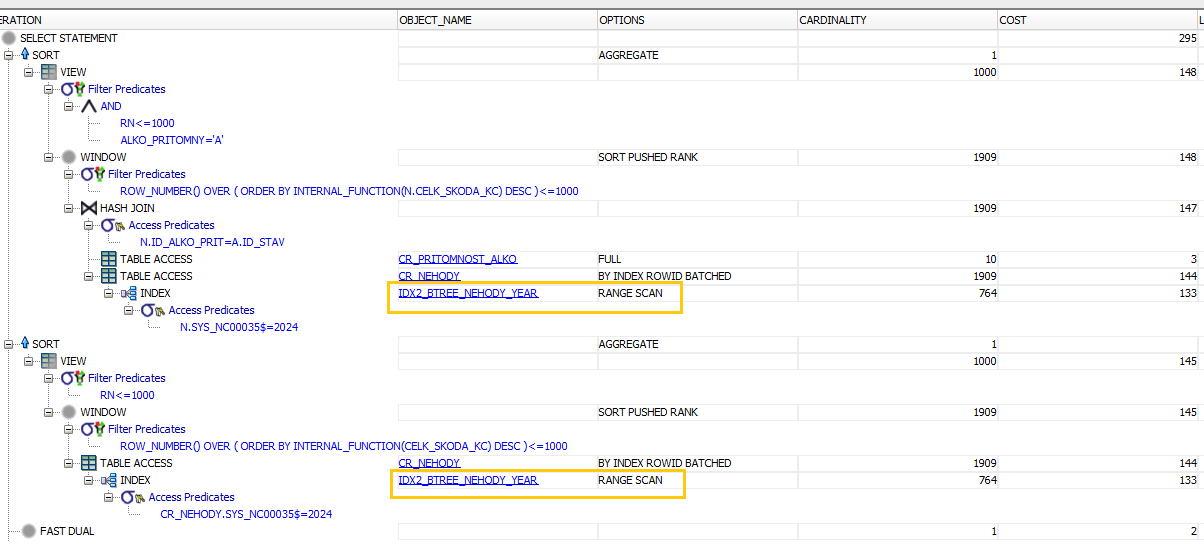
* z časovej pečiatky potrebujeme získať rok, preto nemôžeme použiť index pre stĺpec *cas*
* takýto index vyjadruje index na celkovú hodnotu časovej pečiatky, preto sa nepoužije (obrázok 27), lebo filter záznamov sa robí až na základe výsledku funkcie spracujúcej čas, ktorý sa porovná s hľadanou hodnotou



Obrázok 27 – vytvorenie indexu nad stĺpom cas nemá vplyv na kritérium filtrovania podmienky podľa hodnoty roku časovej pečiatky po spracovaní funkciou extract

1. Funkcionálny index nad výsledkom funkcie *extract(year from cas)* (index *idx2\_btree\_nehody\_year* z obrázku 26)

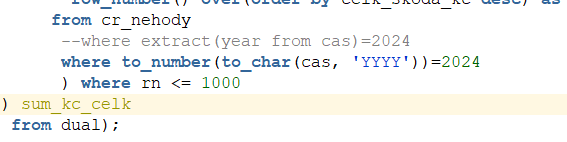
* Vytvoríme klasický index, ktorý je implementovaný ako údajová štruktúra B+ strom
* celkové náklady skriptu sa z nákladov 1531 znížili na 295 jednotiek, pretože sa index použil, dokonca sa použil dvakrát – v dvoch vnorených príkazoch *select* (obrázok 28)



Obrázok 28 – využitie funkcionálneho indexu idx2\_btree\_nehody\_year vo where podmienke extract(year from cas)

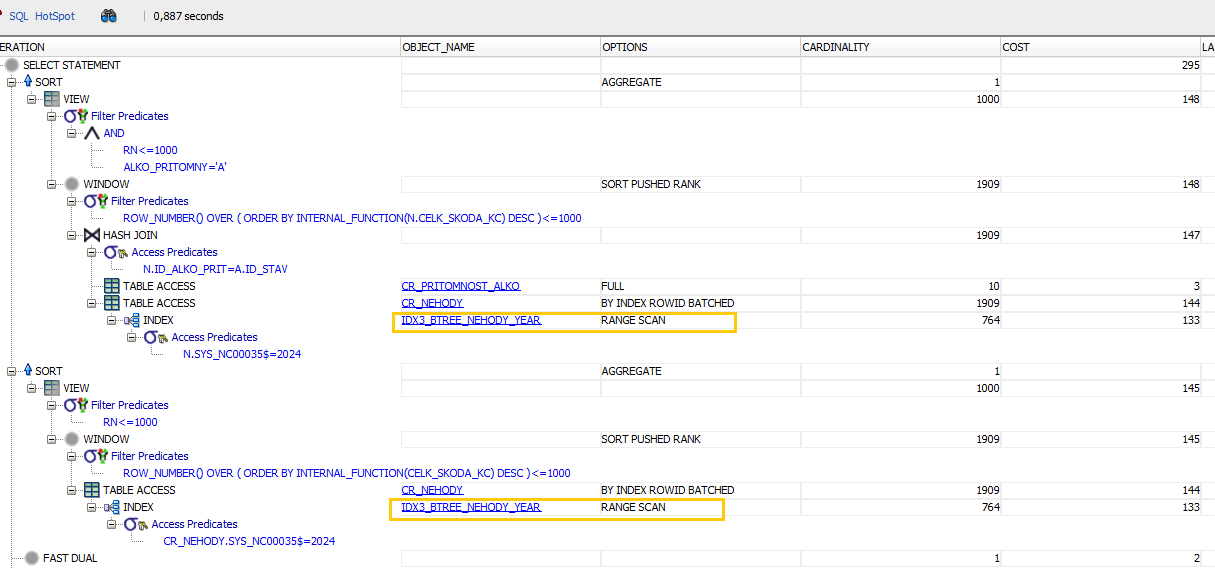
1. Funkcionálny index nad výsledkom funkcie *to\_number(to\_char(cas, ‘YYYY’))* (index *idx3\_btree\_nehody\_year* z obrázku 26)

* Musíme si dať pozor, aby sme výsledok funkcie *to\_char(cas, ‘YYYY’)* konvertovali na číslo pre správne a dobré fungovanie
* Taktiež sme vytvorili B+ strom index
* Celkové náklady sú rovnaké ako pre variant z predchádzajúceho bodu b) (výsledok použitia variantu c) na obrázku 30)
* Ak by sme tento index chceli použiť, museli by sme upraviť podmienku *where* na tvar, v akom je definovaný index, t.j. *where to\_char(cas, ‘YYYY’)=2024* (príklad na obrázku 29)



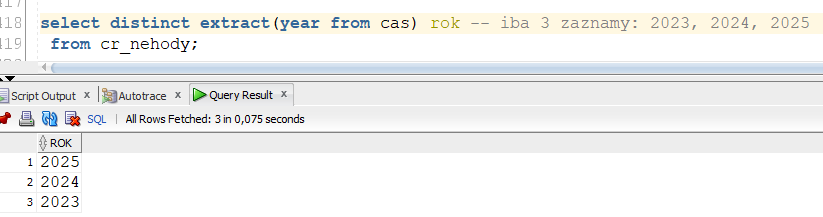
Obrázok 29 – zámena podmienky where kvôli zmene funkcionálneho indexu

* Variant b) ak c) fungujú, lebo je splnená podmienka, že funkcia je deterministická pre získanie roku



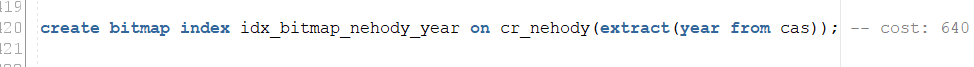
Obrázok 30 - využitie funkcionálneho indexu idx3\_btree\_nehody\_year vo where podmienke to\_number(to\_char(cas, ‘YYYY’))

Keď si skúsime vypísať všetky možné hodnoty výsledkov funkcie *extract(year from cas)*, dostaneme len toľko rôznych hodnôt koľko rôznych rokov v hodnote stĺpca *cas* nájdeme. My sme importovali iba dáta pre rok 2023, 2024 a 2025 (obrázok 31).



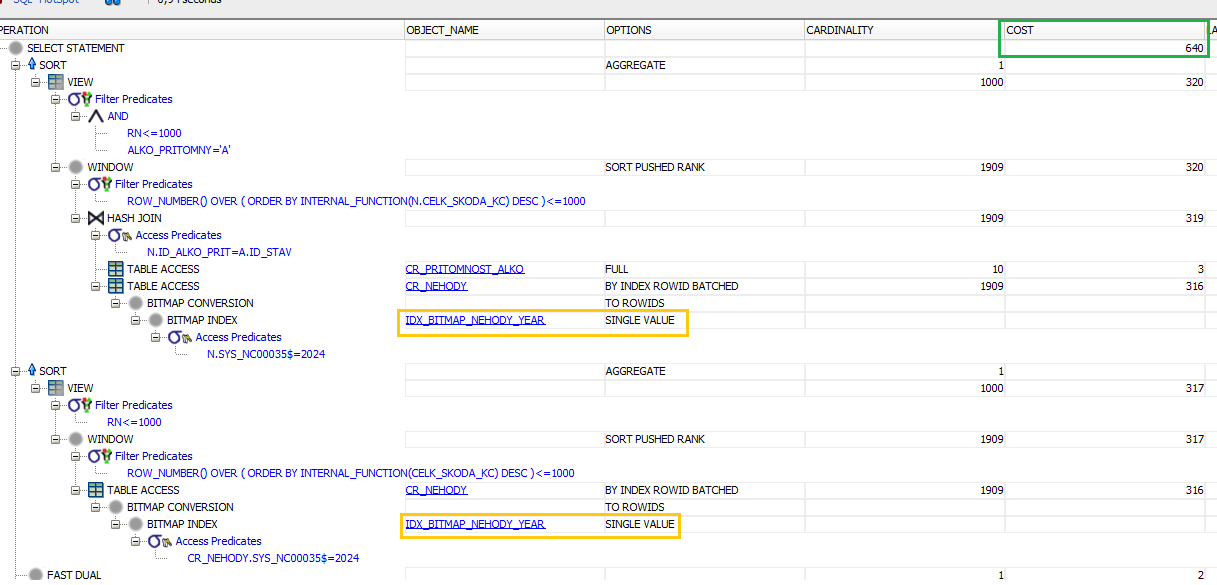
Obrázok 31 – všetky možné hodnoty roku pre naimportované dáta

Ako vidíme, máme len 3 možnosti a celkový počet záznamov tabuľky s nehodami podľa obrázku 7 okolo 190000. To znamená, že selektivita stĺpca je 100\*(3/190000), čo je približne 0.1% a mohli by sme skúsiť použiť bitmapový index, ktorý je ukázaný na obrázku 32.



Obrázok 32 – bitmapový funkcionálny index pre získanie roku z časovej pečiatky premennej cas

Celkové náklady s využitím bitmapového indexu sa ukázali byť vyššie (obrázok 33). Nevýhodou využitia bitmapového indexu v tomto kontexte by bolo, že by do tabuľky nehôd postupne rokmi mohli pribúdať záznamy s časovými pečiatkami z nasledujúcich rokov. Vtedy by bolo nutné vykonať nad indexom rebuild, aby sa pridal do indexu nový stĺpec (nová hodnota roku, ktorú môže index nadobúdať). Preto je v tomto kontexte pre optimalizáciu rýchlosti vykonania najlepšie využiť B+ strom funkcionálny index variantu b) alebo c). Väčšina doteraz ukázaných príkladov obsahovala filter na konkrétny rok, takže by sme tým zrýchlili vykonanie všetkých týchto skriptov.

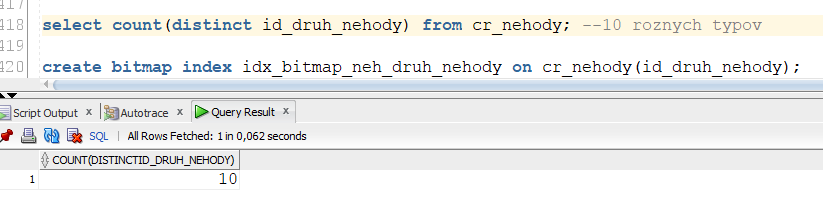


Obrázok 33 – využitie bitmapového funkcionálneho indexu pre zrýchlenie filtrovania záznamov podľa roku

## Index pre výpis TOP 3 najčastejšie zrazených zvierat za každý mesiac

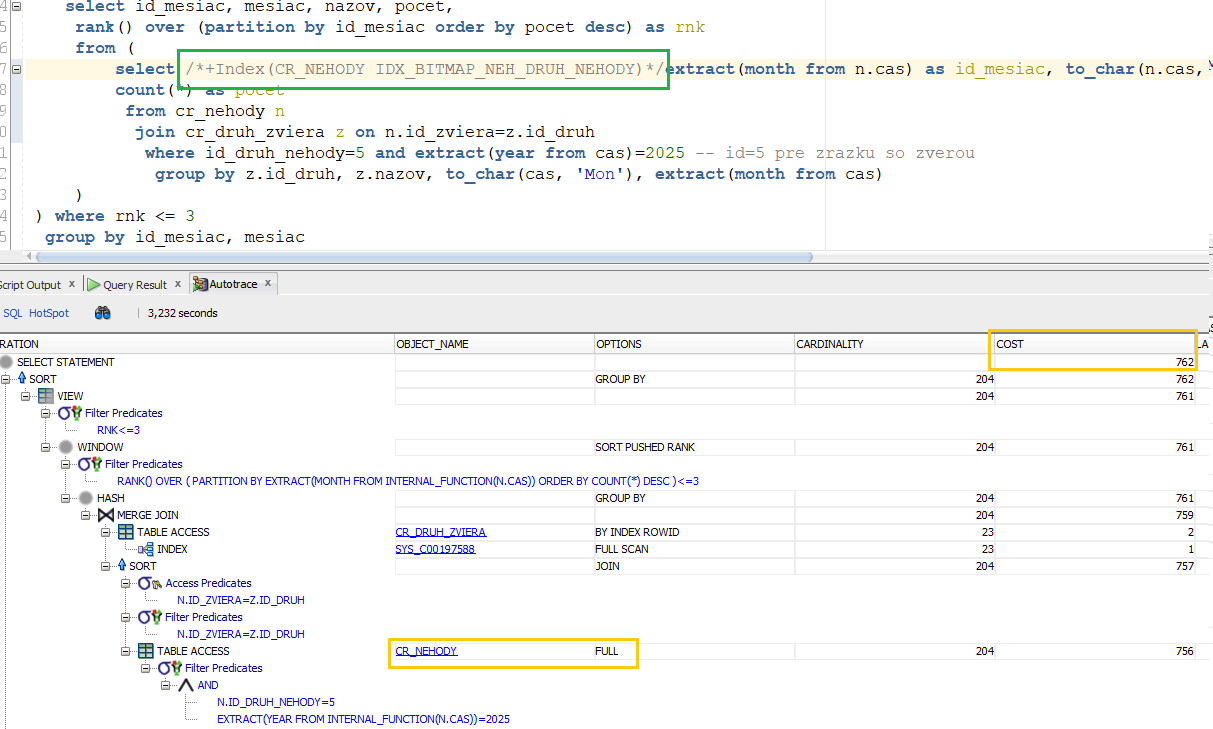
V tomto skripte (obrázok 22) sa opäť zameriame iba na podmienku *where*, lebo nie sme schopní index definovať pre všetky hodnoty, ktoré sa v príkaze *select* používajú, pretože dochádza k spájaniu tabuliek a vyberanie atribútu z pripojenej tabuľky. V podmienke *where* filtrujeme záznamy podľa typu nehody a roku, v ktorom k nehode došlo.

Vyskúšame najprv použiť bitmapový index *idx\_bitmap\_neh\_druh\_nehody* iba na druh nehody (obrázok 34), pretože môže nadobúdať iba 10 jedinečných hodnôt, čo znamená, že selektivita tohto stĺpca v tabuľke s nehodami sa blíži k nule a taktiež nemienime pridávať nové typy nehôd, čo je pre tento typ indexu žiaduce.



Obrázok 34 - bitmapový index idx\_bitmap\_neh\_druh\_nehody

Po vykonaní plánu však zistíme, že sa bitmapový index nepoužil a to ani vtedy, keď sme príkazu *select* pridali HINT (obrázok 35).



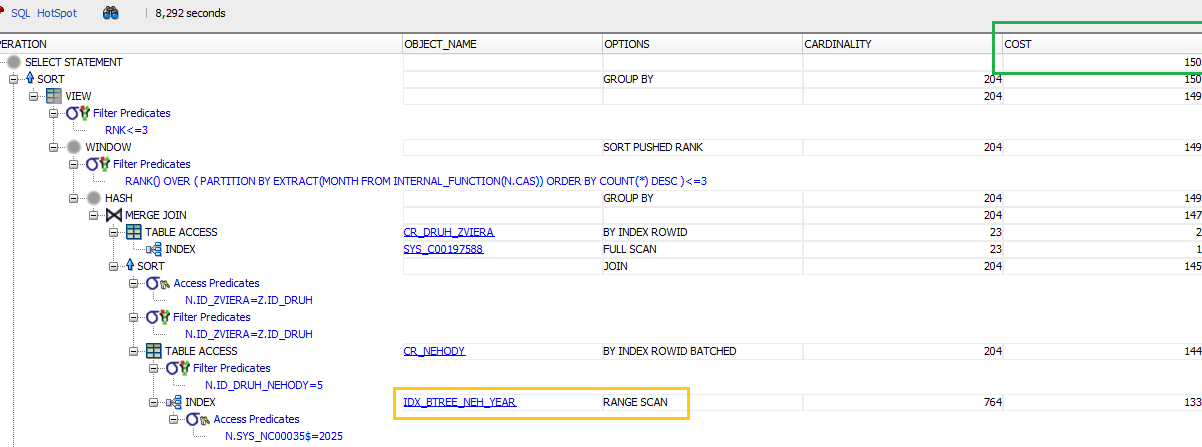
Obrázok 35 – HINT pre použitie bitmapového indexu

Teraz skúsme vytvoriť klasický B+ strom funkcionálny index nad získaním roku zo stĺpca *cas* (obrázok 36).



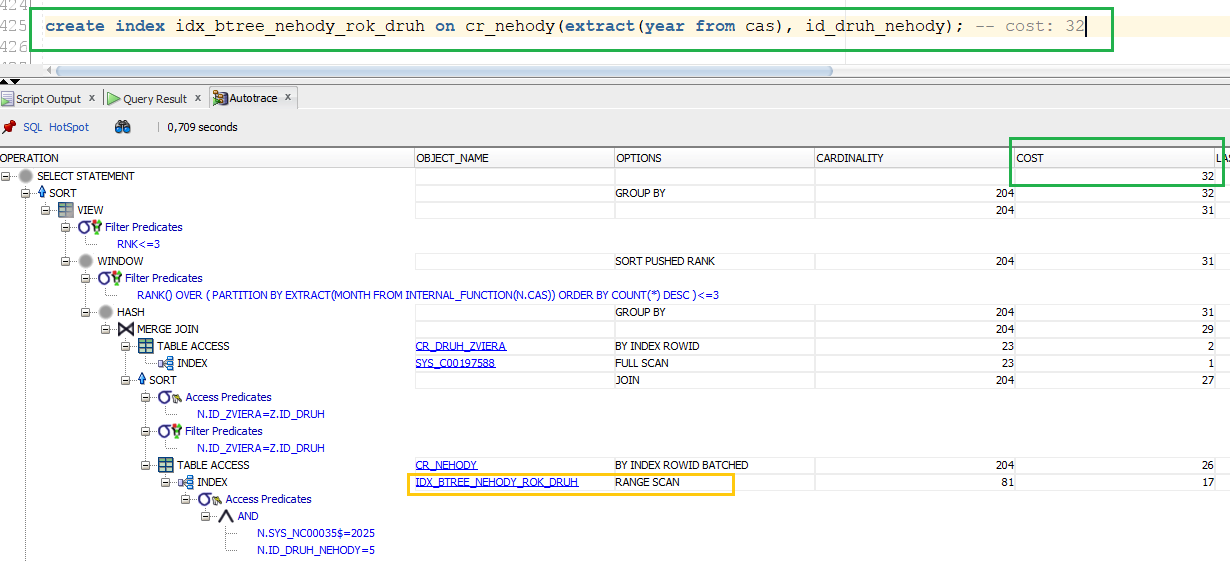
Obrázok 36 – funkcionálny index na získanie roku zo stĺpca cas

Tento index sa už využil vo vykonanom pláne a rapídne znížil náklady zo 762 jednotiek na 150 (obrázok 37).



Obrázok 37 – využitie funkionálneho indexu idx\_btree\_neh\_year

Môžeme sa teraz ešte pokúsiť znížiť dobu vykonania vytvorením kompozitného funkcionálneho indexu. Index sa bude skladať z extrahovaného roku z časovej pečiatky a druhu nehody (obrázok 38). Vďaka tomu indexu sa nám podarilo zredukovať náklady na 32 jednotiek z pôvodných 762.



Obrázok 38 – aplikácia kompozitného funkcionálneho indexu pre extrahovaný rok a druh nehody

## Optimalizácia spojenia tabuliek nehôd a vozidiel

Na záver vyskúšame zrýchlenie príkazu *select* na základe optimalizácie spojenia tabuliek nehôd a vozidiel asociovaných s nehodami.

# Záver