DBS Zadanie 3

Dokumentácia

Rodion Burmistrov

V rámci Zadania číslo 3 bolo potrebné realizovať databázu pre RPG combat hru, navrhnutú v predošlom zadaní. V rámci realizácie sú zarátane tak isto aj rôzne SQL procedúry, indexy, jedna funkcia a pohľady. Ďalej v dokumente je popísaný navrhnutý systém, jeho entity a ich vzťahy, vrátane požadovaných procesov.

**Zmeny Návrhu Databázy:**

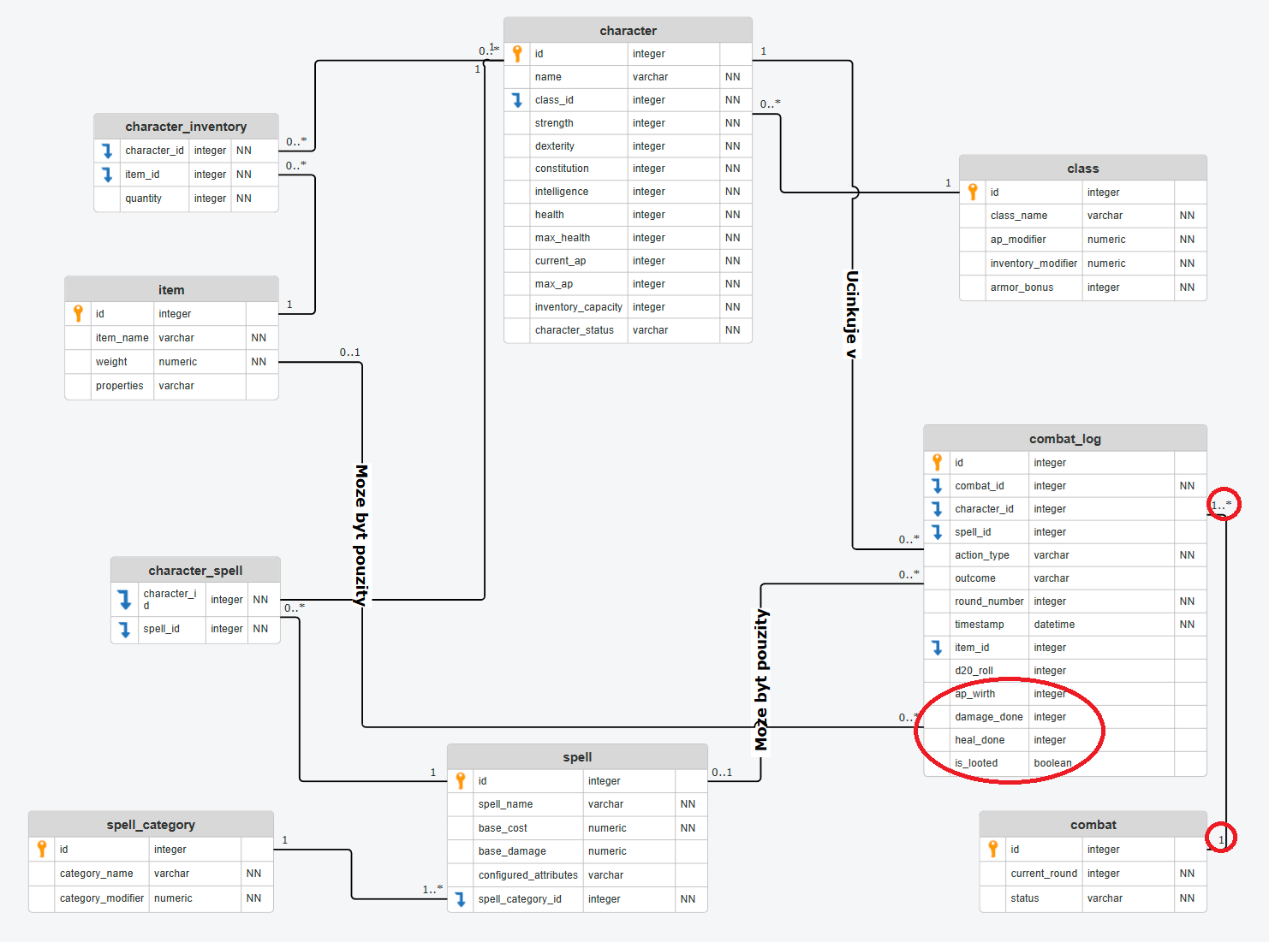
V priebehu kontrolovania zadania číslo 2 sa zistilo, že pôvodný Návrh nebol úplne úspešný. Zmeny boli nasledujúce:

1. **Popletený vzťah combat\_log – combat.**

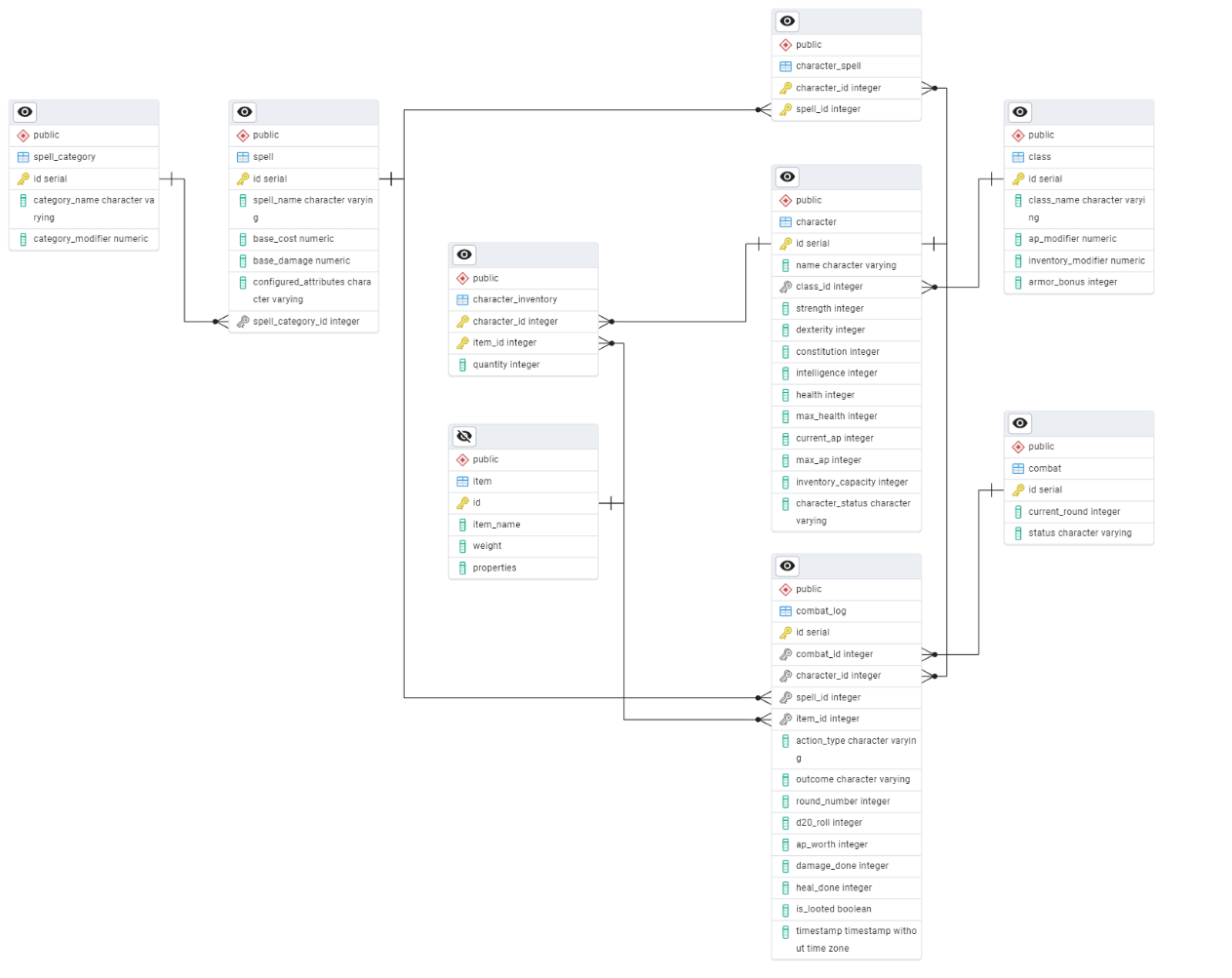
Povodne ten vzťah sa čítal ako „combat ma iba jeden combat\_log, a combat\_log ma viac combatov“, čo už bola chyba, vzniknutá z nepozornosti pri navrhovaní diagramu, aj kde v popise sa myslelo správne, teda „combat ma viac combat\_logov, ale combat\_log patri pravé jednému combatu“.

1. **Pridane atribúty do combat\_logu.**

Počas prezentácie, ako aj počas tvorby databázy a jej procedúr bolo zistene, že návrh tabuľky combat\_log nebol dokonalý. Teda chceli by sme vidieť viac podrobne informácie, ktoré sa deju počas bitky. Ako napríklad koľko energie bolo minúte na kúzlo, aký pôsobilo dane kúzlo damage, alebo aký to kúzlo pôsobilo heal. Tak vznikli atribúty *ap\_worth, damage\_done, heal\_done* typov INTEGER. Tak isto na implementáciu lootingu počas bitky bolo rozhodnuté pridať atribút is\_looted typu BOOLEAN, pre jednoduchšie sledovanie stavu itemov v bitke. Teda ci je zdvihnutý (TRUE), alebo nie (FALSE), povodne cely čas je tato hodnota FALSE, aj keď nejde o combat\_log, ktorý by obsahoval nejaký loot. Na Obrázku 1 je znázornený opravený Relačný Diagram zo zadania 2, a na Obrázku 2 je znázornený EDR diagram implementovanej databázy, vytvorený v pgAdmin 4.



Obrázok 1



Obrázok 2

**Logicko-fyzické mapovanie modelu**

Ďalej databázy sa nedotkli žiadne zmeny, logicko-fyzicke mapovanie projektu zostalo povodne zo zadanie 2. Iba v krátkosti si občerstvime spomienky:

„Postavy“ (tabuľka **character**) patria do určitej triedy (**class**), pričom každá trieda môže mať mnoho postáv a naopak. Postavy môžu používať rôzne kuzla (**spell**), ktoré sú zoskupené do kategórií (**spell\_category**), a zároveň nosiť predmety (**item**). Medzi postavami a kuzlami/predmetmi existujú pivotne tabuľky (**character\_spell**, **character\_inventory**) na zaznamenanie toho, kto ktoré schopnosti a vybavenie ovláda a v akom množstve.

Každá postava sa môže zúčastniť bojov (**combat**), kde sa v **combat\_log** sleduje priebeh jednotlivých kôl: kto, kedy a aký typ akcie vykonal (útok, použitie kuzla, alebo predmetu), s akým výsledkom (hit/miss), koľko poškodenia/spätnej liečby spôsobí a s akým hodom d20.

Nižšie stručne zhrnuté významy jednotlivých tabuliek:

* **class**: ukladá ID, názov triedy (napr. „Mág“), modifikátor akčných bodov (ap\_modifier), kapacitu inventára (inventory\_modifier) a bonus k brneniu (armor\_bonus).
* **character**: reprezentuje postavy; okrem ID a mena obsahuje FK na **class**, štatistiky (strength, dexterity, constitution, intelligence), aktuálne/maximálne zdravie a akčné body, kapacitu inventaru a stav postavy (character\_status).
* **spell\_category**: triedi kuzla do skupín (napr. Fire Magic alebo Sword Arts) a určuje globálny modifikátor ceny čarov (category\_modifier).
* **spell**: obsahuje ID, názov, základnú cenu (base\_cost), základnú hodnotu efektu (base\_damage), atribút, ktorý špecifikuje, z akého štatistiky sa pripočítava výkon (configured\_attributes), a FK na **spell\_category**.
* **item**: zoznam vybavenia – každé s ID, názvom, váhou a vlastnosťami (properties), podľa ktorých sa určuje efekt (napr. liečivý, ak to je Healing Potion, alebo zbraňové poškodenie, ak to je Sword).
* **character\_spell**: pivotna tabuľka many-to-many medzi postavami a čarami, ktorá obsahuje kúzla, ktoré môže postava použiť.
* **character\_inventory**: pivotna tabuľka many-to-many medzi postavami a predmetmi, obsahuje aj quantity, ktoré udáva, koľko daného predmetu postava nesie.
* **combat**: eviduje jednotlivé boje s unikátnym ID, aktuálnym číslom kola (current\_round) a stavom bitky (napr. „waiting“, „active“, „ended“).
* **combat\_log**: zaznamenáva každú akciu počas boja – s FK na **combat**, **character**, voliteľne **spell** alebo **item**, typ akcie (action\_type), výsledok (outcome), číslo kola, hod d20, spôsobené poškodenie alebo liečenie, minúte AP body, stav lootov a čas akcie (timestamp).

**Procesne toky a prototypy postupov**

1. **sp\_cast\_spell a f\_effective\_spell\_cost**

Proces posielanie kúzla, počas bitky prebieha nasledovne. Najprv sa overí, či je boj aktívny, či sú Caster a Target su jeho účastníkmi a či Caster ovláda dané kúzlo. Potom sa vypočíta efektívna cena kúzla pomocou funkcie **f\_effective\_spell\_cost**, ktorá berie základnú cenu spellu, modifikátor kategórie, relevantný atribút Castera a modifikátor akčných bodov klasy (AP). A použije vzorec v\_base\_cost \* v\_category\_modifier \* (1 - (v\_attribute\_value::NUMERIC / 100)) \* v\_ap\_modifie. Ak kúzliaci nemá dostatok AP, pokus zlyhá a jeho ťah sa skončí. Ak áno, AP sa odpočítajú a nasleduje hod d20 a počítanie útoku.

Úspešnosť útoku sa určuje hodením kockou d20, ku ktorej sa pripočíta bonus z atribútu Castera, a výsledok sa porovná s cieľovým brnením (AC), vypočítaným ako 10 + (dexterity / 2) + class.armor\_bonus. Ak súčet hodu d20 a príslušného atribútu nepresiahne AC, kúzlo minie cieľ a ťah sa ukonči. Pri zásahu sa vypočíta poškodenie vzorcom v\_base\_damage \* (1 + v\_attribute\_value / 20.0), ďalej sa aplikuje na cieľ a skontroluje sa, či jeho zdravie kleslo na nulu alebo menej – vtedy sa boj ukončí. Celý proces je zaznamenaný do combat\_logu hry. Sekvenčný diagram, znázornený na Obrázku 3, predstavuje tok daného procesu a principiálne sa nelíši s tým, ktorý bol preukázaný v Zadaní 2

**Изображение выглядит как текст, снимок экрана, Параллельный, документ

Контент, сгенерированный ИИ, может содержать ошибки.**

Obrázok 3

1. **sp\_rest\_character**

Táto procedúra umožňuje postave oddýchnuť a obnoviť zdravie a akčné body (AP), ale iba ak nie je v boji. A to tak, ze skontroluje, či má postava stav character\_status = 'Out of Battle'. Ak áno, aktualizuje jej zdravie (health) a AP (current\_ap) na maximálne hodnoty (max\_health a max\_ap) v databáze. Ak je postava v boji, procedúra skončí s upozornením, že oddych počas bitky nie je možný. Sekvenčný diagram, znázornený na Obrázku 4, predstavuje tok daného procesu a principiálne sa nelíši s tým, ktorý bol preukázaný v Zadaní 2

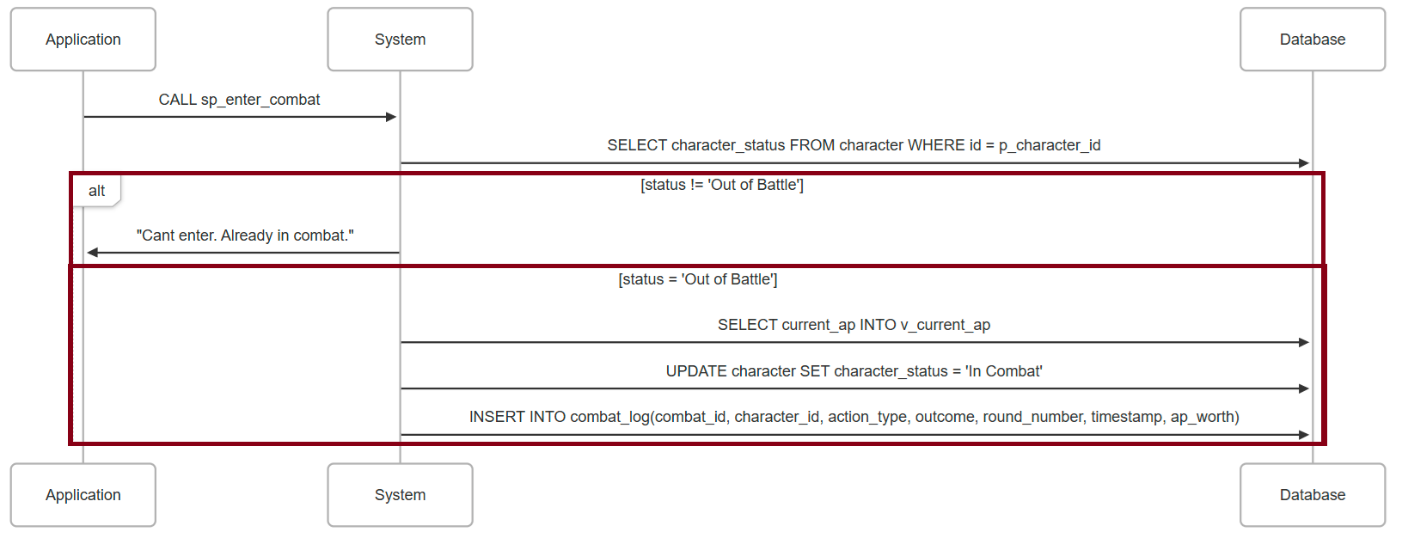
**Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.**

Obrázok 4

1. **sp\_enter\_combat**

Procedúra spracováva vstup postavy do boja. Overí, či je postava mimo boja (character\_status = 'Out of Battle'), inak vyhodí chybu. Ak je podmienka splnená, získa aktuálne AP postavy (current\_ap), zmení jej stav na ‘In Combat’ a vytvorí záznam v combat\_log s akciou enter combat, označenou ako úspešná, spolu s hodnotou AP pri vstupe. Nezahŕňa vytvorenie nového boja ani čakanie na druhého hráča. Oproti pôvodnému návrhu je procedúra jednoduchšia, nakoľko tu sa za predpoklad berie to, že jednotlivý Combat, na ktorý sa hrací pripoja už je vytvorený. Seqvencny diagram, znazorneny na Obrazku 5, predstavuje tok daného procesu.



Obrázok 5

1. **sp\_rest\_character**

Procedúra obnovuje kolo v aktívnom boji. Skontroluje, či je boj aktívny (status = 'active'). Ak áno, obnoví AP všetkých účastníkov na ich maximálne hodnoty (max\_ap), zvýši číslo kola (current\_round) o 1 a náhodne vyberie predmet z tabuľky item pomocou RANDOM(). Záznam o novom kole a vybranom predmete sa zapíše do combat\_log s akciou New Round: Item dropped. Sekvenčný diagram, znázornený na Obrázku 6, predstavuje tok daného procesu a principiálne sa nelíši s tým, ktorý bol preukázaný v Zadaní 2.

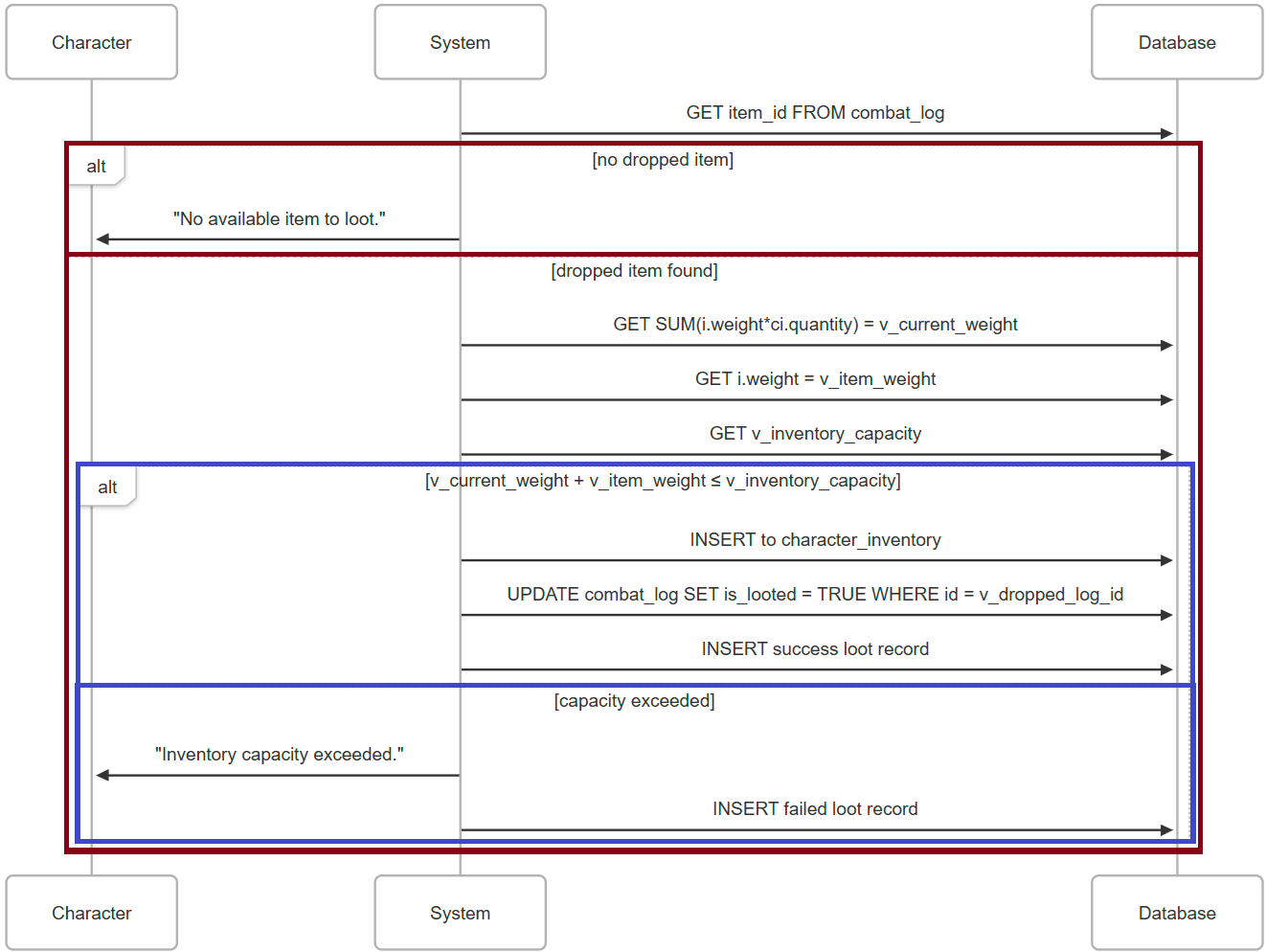
**Изображение выглядит как текст, линия, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.**

Obrázok 6

1. **sp\_loot\_item**

Táto procedúra umožňuje postave zobrať predmet z bojového poľa. Overí, či je boj aktívny (status = 'active') a či je predmet dostupný (is\_looted = FALSE). Skontroluje váhu predmetu a aktuálnu váhu inventára postavy proti jej kapacite. Ak miesto je volne, predmet sa pridá do character\_inventory, označí sa ako zozbieraný v combat\_log a zapíše sa úspešný výsledok. Ak kapacita nestačí, zapíše neúspešný pokus s chybovým hlásením. Sekvenčný diagram, znázornený na Obrázku 7, predstavuje tok daného procesu

****

Obrázok 7

**Zoznam vytvorených/navrhovaných indexov**

Indexy boli navrhované postupne počas tvorby a testovania celej databázy. Bral sa ohľad na najčastejšie používane atribúty, použite pri filtrácii dát cez SELECT WHERE/ORDER BY. Indexy sú znázornene na Obrázku 8 a hlavne boli teda:

Vyhľadávanie konkrétnej bitky (combat\_id) v combat\_loge.

Vyhľadávanie combat\_logov pre vybraného charactera\_id.

Kombinačne vyhľadávanie oboch, alebo ORDER BY podlá combat\_id/character\_id.

Vyhľadávanie toho, aké kúzla vie použiť konkrétna postava.

Vyhľadávanie stavu inventára konkrétnej postavy .

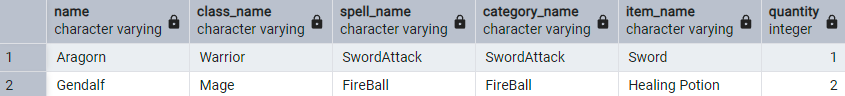
Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.

Obrázok 8

**Pokyny na načítanie vzorových údajov a vykonanie akceptačných testov.**

**TEST DATABASE\_CREATION**

Jednoducho zbehnúť DATABASE\_CREATION.sql v pgAdmine, aby vytvoriť databázu. Po načítaní môžeme aj skúsiť, ci to zbehlo v poriadku. A to použitím Querry testquerry.sql. Výpis by mal byt presným obsahom Obrázku 9 .

Obrázok 9

Ďalej zbehneme všetky zvyšne funkcie a procedúry.

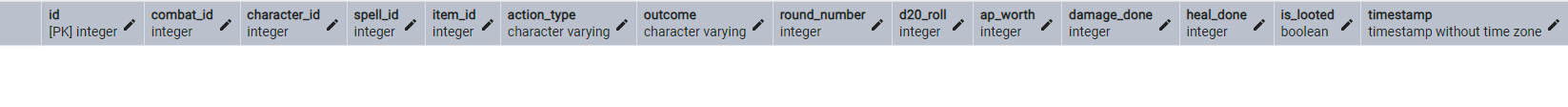
**Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.TEST sp\_enter\_combat**

Obrázok 10

Pouzijeme jednoduchy SELECT, aby preukazat prazdnost tabulky combat\_log.

SELECT \*

FROM combat\_log

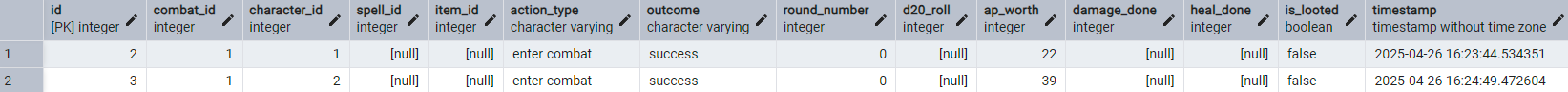
Obrázok 11

Ďalej zavoláme dve procedúry, aby hrací sa dostali do combatu.

CALL sp\_enter\_combat(1,1);

CALL sp\_enter\_combat(1,2)

Obrázok 12



Na ďalší test treba “aktivovať” combat, lebo mame round 0, teda čakáme kým lobby sa naplní. Keď sa to stane, očakáva sa zmena stavu combatu. Ale v našom prípade mali by sme ju zmeniť ručne, lebo všetky procedúry sú obmedzene danou aktiváciou. Pri každej procedúre bude písať niečo podobne:



Musíme teda napísať:

UPDATE combat SET status = 'active' where id = X (teda X=1, kedze testujeme bitku 1)

Poznamka: Dalsie combaty bude treba vytvarat rucne cez INSERT:

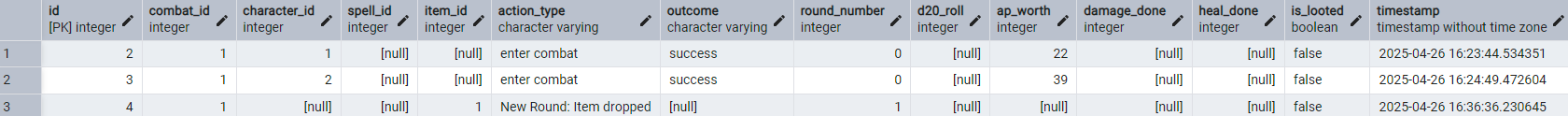
INSERT INTO combat (current\_round, status) VALUES (0, 'waiting');

**TEST sp\_reset\_round**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Obrázok 13

CALL sp\_reset\_round(1)

Obrázok 14

Jak bolo spomínane, spawne sa radnom Item na začiatku kola, zdvihne ho ten, kto rýchlejšie použije dalsiu procedúru.

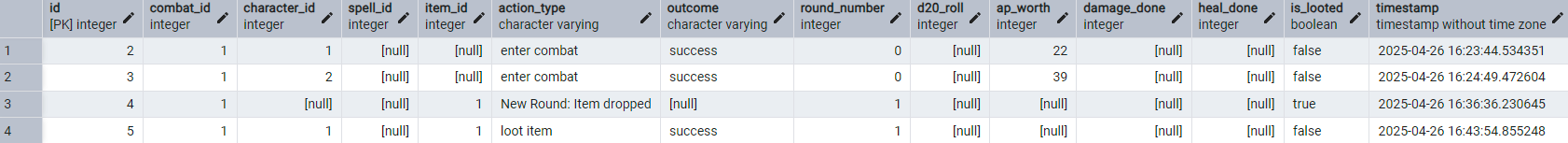
Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.**TEST sp\_loot\_item**

Obrázok 15

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

CALL sp\_loot\_item(1,1,item\_id)

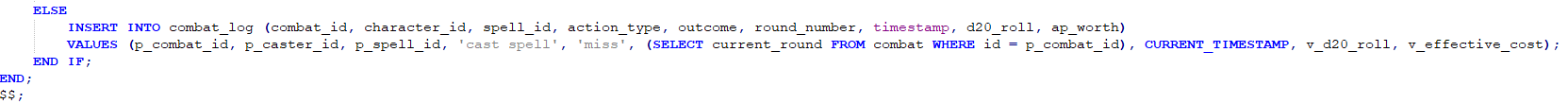
Obrázok 16

Ďalej sa môže uskutočniť bitka (akože vie sa uskutočniť aj pred lootingom, ale tym, že postupne chceme otestovať projekt, začneme ju teraz).

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Контент, сгенерированный ИИ, может содержать ошибки.**TEST sp\_cast\_spell**

Изображение выглядит как текст, снимок экрана, линия, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Obrázok 17

Tato procedúra tak isto zahŕňa aj funkciu počítania damage, čo je f\_effective\_spell\_cost:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.Procedúra sp\_cast\_spell tak isto môže dopadnúť aj smrťou niekoho, ak tak útok dopadne - pre jednoduchšie testy bola vytvorená procedúra sp\_end\_combat, ktorá pošle všetkých účastníkov domou a zavrie combat.

Obrázok 18

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Obrázok 19

Teda ďalej, podlá scenára by hrací mali sa navzájom ničiť, a to tak že budú volať:

CALL sp\_cast\_spell(1,1,2,1); pre hraca 1

CALL sp\_cast\_spell(1,2,1,2); pre hraca 2

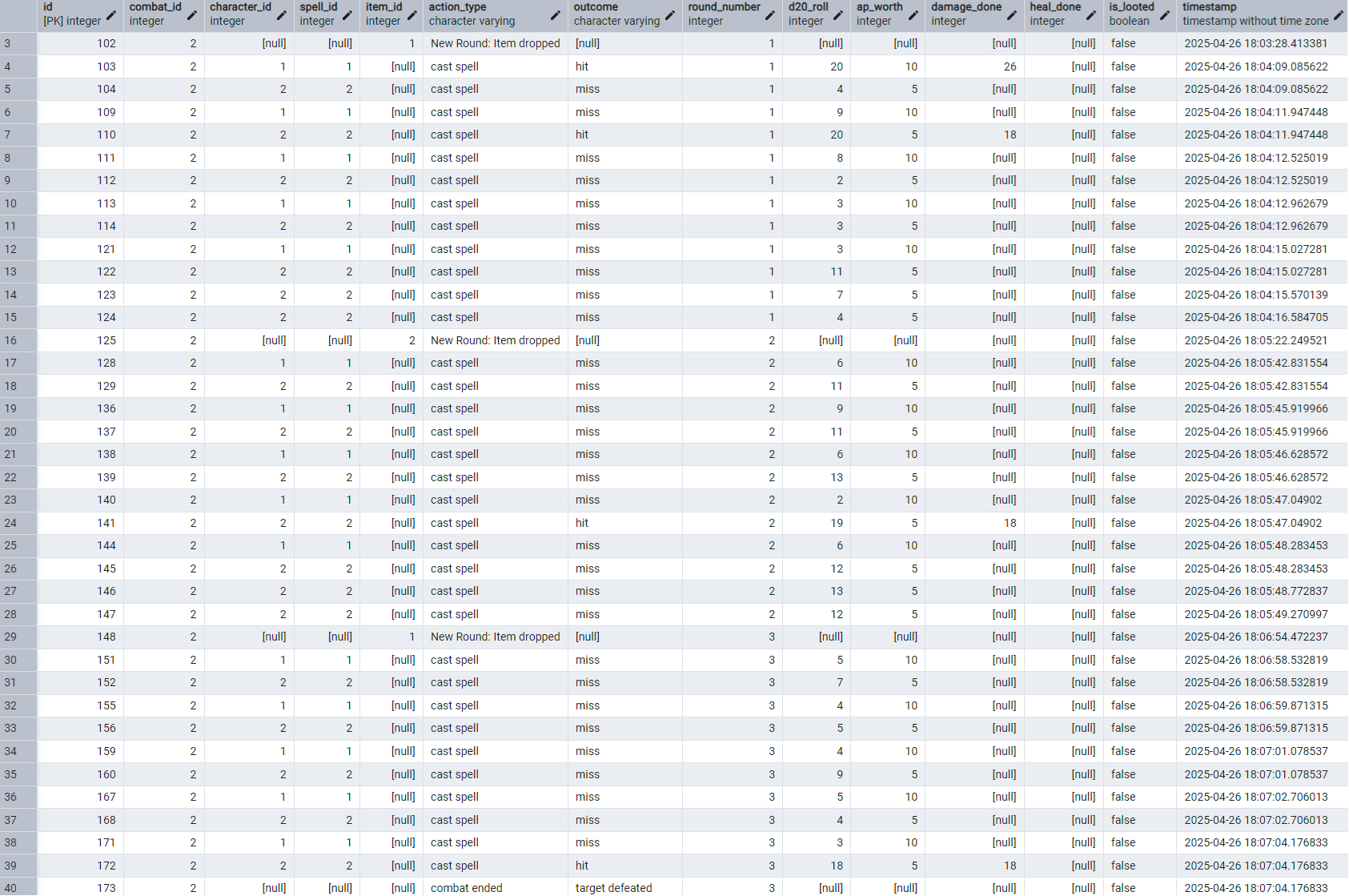
Kým niekto nezomrie, alebo kým niekomu nedôjde AP, teda kým nevypíše:



Alebo:



Potom combat\_log pre daný combat\_id po celej bitke môže vyzerať takto:



Obrázok 20

**TEST sp\_rest\_character**

Изображение выглядит как текст, Шрифт, снимок экрана, линия

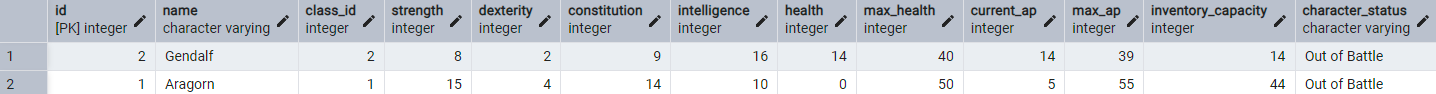
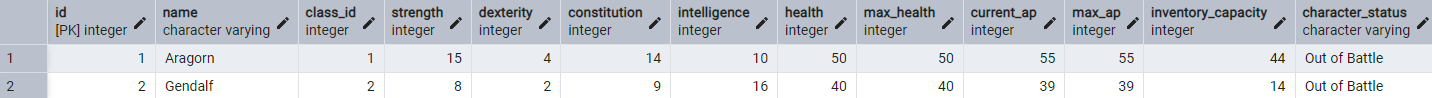
Контент, сгенерированный ИИ, может содержать ошибки.

Obrázok 21

Kým hrací sú mimo bitky, vedia si oddýchnuť, a to použitím:

CALL sp\_rest\_character(id)

Teda bude to vyzerať takto pre našich dvoch postav do a po použití procedúry po danej bitke:



Obrázok 22

**VIEWS**

Tak isto boli vytvorené aj špeciálne views, aby pozrieť nejaké štatistika databasy. Prebehneme si ich v krátkosti.

**v\_combat\_damage**

**Изображение выглядит как текст, Шрифт, линия, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.**

**Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.**

Obrázok 23

**v\_combat\_state**

**Изображение выглядит как текст, Шрифт, линия, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.** **Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.**

Obrázok 24

**v\_most\_damage**

**Изображение выглядит как текст, Шрифт, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.**

Obrázok 25

**v\_spell\_statistics**

**Изображение выглядит как текст, Шрифт, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.Изображение выглядит как текст, Шрифт, линия, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.**

Obrázok 26

**v\_strongest\_characters**

**Изображение выглядит как текст, Шрифт, линия, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.** **Изображение выглядит как текст, Шрифт, линия, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.**

Obrázok 27