DBS Zadanie 3

Dokumentácia

Rodion Burmistrov

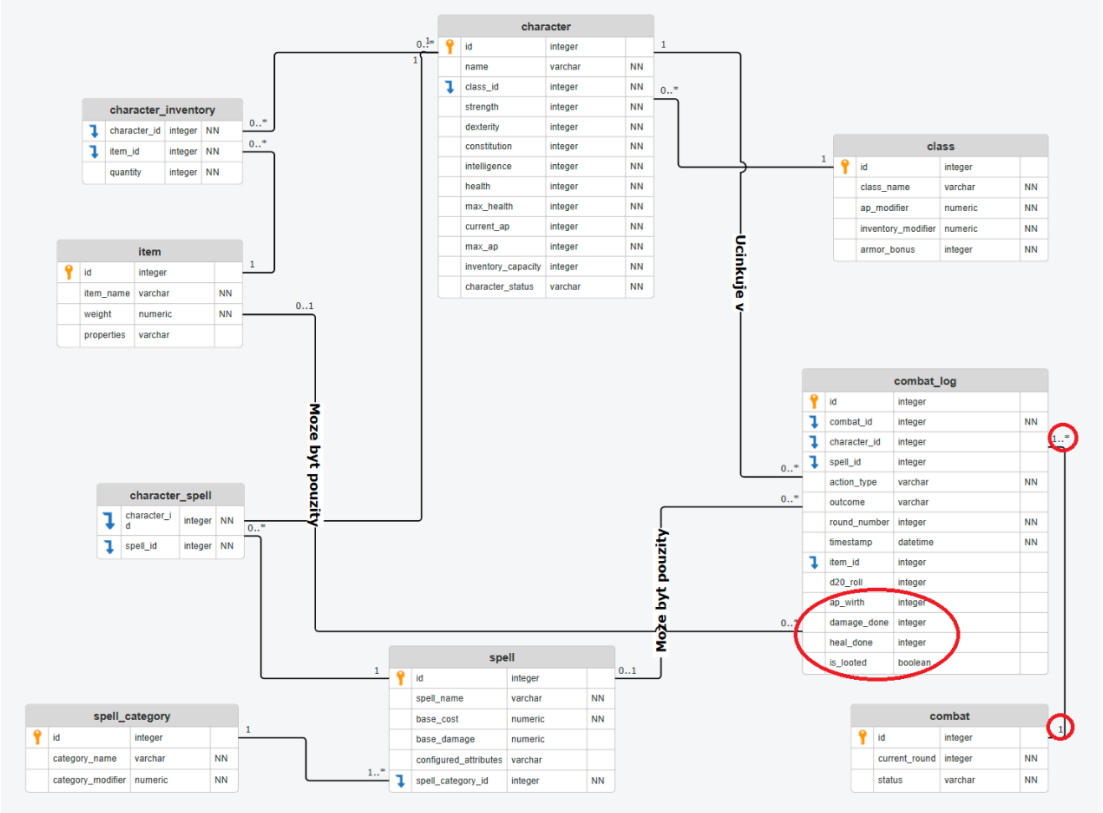
V rámci Zadania číslo 3 bolo potrebné realizovať databázu pre RPG combat hru, navrhnutú v predošlom zadaní. V rámci realizácie sú zarátane tak isto aj rôzne SQL procedúry, indexy, jedna funkcia a pohľady. Ďalej v dokumente je popísaný navrhnutý systém, jeho entity a ich vzťahy, vrátane požadovaných procesov.

Zmeny Návrhu Databázy:

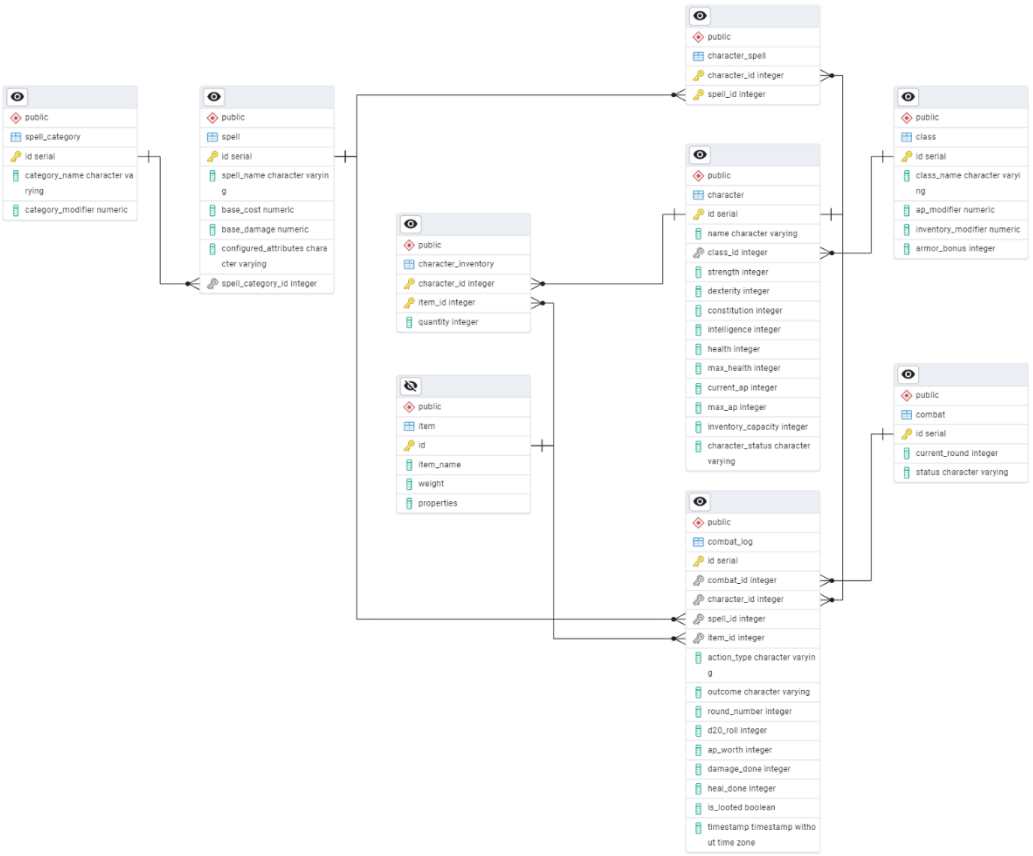
V priebehu kontrolovania zadania číslo 2 sa zistilo, že pôvodný Návrh nebol úplne úspešný. Zmeny boli nasledujúce:

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“.

2. **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:

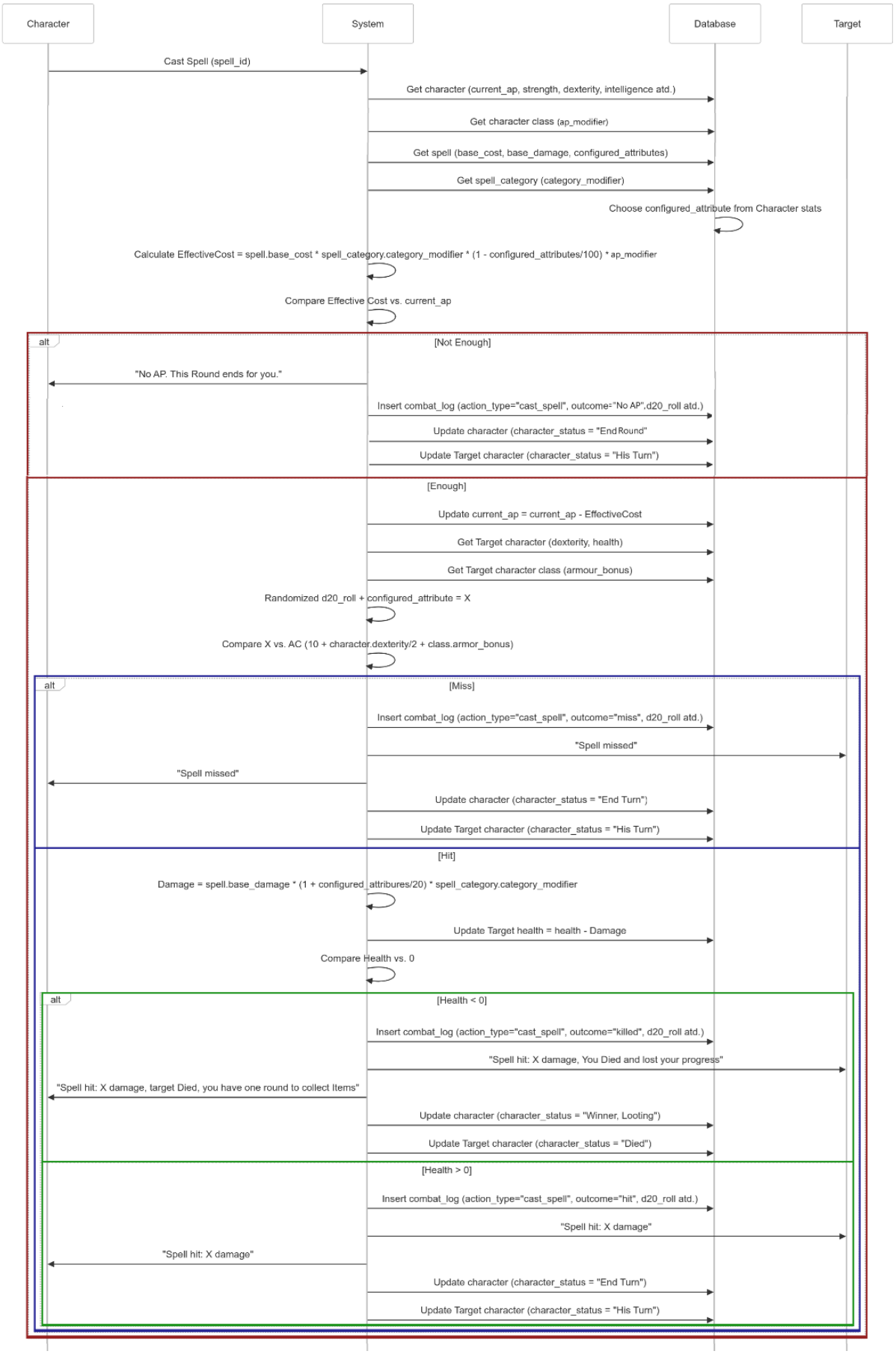
- **class**: ukladá ID, názov triedy (napr. „Mág“), modifikátor akčných bodov (ap_modifier), kapacitu inventára (inventory_modifier) a bonus k brneniu (armor_bonus).
- **character**: reprezentuje postavy; okrem ID a mena obsahuje FK na **class**, štatistiky (strength, dexterity, constitution, intelligence), 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**

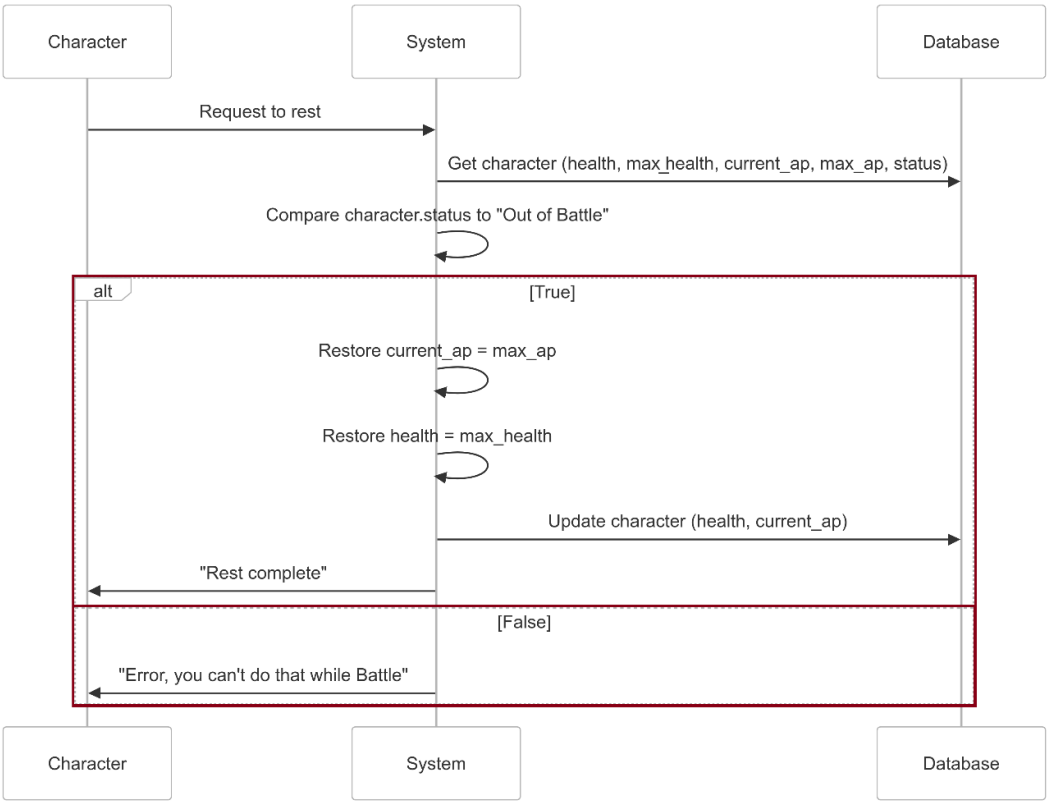
Proces posielanie kúzla, počas bitky prebieha nasledovne. Najprv sa overí, či je boj aktívny, či sú Caster a Target su jeho účastníkmi a či Caster ovláda dané kúzlo. Potom sa vypočíta efektívna cena kúzla pomocou funkcie **f_effective_spell_cost**, ktorá berie základnú cenu spellu, modifikátor kategórie, 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 ukončí. 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


Character
System
Database
Target
Cast Spell (spell_id)
Get character (current_ap, strength, dexterity, intelligence atd.)
Get character class (ap_modifier)
Get spell (base_cost, base_damage, configured_attributes)
Get spell_category (category_modifier)
Choose configured_attribute from Character stats
Calculate EffectiveCost = spell.base_cost * spell_category.category_modifier * (1 - configured_attributes/100) * ap_modifier
Compare Effective Cost vs. current_ap
alt
[Not Enough]
"No AP. This Round ends for you."
Insert combat_log (action_type="cast_spell", outcome="No AP",d20_roll atd.)
Update character (character_status = "End Round"
Update Target character (character_status = "His Turn")
[Enough]
Update current_ap = current_ap - EffectiveCost
Get Target character (dexterity, health)
Get Target character class (armour_bonus)
Randomized d20_roll + configured_attribute = X
Compare X vs. AC (10 + character.dexterity/2 + class.armor_bonus)
alt
[Miss]
Insert combat_log (action_type="cast_spell", outcome="miss", d20_roll atd.)
"Spell missed"
"Spell missed"
Update character (character_status = "End Turn")
Update Target character (character_status = "His Turn")
[Hit]
Damage = spell.base_damage * (1 + configured_attribures/20) * spell_category.category_modifier
Update Target health = health - Damage
Compare Health vs. 0
alt
[Health < 0]
Insert combat_log (action_type="cast_spell", outcome="killed", d20_roll atd.)
"Spell hit: X damage, You Died and lost your progress"
"Spell hit: X damage, target Died, you have one round to collect Items"
Update character (character_status = "Winner, Looting")
Update Target character (character_status = "Died")
[Health > 0]
Insert combat_log (action_type="cast_spell", outcome="hit", d20_roll atd.)
"Spell hit: X damage"
"Spell hit: X damage"
Update character (character_status = "End Turn")
Update Target character (character_status = "His Turn")

Obrázok 3

2. sp_rest_character

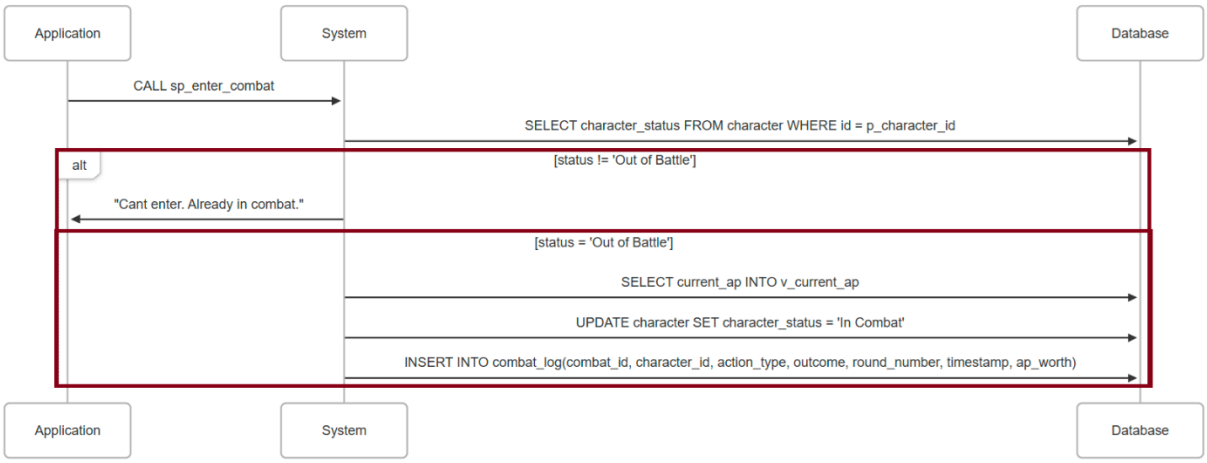
Táto procedúra umožňuje postave oddýchnuť a obnoviť zdravie a akčné body (AP), ale iba ak nie je v boji. A to tak, ze skontroluje, či má postava stav character_status = 'Out of Battle'. Ak áno, aktualizuje jej zdravie (health) a AP (current_ap) na maximálne hodnoty (max_health a max_ap) v databáze. Ak 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



Obrázok 4

3. sp_enter_combat

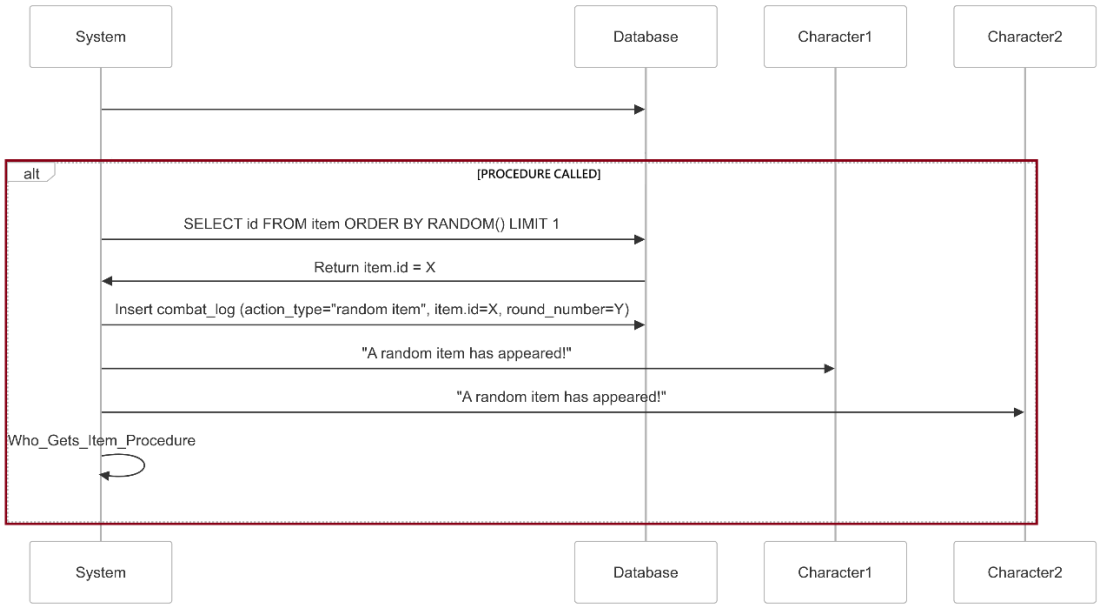
Procedúra spracováva vstup postavy do boja. Overí, či je postava mimo boja (character_status = 'Out of Battle'), inak vyhodí chybu. Ak je podmienka splnená, získa aktuálne AP postavy (current_ap), zmení jej stav na ‘In Combat’ a vytvorí záznam v combat_log s akciou enter combat, označenou ako úspešná, 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

4. sp_rest_character

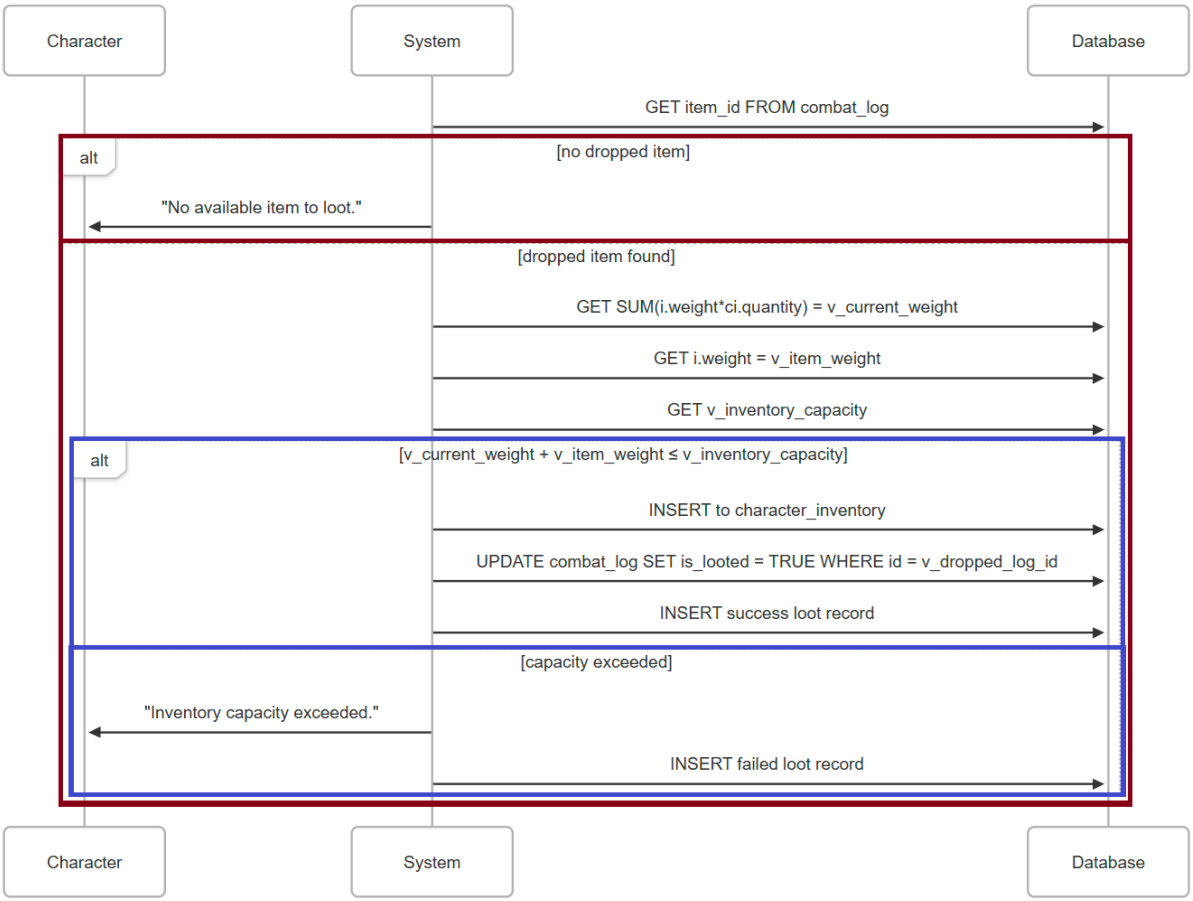
Procedúra obnovuje kolo v aktívnom boji. Skontroluje, či je boj aktívny (status = 'active'). Ak áno, obnoví AP všetkých účastníkov na ich maximálne hodnoty (max_ap), zvýši číslo kola (current_round) o 1 a náhodne vyberie predmet z tabuľky item pomocou RANDOM(). Záznam o novom kole a vybranom predmete 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.



Obrázok 6

5. 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 .

```
-- INDEXY
-- pouzil som iba tieto, lebo hlavne som pouzival SELECT iba na tieto atributy pocas testingu
CREATE INDEX idx_combat_log_combat_id ON combat_log(combat_id);
CREATE INDEX idx_combat_log_character_id ON combat_log(character_id);
CREATE INDEX idx_combat_log_combat_character_id ON combat_log(combat_id, character_id);
CREATE INDEX idx_character_spell_character_id ON character_spell(character_id);
CREATE INDEX idx_character_inventory_character_id ON character_inventory(character_id);
```

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 .

	name character varying	class_name character varying	spell_name character varying	category_name character varying	item_name character varying	quantity integer
1	Aragorn	Warrior	SwordAttack	SwordAttack	Sword	1
2	Gendalf	Mage	FireBall	FireBall	Healing Potion	2

Obrázok 9

Ďalej zbehneme všetky zvyšne funkcie a procedúry.

TEST sp_enter_combat

```
CREATE OR REPLACE PROCEDURE sp_enter_combat(p_combat_id INTEGER, p_character_id INTEGER)
LANGUAGE plpgsql AS $$
DECLARE
    v_current_ap INTEGER;
BEGIN

    -- Check if character je free
    IF (SELECT character_status FROM character WHERE id = p_character_id) != 'Out of Battle' THEN
        RAISE EXCEPTION 'Cant enter. Already in combat.';
    END IF;

    -- vytiehneme aktualny AP z characteru
    SELECT current_ap INTO v_current_ap FROM character WHERE id = p_character_id;

    --update characteru, ze je v bitke, toto bude potrebne dalej pre to, aby nevedel pouzit rest
    UPDATE character SET character_status = 'In Combat' WHERE id = p_character_id;
    INSERT INTO combat_log (combat_id, character_id, action_type, outcome, round_number, timestamp, ap_worth)
    VALUES (p_combat_id, p_character_id, 'enter combat', 'success', 0, CURRENT_TIMESTAMP, v_current_ap);
END;
$$;
```

Obrázok 10

Pouzijeme jednoduchy SELECT, aby preukazat prazdnost tabulky combat_log.

```
SELECT *
FROM combat_log
```

	id [PK] integer	combat_id integer	character_id integer	spell_id integer	item_id integer	action_type character varying	outcome character varying	round_number integer	d20_roll integer	ap_worth integer	damage_done integer	heal_done integer	is_looted boolean	timestamp timestamp without time zone

Obrázok 11

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

```
CALL sp_enter_combat(1,1);
CALL sp_enter_combat(1,2)
```

	id [PK] integer	combat_id integer	character_id integer	spell_id integer	item_id integer	action_type character varying	outcome character varying	round_number integer	d20_roll integer	ap_worth integer	damage_done integer	heal_done integer	is_looted boolean	timestamp timestamp without time zone
1	2	1	1	[null]	[null]	enter combat	success	0	[null]	22	[null]	[null]	false	2025-04-26 16:23:44.534351
2	3	1	2	[null]	[null]	enter combat	success	0	[null]	39	[null]	[null]	false	2025-04-26 16:24:49.472604

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:

```
ERROR:  Combat is not active.
```

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

```
CREATE OR REPLACE PROCEDURE sp_reset_round(p_combat_id INTEGER)
LANGUAGE plpgsql AS $$
DECLARE
    v_random_item_id INTEGER;
BEGIN
    -- Ci bitka vobec je
    IF (SELECT status FROM combat WHERE id = p_combat_id) != 'active' THEN
        RAISE EXCEPTION 'Combat is not active.';
    END IF;

    -- Refresh AP pre vsetkych ucastnikov
    UPDATE character c
    SET current_ap = c.max_ap
    WHERE c.id IN (SELECT DISTINCT character_id
                   FROM combat_log
                   WHERE combat_id = p_combat_id
                   AND character_id IS NOT NULL);

    -- +1 round do combatu
    UPDATE combat
    SET current_round = current_round + 1
    WHERE id = p_combat_id;

    -- Select random item
    SELECT id INTO v_random_item_id
    FROM item
    ORDER BY RANDOM()
    LIMIT 1;

    -- Log s refresh roundom a itemom
    INSERT INTO combat_log (combat_id, action_type, item_id, round_number, timestamp)
    VALUES (p_combat_id, 'New Round: Item dropped', v_random_item_id,
            (SELECT current_round FROM combat WHERE id = p_combat_id),
            CURRENT_TIMESTAMP);
END;
$$;
```

Obrázok 13

CALL sp_reset_round(1)

	id [PK] integer	combat_id integer	character_id integer	spell_id integer	item_id integer	action_type character varying	outcome character varying	round_number integer	d20_roll integer	ap_worth integer	damage_done integer	heal_done integer	is_looted boolean	timestamp timestamp without time zone
1	2	1	1	[null]	[null]	enter combat	success	0	[null]	22	[null]	[null]	false	2025-04-26 16:23:44.534351
2	3	1	2	[null]	[null]	enter combat	success	0	[null]	39	[null]	[null]	false	2025-04-26 16:24:49.472604
3	4	1	[null]	[null]	1	New Round: Item dropped	[null]	1	[null]	[null]	[null]	[null]	false	2025-04-26 16:36:36.230645

Obrázok 14

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

TEST sp_loot_item

```
CREATE OR REPLACE PROCEDURE sp_loot_item(p_combat_id INTEGER, p_character_id INTEGER, p_item_id INTEGER)
LANGUAGE plpgsql AS $$
DECLARE
    v_dropped_log_id INTEGER;
    v_current_weight NUMERIC;
    v_item_weight NUMERIC;
    v_inventory_capacity INTEGER;
BEGIN
    -- Check if combat je active
    IF (SELECT status FROM combat WHERE id = p_combat_id) != 'active' THEN
        RAISE EXCEPTION 'Combat is not active.';
    END IF;

    -- Najst nieco, co este lezi
    SELECT id INTO v_dropped_log_id
    FROM combat_log
    WHERE combat_id = p_combat_id
      AND action_type = 'New Round: Item dropped'
      AND item_id = p_item_id
      AND is_looted = FALSE
    ORDER BY id ASC
    LIMIT 1;

    -- Ak neneslo, tak nic
    IF v_dropped_log_id IS NULL THEN
        RAISE NOTICE 'No available item to loot.';
        RETURN;
    END IF;

    -- Kolko vazi to, co ma teraz
    SELECT COALESCE(SUM(i.weight * ci.quantity), 0)
    INTO v_current_weight
    FROM character_inventory ci
    JOIN item i ON ci.item_id = i.id
    WHERE ci.character_id = p_character_id;

    -- Vaha Itemu a kapacita inventara
    SELECT weight INTO v_item_weight FROM item WHERE id = p_item_id;
    SELECT inventory_capacity INTO v_inventory_capacity FROM character WHERE id = p_character_id;

    -- Porovnanie ci to vie zobrat
    IF v_current_weight + v_item_weight <= v_inventory_capacity THEN
        -- Pridanie do inventaru
        INSERT INTO character_inventory (character_id, item_id, quantity)
        VALUES (p_character_id, p_item_id, 1)
        ON CONFLICT (character_id, item_id) DO UPDATE  -- ak uz podobny zaznam char+idem je, tak iba spravime +1
        SET quantity = character_inventory.quantity + 1;

        -- Update current logu
        UPDATE combat_log
        SET is_looted = TRUE
        WHERE id = v_dropped_log_id;

        -- Uspech, alebo neuspech
        INSERT INTO combat_log (combat_id, character_id, item_id, action_type, outcome, round_number, timestamp)
        VALUES (p_combat_id, p_character_id, p_item_id, 'loot item', 'success',
                (SELECT current_round FROM combat WHERE id = p_combat_id),
                CURRENT_TIMESTAMP);
    ELSE
        RAISE NOTICE 'Inventory capacity exceeded.';
        INSERT INTO combat_log (combat_id, character_id, item_id, action_type, outcome, round_number, timestamp)
        VALUES (p_combat_id, p_character_id, p_item_id, 'loot item', 'failed',
                (SELECT current_round FROM combat WHERE id = p_combat_id),
                CURRENT_TIMESTAMP);
    END IF;
END;
$$;
```

Obrázok 15

CALL sp_loot_item(1,1,item_id)

	id [PK] integer	combat_id integer	character_id integer	spell_id integer	item_id integer	action_type character varying	outcome character varying	round_number integer	d20_roll integer	ap_worth integer	damage_done integer	heal_done integer	is_looted boolean	timestamp timestamp without time zone
1	2	1	1	[null]	[null]	enter combat	success	0	[null]	22	[null]	[null]	false	2025-04-26 16:23:44.534351
2	3	1	2	[null]	[null]	enter combat	success	0	[null]	39	[null]	[null]	false	2025-04-26 16:24:49.472604
3	4	1	[null]	[null]	1	New Round: Item dropped	[null]	1	[null]	[null]	[null]	[null]	true	2025-04-26 16:36:36.230645
4	5	1	1	[null]	1	loot item	success	1	[null]	[null]	[null]	[null]	false	2025-04-26 16:43:54.855248

Obrázok 16

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

TEST sp_cast_spell

```
CREATE OR REPLACE PROCEDURE sp_cast_spell(p_combat_id INTEGER, p_caster_id INTEGER, p_target_id INTEGER, p_spell_id INTEGER)
LANGUAGE plpgsql AS $$
DECLARE
    v_current_ap INTEGER;
    v_effective_cost NUMERIC;
    v_d20_roll INTEGER;
    v_attribute_bonus INTEGER;
    v_ac_target INTEGER;
    v_damage NUMERIC;
    v_base_damage NUMERIC;
    v_configured_attribute VARCHAR;
    v_attribute_value INTEGER;
    v_target_health INTEGER;
BEGIN
    -- Pozriet aktivnost combatu
    IF (SELECT status FROM combat WHERE id = p_combat_id) != 'active' THEN
        RAISE EXCEPTION 'Combat is not active.';
    END IF;

    -- Pozrieme pritomnost castera
    IF NOT EXISTS (SELECT 1 FROM combat_log WHERE combat_id = p_combat_id AND character_id = p_caster_id) THEN
        RAISE EXCEPTION 'Caster is not in this combat.';
    END IF;

    -- Pozrieme pritomnost targetu
    IF NOT EXISTS (SELECT 1 FROM combat_log WHERE combat_id = p_combat_id AND character_id = p_target_id) THEN
        RAISE EXCEPTION 'Target is not in this combat.';
    END IF;

    -- Ci caster vobec vie spell
    IF NOT EXISTS (SELECT 1 FROM character_spell WHERE character_id = p_caster_id AND spell_id = p_spell_id) THEN
        RAISE EXCEPTION 'Caster does not know this spell.';
    END IF;

    -- Volanie funkcii na pocitanie ceny spellu, dalej je to iba v_effective_cost
    SELECT current_ap INTO v_current_ap FROM character WHERE id = p_caster_id;
    v_effective_cost := f_effective_spell_cost(p_spell_id, p_caster_id);

    -- Nema AP, teda neide to
    IF v_current_ap < v_effective_cost THEN
        RAISE NOTICE 'Insufficient AP to cast spell.';
        RETURN;
    END IF;

    -- Ak ma, tak sa mu to odcita v tabulke characteri
    UPDATE character SET current_ap = current_ap - v_effective_cost WHERE id = p_caster_id;
   -- Hotovo, pocitame Damage
   SELECT base_damage, configured_attributes INTO v_base_damage, v_configured_attribute FROM spell WHERE id = p_spell_id;

   -- Pozrieme aky to ma atribut na scaling, dalej je v v_attribute_value
   SELECT
       CASE v_configured_attribute
           WHEN 'strength' THEN strength
           WHEN 'dexterity' THEN dexterity
           WHEN 'constitution' THEN constitution
           WHEN 'intelligence' THEN intelligence
           ELSE 0
       END
   INTO v_attribute_value
   FROM character WHERE id = p_caster_id;

   -- Na kalkulaciu tohto som sa pozriel do internetu, konkretne https://stackoverflow.com/questions/1400505/generate-a-random-number-in-the-range-1-10
   -- CEIL teda aj zokruhluje napriklad 0.00001 na 1, ale 0,0000 na 0
   v_d20_roll := CEIL(RANDOM() * 20);
   v_attribute_bonus := v_attribute_value / 5; -- Pre balance, lebo s tym som trochu pohral, aby to realne malo balance
   v_d20_roll := v_d20_roll + v_attribute_bonus; -- toto zvycajne malo od ~10 do az 40, tak muselo sa to vydelit 5, aby to bolo v priemere od 0
                                                -- (co musi postava pouzivat to, co jej nepatri) do 20, ak sa jej velmi podari s tym.
                                                -- a viac priemerne od 4 do 24 (to iba dufat, ze nepriatel nema vsetko na 10 lvl)

   -- Obrana targetu. Pocita sa cez to, co sme mali dane rovnicu, je to take ze v rozmedzi ~12 (ked postava ma dexterity 2 a bonus 1)  az ~25 (dex=10, armour=10)
   SELECT 10 + (dexterity / 2) + cl.armor_bonus
   INTO v_ac_target
   FROM character c
   JOIN class cl ON c.class_id = cl.id
   WHERE c.id = p_target_id;


   --  Porovnavanie stastia a obrany a vysledok, update pre character
   IF v_d20_roll > v_ac_target THEN
       v_damage := v_base_damage * (1 + v_attribute_value / 20.0);
       UPDATE character SET health = GREATEST(0, health - v_damage) WHERE id = p_target_id;
       INSERT INTO combat_log (combat_id, character_id, spell_id, action_type, outcome, round_number, timestamp, d20_roll, ap_worth, damage_done)
       VALUES (p_combat_id, p_caster_id, p_spell_id, 'cast spell', 'hit', (SELECT current_round FROM combat WHERE id = p_combat_id), CURRENT_TIMESTAMP, v_d20_roll, v_effective_cost, v_damage);

       -- Ak nepriatel ma 0, tak volame endbattle
       SELECT health INTO v_target_health FROM character WHERE id = p_target_id;
       IF v_target_health <= 0 THEN
           CALL sp_end_combat(p_combat_id);
           -- Piseme log na prehru
           INSERT INTO combat_log (combat_id, action_type, outcome, round_number, timestamp)
           VALUES (p_combat_id, 'combat ended', 'target defeated', (SELECT current_round FROM combat WHERE id = p_combat_id), CURRENT_TIMESTAMP);
       END IF;
   ELSE
       INSERT INTO combat_log (combat_id, character_id, spell_id, action_type, outcome, round_number, timestamp, d20_roll, ap_worth)
       VALUES (p_combat_id, p_caster_id, p_spell_id, 'cast spell', 'miss', (SELECT current_round FROM combat WHERE id = p_combat_id), CURRENT_TIMESTAMP, v_d20_roll, v_effective_cost);
   END IF;
END;
$$;
```

Obrázok 17

Tato procedúra tak isto zahŕňa aj funkciu počítania damage, čo je f_effective_spell_cost:

```
CREATE OR REPLACE FUNCTION f_effective_spell_cost(p_spell_id INTEGER, p_caster_id INTEGER)
RETURNS NUMERIC AS $$
DECLARE
    v_base_cost NUMERIC;
    v_category_modifier NUMERIC;
    v_configured_attribute VARCHAR;
    v_attribute_value INTEGER;
    v_ap_modifier NUMERIC;
BEGIN

    -- Zvolime cenu, category modifier, a aky je atribut scalingu
    SELECT s.base_cost, sc.category_modifier, s.configured_attributes
    INTO v_base_cost, v_category_modifier, v_configured_attribute
    FROM spell s
    JOIN spell_category sc ON s.spell_category_id = sc.id
    WHERE s.id = p_spell_id;

    -- Jak v sp_cast_spell preberieme hodnotu atributu a ap modifikator
    SELECT
        CASE v_configured_attribute
            WHEN 'strength' THEN c.strength
            WHEN 'dexterity' THEN c.dexterity
            WHEN 'constitution' THEN c.constitution
            WHEN 'intelligence' THEN c.intelligence
            ELSE 0
        END,
        cl.ap_modifier
    INTO v_attribute_value, v_ap_modifier
    FROM character c
    JOIN class cl ON c.class_id = cl.id
    WHERE c.id = p_caster_id;

    -- Vratim cenu spellu, podla rovnici, nadej zadaniem. Typ atributu zmenime na NUMERIC, aby to nebolo vzdy nula. Lebo povodne som to definoval ako Integer
    RETURN v_base_cost * v_category_modifier * (1 - (v_attribute_value::NUMERIC / 100)) * v_ap_modifier;
END;
$$ LANGUAGE plpgsql;
```

Obrázok 18

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.

```
CREATE OR REPLACE PROCEDURE sp_end_combat(p_combat_id INTEGER)
LANGUAGE plpgsql AS $$
BEGIN
    -- Update combat na zavrety
    UPDATE combat
    SET status = 'ended'
    WHERE id = p_combat_id;

    -- Charactery su automaticky "Out of Battle"
    UPDATE "character" c
    SET character_status = 'Out of Battle'
    WHERE c.id IN (SELECT DISTINCT character_id
                   FROM combat_log
                   WHERE combat_id = p_combat_id
                   AND character_id IS NOT NULL);
END;
$$;
```

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:

```
ERROR:  Combat is not active.
```

Alebo:

```
ЗАМЕЧАНИЕ:  Insufficient AP to cast spell.
```

Potom combat_log pre daný combat_id po celej bitke môže vyzerať takto:

	id [PK] integer	combat_id integer	character_id integer	spell_id integer	item_id integer	action_type character varying	outcome character varying	round_number integer	d20_roll integer	ap_worth integer	damage_done integer	heal_done integer	is_looted boolean	timestamp timestamp without time zone
3	102	2	[null]	[null]	1	New Round: Item dropped	[null]	1	[null]	[null]	[null]	[null]	false	2025-04-26 18:03:28.413381
4	103	2	1	1	[null]	cast spell	hit	1	20	10	26	[null]	false	2025-04-26 18:04:09.085622
5	104	2	2	2	[null]	cast spell	miss	1	4	5	[null]	[null]	false	2025-04-26 18:04:09.085622
6	109	2	1	1	[null]	cast spell	miss	1	9	10	[null]	[null]	false	2025-04-26 18:04:11.947448
7	110	2	2	2	[null]	cast spell	hit	1	20	5	18	[null]	false	2025-04-26 18:04:11.947448
8	111	2	1	1	[null]	cast spell	miss	1	8	10	[null]	[null]	false	2025-04-26 18:04:12.525019
9	112	2	2	2	[null]	cast spell	miss	1	2	5	[null]	[null]	false	2025-04-26 18:04:12.525019
10	113	2	1	1	[null]	cast spell	miss	1	3	10	[null]	[null]	false	2025-04-26 18:04:12.962679
11	114	2	2	2	[null]	cast spell	miss	1	3	5	[null]	[null]	false	2025-04-26 18:04:12.962679
12	121	2	1	1	[null]	cast spell	miss	1	3	10	[null]	[null]	false	2025-04-26 18:04:15.027281
13	122	2	2	2	[null]	cast spell	miss	1	11	5	[null]	[null]	false	2025-04-26 18:04:15.027281
14	123	2	2	2	[null]	cast spell	miss	1	7	5	[null]	[null]	false	2025-04-26 18:04:15.570139
15	124	2	2	2	[null]	cast spell	miss	1	4	5	[null]	[null]	false	2025-04-26 18:04:16.584705
16	125	2	[null]	[null]	2	New Round: Item dropped	[null]	2	[null]	[null]	[null]	[null]	false	2025-04-26 18:05:22.249521
17	128	2	1	1	[null]	cast spell	miss	2	6	10	[null]	[null]	false	2025-04-26 18:05:42.831554
18	129	2	2	2	[null]	cast spell	miss	2	11	5	[null]	[null]	false	2025-04-26 18:05:42.831554
19	136	2	1	1	[null]	cast spell	miss	2	9	10	[null]	[null]	false	2025-04-26 18:05:45.919966
20	137	2	2	2	[null]	cast spell	miss	2	11	5	[null]	[null]	false	2025-04-26 18:05:45.919966
21	138	2	1	1	[null]	cast spell	miss	2	6	10	[null]	[null]	false	2025-04-26 18:05:46.628572
22	139	2	2	2	[null]	cast spell	miss	2	13	5	[null]	[null]	false	2025-04-26 18:05:46.628572
23	140	2	1	1	[null]	cast spell	miss	2	2	10	[null]	[null]	false	2025-04-26 18:05:47.04902
24	141	2	2	2	[null]	cast spell	hit	2	19	5	18	[null]	false	2025-04-26 18:05:47.04902
25	144	2	1	1	[null]	cast spell	miss	2	6	10	[null]	[null]	false	2025-04-26 18:05:48.283453
26	145	2	2	2	[null]	cast spell	miss	2	12	5	[null]	[null]	false	2025-04-26 18:05:48.283453
27	146	2	2	2	[null]	cast spell	miss	2	13	5	[null]	[null]	false	2025-04-26 18:05:48.772837
28	147	2	2	2	[null]	cast spell	miss	2	12	5	[null]	[null]	false	2025-04-26 18:05:49.270997
29	148	2	[null]	[null]	1	New Round: Item dropped	[null]	3	[null]	[null]	[null]	[null]	false	2025-04-26 18:06:54.472237
30	151	2	1	1	[null]	cast spell	miss	3	5	10	[null]	[null]	false	2025-04-26 18:06:58.532819
31	152	2	2	2	[null]	cast spell	miss	3	7	5	[null]	[null]	false	2025-04-26 18:06:58.532819
32	155	2	1	1	[null]	cast spell	miss	3	4	10	[null]	[null]	false	2025-04-26 18:06:59.871315
33	156	2	2	2	[null]	cast spell	miss	3	5	5	[null]	[null]	false	2025-04-26 18:06:59.871315
34	159	2	1	1	[null]	cast spell	miss	3	4	10	[null]	[null]	false	2025-04-26 18:07:01.078537
35	160	2	2	2	[null]	cast spell	miss	3	9	5	[null]	[null]	false	2025-04-26 18:07:01.078537
36	167	2	1	1	[null]	cast spell	miss	3	5	10	[null]	[null]	false	2025-04-26 18:07:02.706013
37	168	2	2	2	[null]	cast spell	miss	3	4	5	[null]	[null]	false	2025-04-26 18:07:02.706013
38	171	2	1	1	[null]	cast spell	miss	3	3	10	[null]	[null]	false	2025-04-26 18:07:04.176833
39	172	2	2	2	[null]	cast spell	hit	3	18	5	18	[null]	false	2025-04-26 18:07:04.176833
40	173	2	[null]	[null]	[null]	combat ended	target defeated	3	[null]	[null]	[null]	[null]	false	2025-04-26 18:07:04.176833

Obrázok 20

TEST sp_rest_character

```
CREATE OR REPLACE PROCEDURE sp_rest_character(p_character_id INTEGER)
LANGUAGE plpgsql AS $$
BEGIN -- if character nie je v bitke, vie si oddychnut
    IF (SELECT character_status FROM character WHERE id = p_character_id) = 'Out of Battle' THEN
        UPDATE character
        SET health = max_health,
            current_ap = max_ap
        WHERE id = p_character_id;
    ELSE
        RAISE NOTICE 'Cannot rest during combat.';
    END IF;
END;
$$;
```

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:

	id [PK] integer	name character varying	class_id integer	strength integer	dexterity integer	constitution integer	intelligence integer	health integer	max_health integer	current_ap integer	max_ap integer	inventory_capacity integer	character_status character varying
1	2	Gendalf	2	8	2	9	16	14	40	14	39	14	Out of Battle
2	1	Aragorn	1	15	4	14	10	0	50	5	55	44	Out of Battle
	id [PK] integer	name character varying	class_id integer	strength integer	dexterity integer	constitution integer	intelligence integer	health integer	max_health integer	current_ap integer	max_ap integer	inventory_capacity integer	character_status character varying
1	1	Aragorn	1	15	4	14	10	50	50	55	55	44	Out of Battle
2	2	Gendalf	2	8	2	9	16	40	40	39	39	14	Out of Battle

Obrázok 22

VIEWS

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

v_combat_damage

```
CREATE OR REPLACE VIEW v_combat_damage AS
SELECT c.id AS combat_id, SUM(cl.damage_done) AS total_damage
FROM combat c
JOIN combat_log cl ON c.id = cl.combat_id AND cl.action_type = 'cast spell' AND cl.outcome = 'hit'
GROUP BY c.id
ORDER BY total_damage DESC;
-- Pise mnozstvo damagu, spraveneho pocas jednotlivych bitiek
```

	combat_id integer	total_damage bigint
1	2	80
2	1	62

Obrázok 23

v_combat_state

```
CREATE OR REPLACE VIEW v_combat_state AS
SELECT c.id AS combat_id, c.current_round, ch.name, ch.current_ap, ch.character_status, c.status
FROM combat c
JOIN combat_log cl ON c.id = cl.combat_id
JOIN character ch ON cl.character_id = ch.id
GROUP BY c.id, c.current_round, ch.name, ch.current_ap, ch.character_status
ORDER BY c.id;
-- Pise status, round a ucastnikov vsetkych bitiek + ich aktualne AP
```

	combat_id integer	current_round integer	name character varying	current_ap integer	character_status character varying	status character varying
1	1	1	Aragorn	55	Out of Battle	ended
2	1	1	Gendalf	39	Out of Battle	ended
3	2	3	Aragorn	55	Out of Battle	ended
4	2	3	Gendalf	39	Out of Battle	ended

Obrázok 24

v_most_damage

```
CREATE OR REPLACE VIEW v_most_damage AS
SELECT c.name, SUM(cl.damage_done) AS total_damage
FROM character c
JOIN combat_log cl ON c.id = cl.character_id AND cl.action_type = 'cast spell' AND cl.outcome = 'hit'
GROUP BY c.name
ORDER BY total_damage DESC;
-- Pise aka postava dala najviac damagu
```

	name character varying	total_damage bigint
1	Gendalf	90
2	Aragorn	52

Obrázok 25

v_spell_statistics

```sql
CREATE OR REPLACE VIEW v_spell_statistics AS
SELECT s.spell_name, COUNT(cl.spell_id) AS usage_count, ROUND(AVG(cl.damage_done), 2) AS avg_damage,
    ROUND(SUM(CASE WHEN cl.outcome = 'hit' THEN 1 ELSE 0 END) / COUNT(*)::NUMERIC * 100, 2) AS hit_percentage
FROM spell s
JOIN combat_log cl ON s.id = cl.spell_id
WHERE cl.action_type = 'cast spell'
GROUP BY s.spell_name
ORDER BY hit_percentage DESC;
-- Pise kolkokrat sa pouzil spell, jeho priemerny damage a percento kolko krat trafil
```

	spell_name character varying	usage_count bigint	avg_damage numeric	hit_percentage numeric
1	FireBall	26	18.00	19.23
2	SwordAttack	20	26.00	10.00

Obrázok 26

v_strongest_characters

```sql
CREATE OR REPLACE VIEW v_strongest_characters AS
SELECT c.name, (c.strength + c.dexterity + c.constitution + c.intelligence) AS total_stats,
c.current_ap, c.health, SUM(cl.damage_done) AS total_damage_dealt
FROM character c
JOIN combat_log cl ON c.id = cl.character_id AND cl.action_type = 'cast spell' AND cl.outcome = 'hit'
GROUP BY c.name, c.strength, c.dexterity, c.constitution, c.intelligence, c.current_ap, c.health
ORDER BY total_stats DESC, c.current_ap DESC, total_damage_dealt DESC;
-- Pise statistika postav, summu parametrov, aktialne zdravie a AP a kolko spravili calkovo damage
```

	name character varying	total_stats integer	current_ap integer	health integer	total_damage_dealt bigint
1	Aragorn	43	55	50	52
2	Gendalf	35	39	40	90

Obrázok 27