

Projeto de um Banco de Dados de MMORPG

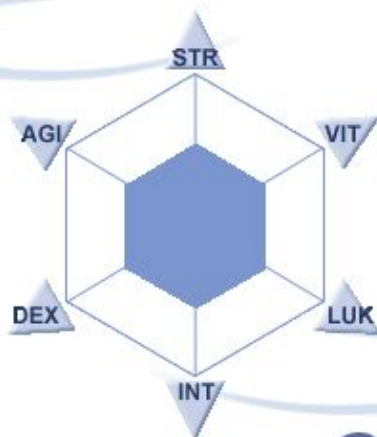
Felipe Genú
Vitor Augusto Pinheiro



Character Select



Name



STR	5
AGI	5
VIT	5
INT	5
DEX	5
LUK	5

Make Your Characters

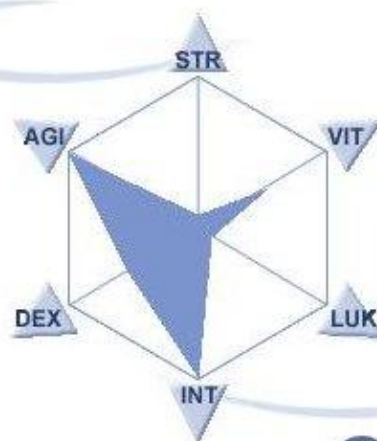
OK

cancel

캐릭터 만들기



Name



STR	1
AGI	9
VIT	5
INT	9
DEX	5
LUK	1

Make Your Characters

확인

취소



Knight



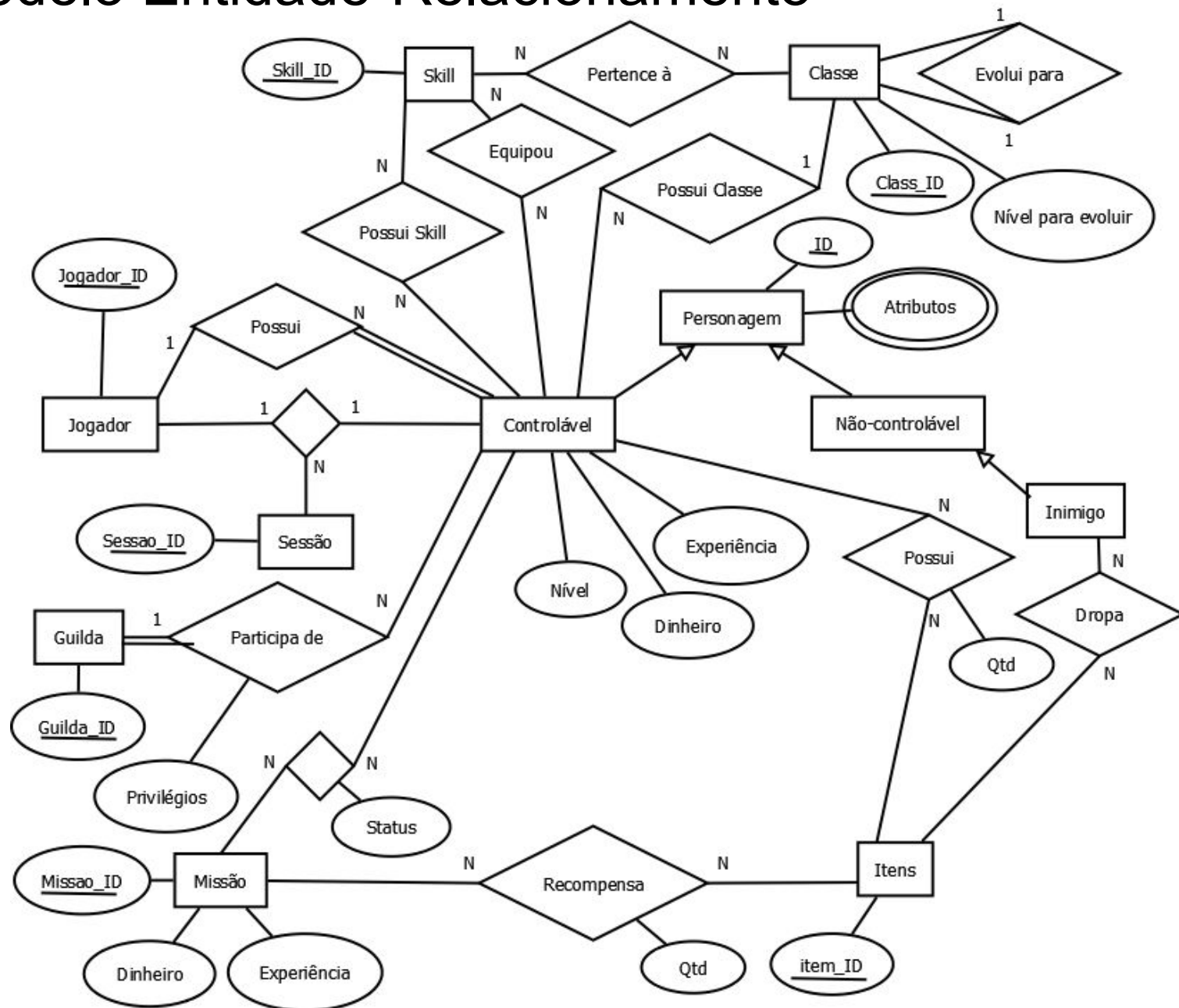
Lord Knight



Rune Knight
















Modelo Entidade-Relacionamento



Restrição 1:

Personagens controláveis recém-criados não podem possuir atributos acima do valor 5.

```
1 CREATE OR REPLACE FUNCTION check_initial_attributes() RETURNS trigger AS
2 $check_initial_attributes$
3 BEGIN
4     IF NEW.nivel = 1 THEN
5         IF (NEW.forca > 5 or NEW.agilidade > 5 or NEW.vitalidade > 5
6            or NEW.destreza > 5) THEN
7             RAISE EXCEPTION 'Personagem de jogador recém criado nao
8                pode ter mais de 5 em quaisquer atributos.';
9             RETURN NULL;
10        END IF;
11
12        IF (NEW.forca < 0 or NEW.agilidade < 0 or NEW.vitalidade < 0
13            or NEW.destreza < 0) THEN
14            RAISE EXCEPTION 'Personagem de jogador nao pode ter
15                menos de 0 em quaisquer atributos.';
16            RETURN NULL;
17        END IF;
18
19        RETURN NEW;
20    END IF;
21    RETURN NEW;
22 END;
23 $check_initial_attributes$ LANGUAGE plpgsql;
24
25 CREATE TRIGGER check_initial_attributes BEFORE INSERT ON public.
26 "Controlavel"
27 FOR EACH ROW EXECUTE PROCEDURE check_initial_attributes();
```

postgres on postgres@PostgreSQL 10

1

2

3

```
INSERT INTO public."Controlavel" VALUES (1056, 1, 1, 1, 6, 0, 500, 1, 100, 9001);  
SELECT * FROM public."Controlavel";
```

[Data Output](#) [Explain](#) [Messages](#) [Query History](#)

ERROR: Personagem de jogador recém criado nao pode ter mais de 5 em quaisquer atributos.
CONTEXT: PL/pgSQL function check_initial_attributes() line 7 at RAISE
SQL state: P0001

Restrição 2:

Personagens controláveis não devem possuir habilidades incompatíveis com sua classe.

```
1  -- Este serve apenas para mudanças na tabela possui_skill
2  CREATE OR REPLACE FUNCTION check_skill() RETURNS trigger AS $check_skill$
3      DECLARE
4          character_class integer;
5  BEGIN
6      -- Armazena id da classe do controlável numa variável.
7      character_class := (
8          SELECT id_classe
9          FROM public."Controlavel" AS c
10         WHERE c.id = NEW.id
11     );
12     -- Remove skills do personagem que não fazem parte da sua classe.
13     DELETE FROM public."Possui_Skill"
14     WHERE id_controlavel = NEW.id
15         AND id_skill NOT IN (
16             SELECT id_skill
17             FROM public."Skill_Pertence_A_Classe"
18             WHERE id_classe = NEW.id_classe
19         );
20     RETURN NEW;
21 END;
22 $check_skill$ LANGUAGE plpgsql;
```

```
25
26 -- Este serve apenas para mudanças na tabela possui skill
27 CREATE OR REPLACE FUNCTION check_skill_valid() RETURNS trigger AS $check_skill_valid$
28 DECLARE
29     character_class integer;
30 BEGIN
31     -- Armazena id da classe do controlável numa variável.
32     character_class := (
33         SELECT id_classe
34         FROM public."Controlavel" AS c
35         WHERE c.id = NEW.id_controlavel
36     );
37     -- Remove skills do personagem que não fazem parte da sua classe.
38     DELETE FROM public."Possui_Skill"
39     WHERE id_controlavel = NEW.id_controlavel
40         AND id_skill NOT IN (
41             SELECT id_skill
42             FROM public."Skill_Pertence_A_Classe"
43             WHERE id_classe = character_class
44         );
45     RETURN NEW;
46 END;
47 $check_skill_valid$ LANGUAGE plpgsql;
48
49
50
```

```

50
51 -- Este serve apenas para mudanças na tabela de skills pertence à classe.
52 CREATE OR REPLACE FUNCTION check_skill_still_valid() RETURNS TRIGGER AS $check_skill_still_v
53 DECLARE
54     linha record;
55 BEGIN
56     IF TG_OP = 'DELETE' THEN
57         linha = OLD;
58     ELSE
59         linha = NEW;
60     END IF;
61     DELETE FROM public."Possui_Skill"
62     WHERE id_skill = linha.id_skill
63     AND id_controlavel IN (
64         SELECT id
65         FROM (
66             SELECT id, id_classe
67             FROM public."Possui_Skill" AS s
68             INNER JOIN public."Controlavel" AS c
69             ON s.id_controlavel = c.id
70             WHERE s.id_skill = linha.id_skill
71
72         ) AS temp
73     WHERE temp.id_classe NOT IN (
74         SELECT id_classe
75         FROM public."Skill_Pertence_A_Classe"
76         WHERE id_skill = linha.id_skill
77     )
78 );
79     RETURN NEW;
80 END;
81 $check_skill_still_valid$ LANGUAGE plpgsql;
82

```

```
83  
84 CREATE TRIGGER check_skill_valid_class AFTER UPDATE ON public."Controlavel"  
85     FOR EACH ROW EXECUTE PROCEDURE check_skill();  
86
```

```
87 CREATE TRIGGER check_skill_valid_skill AFTER INSERT OR UPDATE ON public."Possui_Skill"  
88     FOR EACH ROW EXECUTE PROCEDURE check_skill_valid();  
89
```

```
90 CREATE TRIGGER check_skill_still_valid AFTER UPDATE OR DELETE ON public."Skill_Pertence_A_Clas  
91     FOR EACH ROW EXECUTE PROCEDURE check_skill_still_valid();
```



postgres on postgres@PostgreSQL 10

```
1 UPDATE public."Controlavel" SET id_classe = 102 WHERE id = 1002;  
2  
3 SELECT * FROM public."Possui_Skill";
```

	id_controlavel integer	id_skill integer
1	1001	1
2	1002	1
3	1002	101
4	1024	1
5	1024	200
6	1048	200

	id_controlavel integer	id_skill integer
1	1001	1
2	1002	101
3	1024	1
4	1024	200
5	1048	200



postgres on postgres@PostgreSQL 10

```
1 INSERT INTO public."Possui_Skill" VALUES (1048, 1);  
2  
3 SELECT * FROM public."Possui_Skill";
```

	id_controlavel integer	id_skill integer
1	1001	1
2	1002	101
3	1024	1
4	1024	200
5	1048	200

	id_controlavel integer	id_skill integer
1	1001	1
2	1002	101
3	1024	1
4	1024	200
5	1048	200



postgres on postgres@PostgreSQL 10

```
1 UPDATE public."Possui_Skill" SET id_skill = 200 WHERE id_skill = 1 AND id_controlavel = 1001;  
2  
3 SELECT * FROM public."Possui_Skill";
```

	id_controlavel integer	id_skill integer
1	1001	1
2	1002	101
3	1024	1
4	1024	200
5	1048	200

	id_controlavel integer	id_skill integer
1	1002	101
2	1024	1
3	1024	200
4	1048	200



postgres on postgres@PostgreSQL 10

```
1 DELETE FROM public."Skill_Pertence_A_Classe" WHERE id_skill = 200 AND id_classe = 200;  
2  
3 SELECT * FROM public."Possui_Skill";
```

	id_controlavel integer	id_skill integer
1	1001	1
2	1002	1
3	1002	101
4	1024	1
5	1024	200
6	1048	200

	id_controlavel integer	id_skill integer
1	1001	1
2	1002	1
3	1002	101
4	1024	1
5	1048	200

Restrição 3:

Personagens controláveis podem possuir no máximo 5 habilidades equipadas.

```
1 CREATE OR REPLACE FUNCTION check_skill_qty() RETURNS trigger AS $check_skill_qty$
2 DECLARE
3     qty integer;
4 BEGIN
5     qty := (
6         SELECT COUNT(id_skill)
7         FROM public."Equipou_Skill"
8         WHERE id_controlavel = NEW.id_controlavel
9     );
10    IF qty >= 5 THEN
11        RAISE EXCEPTION 'Número máximo de skills equipada. Desequipe uma skill para poder equ
12    ELSE
13        RETURN NEW;
14    END IF;
15 END;
16 $check_skill_qty$ LANGUAGE plpgsql;
17
18 CREATE TRIGGER check_skill_qty BEFORE INSERT OR UPDATE ON public."Equipou_Skill"
19     FOR EACH ROW EXECUTE PROCEDURE check_skill_qty();
```

No limit














postgres on postgres@PostgreSQL 10

```
1 INSERT INTO public."Possui_Skill"
2 VALUES (1001, 1001),
3         (1001, 1002),
4         (1001, 1003),
5         (1001, 1004),
6         (1001, 1005);
7 SELECT * FROM public."Possui_Skill" WHERE id_controlavel = 1001;
```

Data Output Explain Messages Query History

	id_controlavel integer	id_skill integer
1	1001	1
2	1001	1001
3	1001	1002
4	1001	1003
5	1001	1004
6	1001	1005

✓ Successfully r



postgres on postgres@PostgreSQL 10













```
1 INSERT INTO public."Equipou_Skill"
2 VALUES (1001, 1001),
3          (1001, 1002),
4          (1001, 1003),
5          (1001, 1004);
6 SELECT * FROM public."Equipou_Skill" WHERE id_controlavel = 1001;
```

Data Output Explain Messages Query History







	id_controlavel integer	id_skill integer
1	1001	1
2	1001	1001
3	1001	1002
4	1001	1003
5	1001	1004

✓

Successfully run. 7



No limit



postgres on postgres@PostgreSQL 10

```
1 INSERT INTO public."Equipou_Skill"
2 VALUES (1001, 1001),
3         (1001, 1002),
4         (1001, 1003),
5         (1001, 1004),
6         (1001, 1005);
7
8 SELECT * FROM public."Equipou_Skill" WHERE id_controlavel = 1001;
```

Data Output Explain Messages Query History

ERROR: Número máximo de skills equipada. Desequipe uma skill para poder equipar outra.
CONTEXT: PL/pgSQL function check_skill_qty() line 11 at RAISE
SQL state: P0001

Processamento 1:

Mudança de nível do personagem controlável conforme o ganho de experiência, dado conforme uma fórmula.

```
1  CREATE OR REPLACE FUNCTION level_up() RETURNS TRIGGER AS $$
2      DECLARE
3          new_level integer;
4  BEGIN
5      IF NEW.experiencia <> OLD.experiencia THEN
6          new_level = 1 + floor(0.1 * |/NEW.experiencia);
7          IF new_level > NEW.nivel THEN
8              NEW.nivel = new_level;
9          END IF;
10         END IF;
11         RETURN NEW;
12     END;
13 $$ LANGUAGE plpgsql;

15 CREATE TRIGGER level_up BEFORE UPDATE ON public."Controlavel"
16     FOR EACH ROW EXECUTE PROCEDURE level_up();
```

Processamento 2:

Mudança de classe conforme o aumento do nível do personagem controlável.

```
1  CREATE OR REPLACE FUNCTION class_up() RETURNS trigger AS $$
2  DECLARE
3      current_class record;
4  BEGIN
5      IF NEW.nivel <> OLD.nivel THEN
6          -- pegar o nível da classe que se deve upar
7          SELECT proxima_classe, nivel_proxima_classe INTO current_class
8              FROM public."Classe"
9              WHERE id = NEW.id_classe;
10         -- comparar se o controlavel atingiu esse nivel
11         IF current_class.proxima_classe IS NOT NULL THEN
12             IF NEW.nivel >= current_class.nivel_proxima_classe THEN
13                 UPDATE public."Controlavel"
14                     SET id_classe = current_class.proxima_classe
15                     WHERE id = NEW.id;
16                 -- caso sim, mudar o nível e desequipar todas as skills que a classe não possui
17                 DELETE FROM public."Possui_Skill" AS ps
18                     WHERE ps.id_controlavel = NEW.id
19                     AND ps.id_skill NOT IN (
20                         SELECT id_skill
21                         FROM public."Skill_Pertence_A_Classe"
22                         WHERE id_classe = current_class.proxima_classe
23                     );
24             END IF;
25         END IF;
26         -- caso não, acaba o programa
27     END IF;
28     RETURN NEW;
29 END;
30 $$ LANGUAGE plpgsql;
31
32 CREATE TRIGGER class_up AFTER UPDATE ON public."Controlavel"
33     FOR EACH ROW EXECUTE PROCEDURE class_up();
```

No limit



postgres on postgres@PostgreSQL 10

```
1 SELECT * FROM public."Controlavel";
```

Data Output

[Explain](#)

[Messages](#)

[Query History](#)

	Id integer	experiencia integer	dinheiro integer	nivel integer	forca integer	agilidade integer	vitalidade integer	destreza integer	Id_classe integer	Id_jogador integer
1	1001	0	0	1	1	2	3	4	100	9001
2	1002	0	0	21	2	3	4	5	101	9001
3	1024	0	0	19	3	4	5	6	200	9002
4	1048	0	0	40	4	5	6	7	202	9003



Successfully run. Total query runtime: 64 ms

📁

💾

▼

🔍

▼

📄

📄

🗑️

📝

▼

🔍

▼

No limit

▼

⚡

▼

■

▼

📝

▼

📥

🔗

postgres on postgres@PostgreSQL 10

1

UPDATE public."Controlavel" SET experiencia = (experiencia + 40001) WHERE id = 1024;

2

3

SELECT * FROM public."Controlavel" WHERE id = 1024;

Data Output

Explain

Messages

Query History

	id	experiencia	dinheiro	nivel	forca	agilidade	vitalidade	destreza	id_classe	id_jogador
▲	integer	integer	integer	integer	integer	integer	integer	integer	integer	integer
1	1024	40001	0	21	3	4	5	6	201	9002

✓

Successfully run. Total query runtime: 88 ms

