VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta informačních technologií

DATABÁZOVÉ SYSTÉMY

2019/2020

Projektová dokumentace

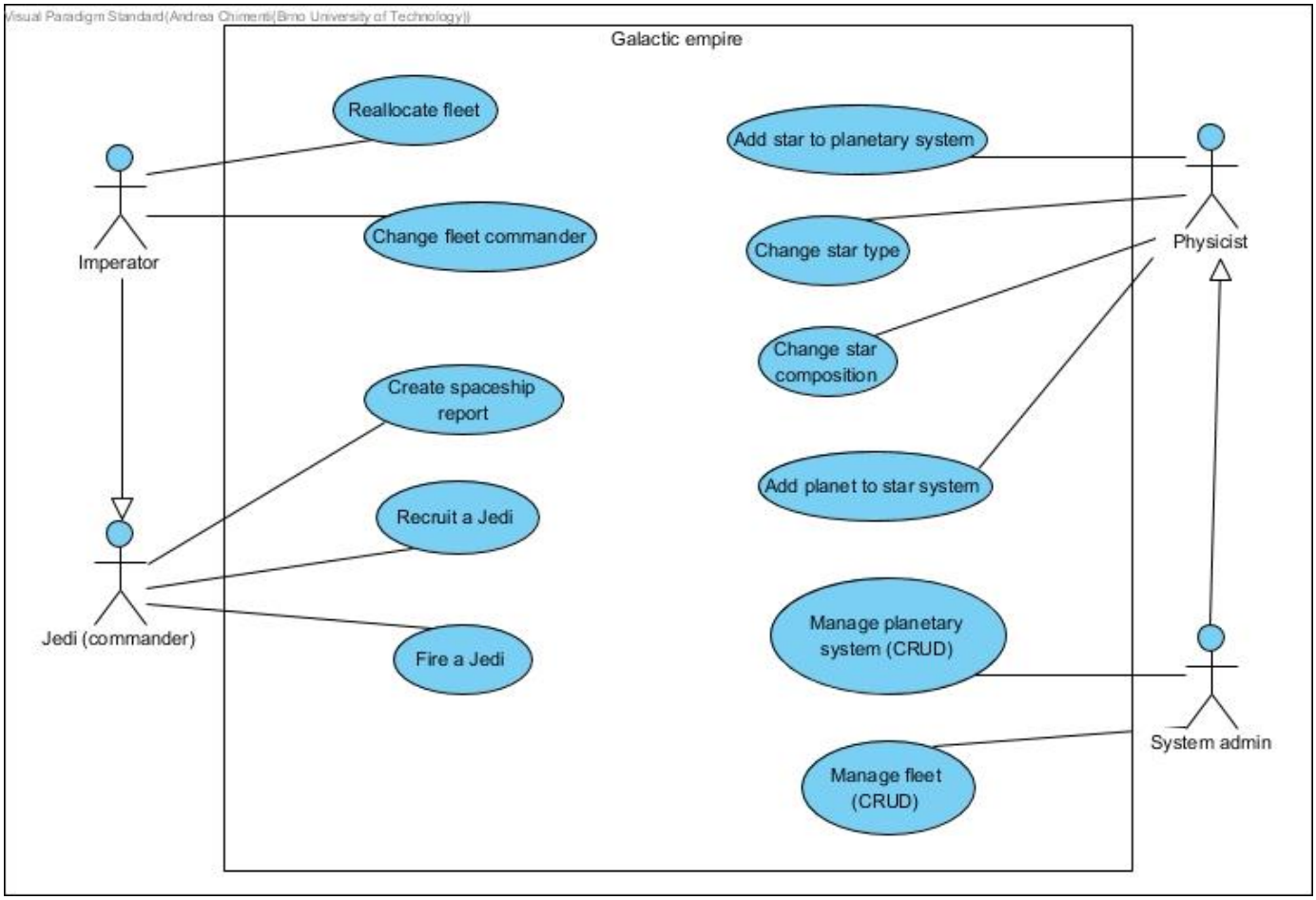
Galaktické impérium

Andrea Chimenti (xchime00)

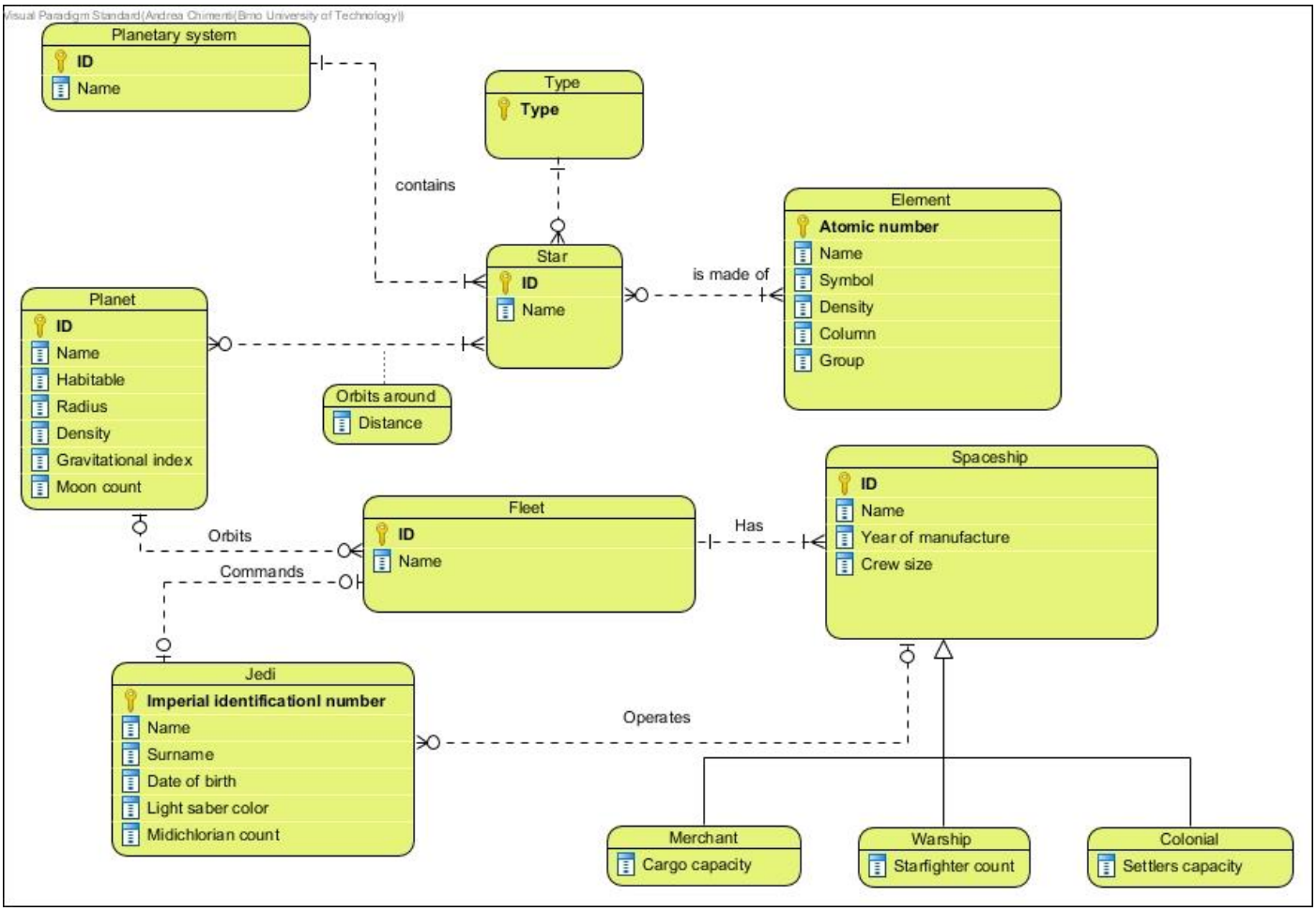
Jan Klhůfek (xklhuf01)

Brno, 27. dubna 2020

1. USE CASE DIAGRAM



2. ENTITY RELATIONSHIP DIAGRAM



3. IMPLEMENTAČNÍ DETAILY

Skript je rozdělen do několika částí, každá z nich je popsána níže. Pořadí v dokumentaci odpovídá pořadí ve skriptu.

DROP

Při opakovaném spuštění skriptu, je nejprve potřeba uvolnit z databáze již existující objekty.

CREATE TABLES

Vytvoření schématu databáze podle ER diagramu. Generalizace / specializace u vesmírných lodí je disjunktní a částečná. Pro implementaci byla tedy zvolena metoda jedné tabulky, která zajišťuje že loď může být vždy jen jednoho typu (případně obecného). Správné vyplnění atributů s ohledem na konkrétní typ lodě je hlídáno omezeními definovanými ve schématu tabulky lodě.

INSERT

Naplnění tabulek demonstračními daty.

SELECT

Demonstrace použití příkazů `select`.

TRIGGERS

`star_insertion`:
Trigger, který je spuštěn vždy před operací `insert` v tabulce `star`. Jeho cílem je zajistit aby (v případě že `id` nebylo zadáno ručně) bylo automaticky vygenerováno nové `id`, které v tabulce ještě neexistuje. K tomuto účelu je použita sekvence `star_pk_num`, která generuje posloupnost čísel od jedničky. V případě že se vygeneruje `id` které již bylo zabráno trigger vygeneruje další hodnotu dokud se nepodaří nalézt první volné.

`jedi_on_board`:
Trigger, který zajišťuje aktualizaci velikosti posádky na lodi v případě přesunutí jediho. Pokud je na loď přidán jedi, posádka se zvětší o jedna, pokud je jedi odebrán, posádka se zmenší o jedna.

PROCEDURES

`count_habitable_planets_in_system ("ps_id" NUMBER):`
Procedura spočítá počet obyvatelných planet v planetárním systému určeného parametrem `ps_id`. Pomocí kurzoru `habitable_planet_count` se nejprve najdou všechny záznamy o obyvatelných planetách v systému a následně se v těle procedury spočítá jejich počet. Výsledkem je název planetárního systému s příslušejícím počtem obyvatelných planet. V případě zadání neexistujícího `ps_id` dojde k chybě.

`jedis_in_fleet ("f_id" NUMBER, "midi_count" NUMBER):`
Procedura spočítá a vypíše kolik jediů, s počtem midi-chlorianů vyšším než `midi_count`, se nachází na lodích náležících flotile s identifikačním číslem `f_id`. Procedura definuje svou vlastní výjimku `NO_JEDI_FOUND`, která je vyvolána v případě, že nebyl nalezen žádný Jedi s požadovanými vlastnostmi. V případě zadání neexistující `f_id` nebo příliš nízkého `midi_count` (jedi musí mít alespoň 7000 midi-chlorianů), dojde k chybě.

EXPLAIN PLANS

Explain plan neprovede dotaz, ale jen vyvolá optimalizátor. Výsledek je uložen do systémové tabulky `plan_table`. Plán využívá databázového dotazu, který pro každou flotilu vypíše součet členů posádek u jednotlivých typů lodí. Pro lepší optimalizaci byl vytvořen index `fleet_spaceships`, skládající se ze 3 sloupců, které jsou nejčastějšími predikáty dotazu. Bylo důležité indexované sloupce uvést v pořadí od nejvíce dotazovaného po nejméně, tedy `fleet_id, type, crew_size`. Po použití indexu činí úspora cpu 30,30 %. Změna indexu se zdá být nejlepším řešením pro optimalizaci, jelikož má větší sílu než změna SQL dotazu či změna dat.

MATERIALIZED VIEW

`fleets_orbiting_planet:`
Materializovaný pohled využívá dotazu `select`, který pro každou planetu v tabulce `planet` spočítá počet flotil na jejím orbitě. Po vytvoření materializovaného pohledu se výsledek dotazu uloží lokálně a je přístupný jen pro čtení. V případě modifikace dat na serveru se změna v materializovaném pohledu projeví až po zavolání operace `commit`.

PERMISSIONS

Udělení přístupových práv druhému členovi týmu (`xklhuf01`).