

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ
НАПРАВЛЕНИЕ СИСТЕМНОГО И ПРИКЛАДНОГО ПРОГРАММНОГО
ОБЕСПЕЧЕНИЯ

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1
курса «Информационные системы и базы данных»
по теме: «Построение инфологической и даталогической модели»
Вариант № 335047

Выполнил студент:
Тюрин Иван Николаевич
группа: Р33102

Преподаватель:
Сагадак Алина Алексеевна

Санкт-Петербург, 2023 г.

Содержание

Лабораторная работа № 1. Построение инфологической и дата-логической модели	2
1. Задание варианта № 335047	2
2. Выполнение задания	3
3. Вывод	8

Лабораторная работа № 1

Построение инфологической и даталогической модели

1. Задание варианта № 335047

’ ’ ’

Составить инфологическую и даталогическую модели по следующему тексту и реализовать базу данных PostgreSQL на helios.se.ifmo.ru.

*Элли знала, что на заре истории нашей планеты в растительном мире царили такие же законы борьбы за выживание, как и в животном, а порой даже и более жестокие. Яд, который вырабатывали *Serpena veriformans*, был всего лишь незначительным примером многообразного арсенала "химического оружия" растений. Например, некоторые растения выделяют терпены /Органические соединения, находящиеся в смоле хвойных деревьев/, пропитывая почву ядом и подавляя тем самым рост других растений.*

’ ’ ’

2. Выполнение задания

Текст взят из рассказа про фантастический «Парк Юрского периода». Пришлось посмотреть окружающий текст, чтобы понять, что является предметной областью в моем случае. Среди важных сущностей я выделил следующие:

- Character – **стержневая** сущность, персоны из книги, действующие лица. Например, Элли, «другие люди», профессор Дуглас.
- Location – **характеристическая** сущность, локация в которой что-то происходит. Например, бассейн, лес и пр.
- Biology Entity – **стержневая** сущность, сущность в иерархии живых существ, те кто представлены в Парке Юрского периода. Например, *Selenna Veriformans*.
- Biology Hierarchy – **характеристическая** сущность, иерархии, к которым принадлежат сущности представленные в книге.
- Live Epoch – **характеристическая** сущность, эпоха, в которой относятся сущности, из какой эпохи их возродили на острове.
- Protection Method – **ассоциативная** сущность, метод защиты биологических существ от других биологических существ, средства о которых идет речь в тексте.
- Character Join Biology Entity – **ассоциативная** сущность, объединяющая Character и Biology Entity

Так же определил связи между выделенными сущностями.

- Character знает одного или нескольких Biology Entity, находится в какой-то Location, имеет имя.
- Location задает название локации.
- Biology Entity имеет определенное место в иерархии живых существ, имеет один или несколько Protection Method от других существ, жило в какой-то определенной Live Epoch и располагается в данный момент в Location.
- Biology Hierarchy определяет название иерархии и какие-то нестрогие структурированные атрибуты этой иерархии.
- Live Epoch определяет название эпохи и ее временной интервал с помощью нестрогих структурированных атрибутов.

- Protection Method определяет связь между двумя живыми существами и свойства этой связи.

Сущности были представлены на концептуальной (инфологической) и даталогической модели в виде ER-диаграммы (см. [1.1](#)).

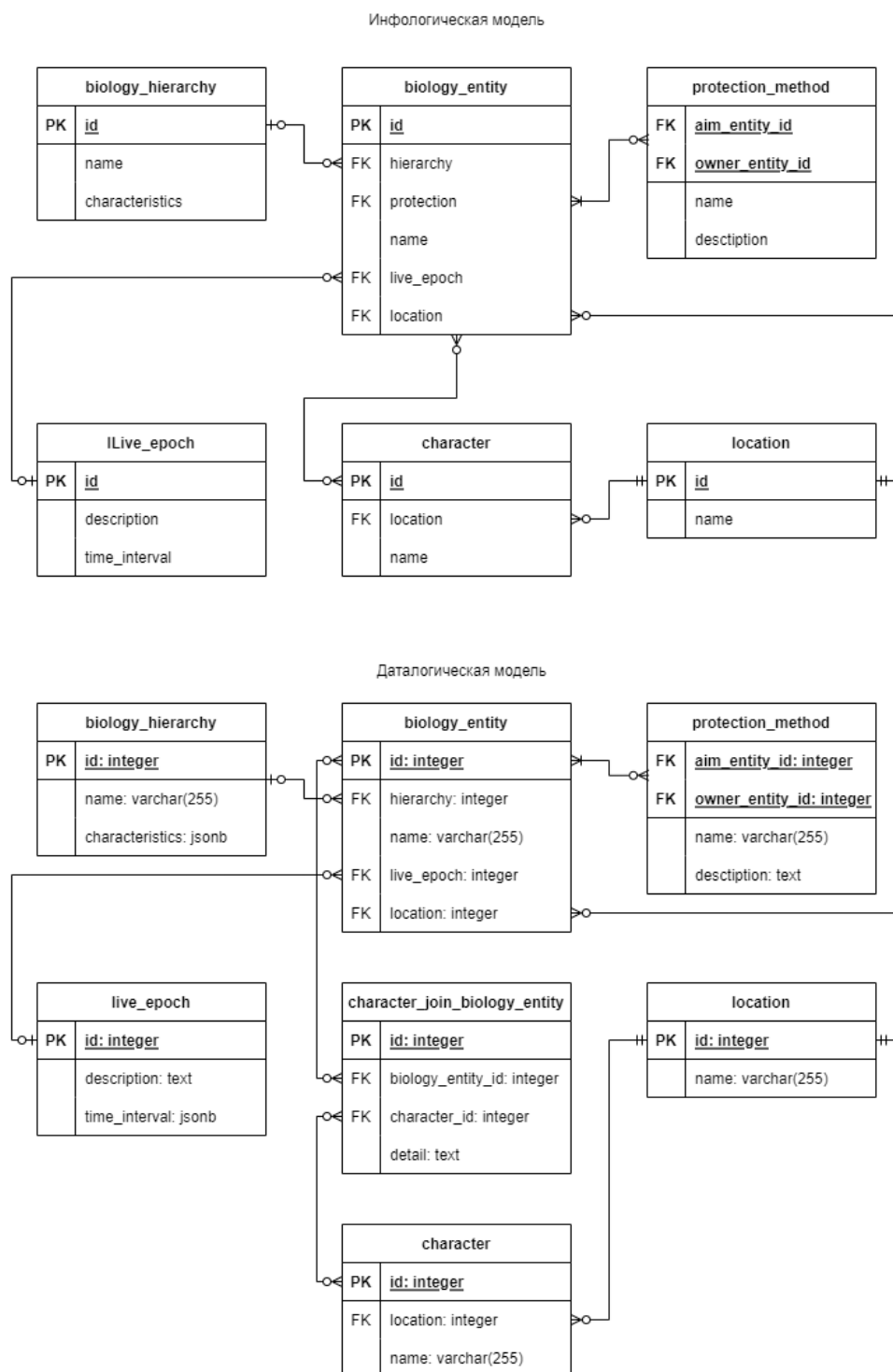


Рис. 1.1: ER-диаграмма инфологической и даталогическая модели

SQL скрипты для создания нужных таблиц представлены в листингах 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7.

```

1 CREATE TABLE IF NOT EXISTS "character"
2 (
3     "id" integer PRIMARY KEY GENERATED ALWAYS AS IDENTITY,
4     "name" character varying(255) NOT NULL,
5     "location" integer,
6     CONSTRAINT "location_fk"
7     FOREIGN KEY ("location")
8         REFERENCES "location" ("id")
9         ON DELETE RESTRICT
10 );
11
12 ALTER TABLE IF EXISTS "character"
13     OWNER TO s335047;

```

Листинг 1.1: DDL Character

```

1 CREATE TABLE IF NOT EXISTS "biology_entity"
2 (
3     "id" integer PRIMARY KEY GENERATED ALWAYS AS IDENTITY,
4     "hierarchy" integer,
5     "name" character varying(255) NOT NULL,
6     "live_epoch" integer,
7     "location" integer NOT NULL,
8     CONSTRAINT "biology_hierarchy_fk"
9     FOREIGN KEY ("hierarchy")
10         REFERENCES "biology_hierarchy" ("id")
11         ON DELETE NO ACTION,
12     CONSTRAINT "live_epoch_fk"
13     FOREIGN KEY ("live_epoch")
14         REFERENCES "live_epoch" ("id")
15         ON DELETE NO ACTION,
16     CONSTRAINT "location_fk"
17     FOREIGN KEY ("location")
18         REFERENCES "location" ("id")
19         ON DELETE RESTRICT
20 );
21
22 ALTER TABLE IF EXISTS "biology_entity"
23     OWNER TO s335047;

```

Листинг 1.2: DDL Biology Entity

```

1 CREATE TABLE If NOT EXISTS "biology_hierarchy"
2 (
3     "id" integer PRIMARY KEY GENERATED ALWAYS AS IDENTITY,
4     "name" character varying(255) NOT NULL,
5     "characteristics" jsonb
6 );
7
8 ALTER TABLE IF EXISTS "biology_hierarchy"
9     OWNER TO s335047;

```

Листинг 1.3: DDL Biology Hierarchy

```

1 CREATE TABLE IF NOT EXISTS "live_epoch"
2 (
3     "id" integer CONSTRAINT "live_epoch_pk" PRIMARY KEY GENERATED ALWAYS
4     AS IDENTITY,
5     "description" text NOT NULL,
6     "time_interval" jsonb
7 );
8 ALTER TABLE IF EXISTS "live_epoch"
9     OWNER TO s335047;

```

Листинг 1.4: DDL Live Epoch

```

1 CREATE TABLE IF NOT EXISTS "location"
2 (
3     "id" integer PRIMARY KEY GENERATED ALWAYS AS IDENTITY,
4     "name" character varying(255) NOT NULL
5 );
6
7 ALTER TABLE IF EXISTS "location"
8     OWNER TO s335047;

```

Листинг 1.5: DDL Location

```

1 CREATE TABLE IF NOT EXISTS "protection_method"
2 (
3     "aim_entity_id" integer NOT NULL,
4     "owner_entity_id" integer NOT NULL,
5     "name" character varying(255) NOT NULL,
6     "description" text NOT NULL,
7     CONSTRAINT "aim_entity_fk"
8     FOREIGN KEY ("aim_entity_id")
9         REFERENCES "biology_entity" ("id")
10        ON DELETE CASCADE,
11     CONSTRAINT "owner_entity_fk"
12     FOREIGN KEY ("owner_entity_id")
13        REFERENCES "biology_entity" ("id")
14        ON DELETE CASCADE,
15     CONSTRAINT "protection_method_pk"
16     PRIMARY KEY ("aim_entity_id", "owner_entity_id")
17 );
18
19 ALTER TABLE IF EXISTS "protection_method"
20     OWNER TO s335047;

```

Листинг 1.6: DDL Protection Method


```

1 CREATE TABLE IF NOT EXISTS "character_join_biology_entity"
2 (
3     "id" integer PRIMARY KEY GENERATED ALWAYS AS IDENTITY,
4     "character_id" integer NOT NULL,
5     "biology_entity_id" integer NOT NULL,
6     "detail" text,
7     CONSTRAINT "character_id_fk" FOREIGN KEY ("character_id")
8         REFERENCES "character" ("id")
9         ON DELETE CASCADE,
10    CONSTRAINT "biology_entity_id_fk" FOREIGN KEY ("biology_entity_id")
11        REFERENCES "biology_entity" ("id")
12        ON DELETE CASCADE
13 );
14
15 ALTER TABLE IF EXISTS "character_join_biology_entity"
16     OWNER TO s335047;

```

Листинг 1.7: DDL Character Join Biology Entity

Написанные скрипты выполнялись с помощью **psql** в базе данных **studs**. Для этого нужно было пробросить порт с локальной машины на сервер и запустив утилиту, выполнить команду `\i` с указанием скрипта, который должен быть выполнен.

Дальнейшее заполнение таблиц данными выполнялось с помощью графического приложения pgAdmin и с использованием SQL/DML скриптов (см. 1.8).

```

1 INSERT INTO "character" (name, location)
2 SELECT 'Элли', loc.id
3 FROM (SELECT id FROM "location" WHERE name LIKE 'Бассейн') loc;
4
5 INSERT INTO "character" (name, location)
6 SELECT 'Дуглас', loc.id
7 FROM (SELECT id FROM "location" WHERE name LIKE 'В пределах планеты Земля')
8     loc;

```

Листинг 1.8: DML Add Character

3. Вывод

В результате выполнения работы изучил правилами составления инфологических и даталогичских моделей, написания SQL скриптов для СУБД PostgreSQL; поработал с утилитой **psql** и приложением pgAdmin для управления удаленной БД.

```

1 INSERT INTO "biology_entity" ("hierarchy", "name", "live_epoch", "location
   ")
2 SELECT h."id", 'Папоротник', le."id", loc."id"
3 FROM (SELECT id FROM "biology_hierarchy" WHERE "name" LIKE 'Папоротники') h,
4      (SELECT id FROM "live_epoch" WHERE "description" LIKE 'очень давно') le
5      ,
6      (SELECT id FROM "location" WHERE "name" LIKE 'Бассейн') loc;
7
8 INSERT INTO "biology_entity" ("hierarchy", "name", "live_epoch", "location
   ")
9 SELECT h."id", 'Ели Дугласа', le."id", loc."id"
10 FROM (SELECT id FROM "biology_hierarchy" WHERE "name" LIKE 'Ели Дугласа') h,
11      (SELECT id FROM "live_epoch" WHERE "description" LIKE 'Настоящее время'
12      ) le,
13      (SELECT id FROM "location" WHERE "name" LIKE 'Лес') loc;
14
15 INSERT INTO "biology_entity" ("hierarchy", "name", "live_epoch", "location
16      ")
17 SELECT h."id", 'Насекомые', le."id", loc."id"
18 FROM (SELECT id FROM "biology_hierarchy" WHERE "name" LIKE 'Насекомые') h,
19      (SELECT id FROM "live_epoch" WHERE "description" LIKE 'очень давно') le
20      ,
21      (SELECT id FROM "location" WHERE "name" LIKE 'Лес') loc;

```

Листинг 1.9: DML Add Biology Entity

```

1 INSERT INTO "biology_hierarchy"
2 VALUES (DEFAULT, 'Папоротники', 'царство{"": растения"", род"": папоротники""}'');
3
4 INSERT INTO "biology_hierarchy"
5 VALUES (DEFAULT, 'Хищники', '{ класс"": хищники"" }');
6
7 INSERT INTO "biology_hierarchy"
8 VALUES (DEFAULT, 'Ели Дугласа', 'царство{"": растения"", род"": ели""}'');
9
10 INSERT INTO "biology_hierarchy"
11 VALUES (DEFAULT, 'Насекомые', 'царство{"": насекомые""}'');

```

Листинг 1.10: DML Add Biology Hierarchy

```

1 INSERT INTO "live_epoch" ( "description", "time_interval")
2 VALUES ( 'очень давно', 'прям{"оченьдавно__": рил"}');
3
4 INSERT INTO "live_epoch" ( "description", "time_interval")
5 VALUES ( 'kek', '{"epoch": "lol"}');
6
7 INSERT INTO "live_epoch" ( "description", "time_interval")
8 VALUES ( 'Настоящее время', '{"epoch": "now"}');
9
10 INSERT INTO "live_epoch" ( "description", "time_interval")
11 VALUES ( 'Not stated', '{"epoch": null}');

```

Листинг 1.11: DML Add Live Epoch

```

1 INSERT INTO "location" ("name")
2 VALUES ('Бассейн');
3
4 INSERT INTO "location" ("name")
5 VALUES ('Лес');
6
7 INSERT INTO "location" ("name")
8 VALUES ('В пределах планеты Земля');

```

Листинг 1.12: DML Add Location

```

1 INSERT INTO "protection_method" ( aim_entity_id, owner_entity_id, name,
   description)
2 SELECT aim.id, owner.id, 'феромоны', 'отпугивают насекомых, сообщают другим
   деревьям об опасности'
3 FROM (SELECT id FROM "biology_entity" WHERE name LIKE 'Ели Дугласа') aim,
4      (SELECT id from "biology_entity" WHERE name LIKE 'Насекомые') owner;

```

Листинг 1.13: DML Add Protection Method

```

1 INSERT INTO "character_join_biology_entity" (character_id,
   biology_entity_id)
2 SELECT ch.id, be.id
3 FROM (SELECT id FROM "character" WHERE name LIKE 'Элли') ch,
4      (SELECT id FROM "biology_entity" WHERE name LIKE 'Папоротник') be

```

Листинг 1.14: DML Add Character Join Biology Entity

Литература

- [1] Ссылка на личный репозиторий GitHub: <https://github.com/elturin/itmo-is-db/tree/main/lab-1>