



Факультет программной инженерной и компьютерной техники
Базы Данных

Лабораторная работа №1

Вариант 10011

Выполнил: Алхимовици Арсений 408138

P3110

Санкт-Петербург, 2023

Условие и описание Предметной области

климатических зонах. Нам бы хотелось знать, с чем это связано.

Грант быстро просмотрел карты. Если действительно Фонд поддерживает только раскопки в холодных зонах, то это странно, потому что лучшие специалисты по динозаврам работают в жарких странах.

— И здесь есть еще загадки, — продолжал Моррис. — Например, какое отношение имеют динозавры к янтарю?

Из следующих предложений берем еще сущность Руды, которые имеют хар-ку price. Каждая руда это предмет по этому он принимает item_id

Введите вариант: 10011

Описание предметной области, по которой должна быть построена доменная модель:

Грант быстро просмотрел карты. Если действительно Фонд поддерживает только раскопки в холодных зонах, то это странно, потому что лучшие специалисты по динозаврам работают в жарких странах.

Есть люди у которых есть имена, существуют действия (посмотрел, поддерживает, работать) у которых есть характеристика description и он принимает 2 сущности сразу (от кого действие(NOT NULL) и к кому(может быть NULL). Есть локации (холодные зоны, жаркие страны). Существует работа (специалист) и хар-ка description у нее (по динозаврам). Существуют предметы(карты, раскопки)

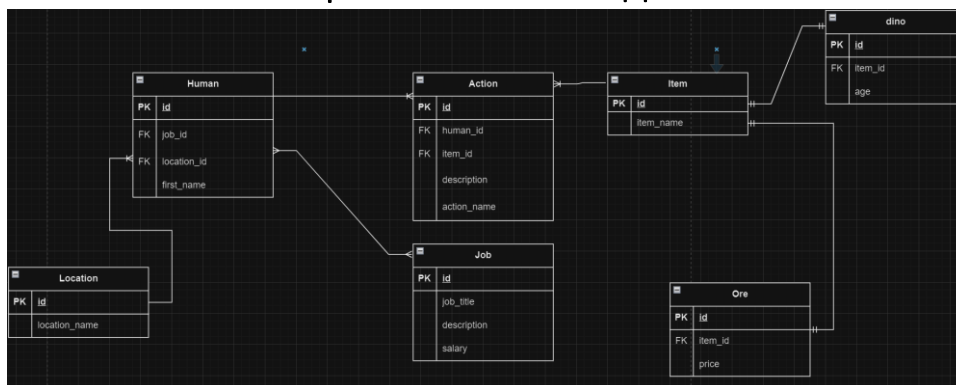
Список сущностей и их классификацию

Стержневая: human (имеет имя, и id на локацию и работу), location_table(локация имеет имя), item (предмет имеет имя)

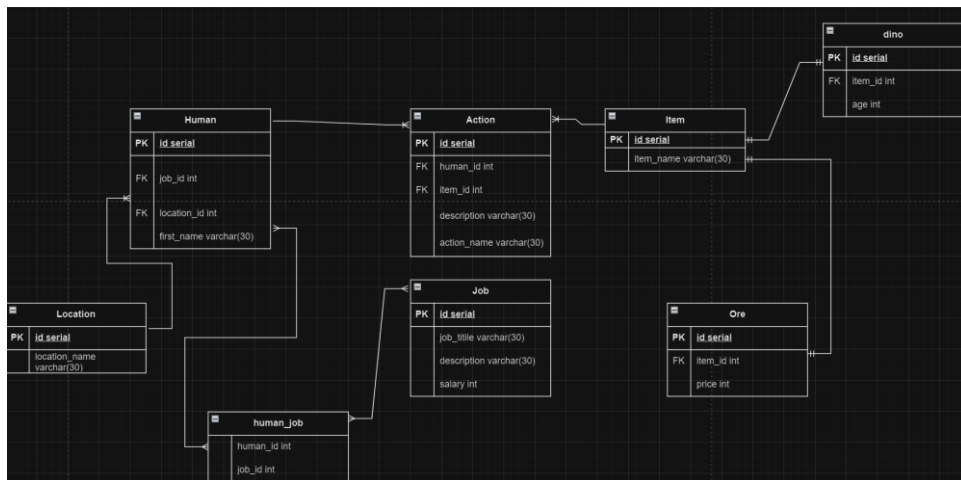
Ассоциация: job (имеет название, дополнение например “специалист по динозаврам” и значение зарплаты), action_table (имеет название, дополнение, а так же строит связь между человеком и предметом)

Характеристика: ore (характеристика предмета руда имеет цену больше нуля), dino (характеристика предмета останки динозавра имеет возраст больше нуля)

Инфологическая модель



Даталогическая модель



Исходный код программы

```
-- delete old tables

DROP TABLE IF EXISTS human, location_table, action_table, job, item, dino,
ore, human_job;
```

```
--create new tables

CREATE TABLE location_table
(
  id serial PRIMARY KEY,
  location_name VARCHAR(30) NOT NULL
);
```

```
CREATE TABLE job
(
  id serial PRIMARY KEY,
  job_title VARCHAR(30) NOT NULL,
  description VARCHAR(30),
  salary int,
  CHECK(salary >= 0)
);
```

```
CREATE TABLE human
(
  id serial PRIMARY KEY,
  job_id int REFERENCES job(id),
  location_id int REFERENCES location_table(id) NOT NULL,
  first_name VARCHAR(30) NOT NULL
);
```

```
CREATE TABLE item
```

```
(  
id serial PRIMARY KEY,  
item_name varchar(30) NOT NULL  
);
```

```
CREATE TABLE action_table  
(  
id serial PRIMARY KEY,  
human_id int REFERENCES human(id),  
item_id int REFERENCES item(id),  
description VARCHAR(30),  
action_name VARCHAR(30) NOT NULL  
);
```

```
CREATE TABLE ore  
(  
id serial PRIMARY KEY,  
item_id int REFERENCES item(id),  
price int,  
CHECK(price >= 0),  
description VARCHAR(30),  
action_name VARCHAR(30)  
);
```

```
CREATE TABLE dino  
(  
id serial PRIMARY KEY,  
item_id int REFERENCES item(id),  
age int NOT NULL,  
CHECK(age>=0)  
);
```

--many-to-many таблицы

```
CREATE TABLE human_job
```

```
(  
human_id int REFERENCES human(id),  
job_id int REFERENCES job(id)  
);
```

--fill tables

```
INSERT INTO location_table (location_name)  
VALUES  
( 'холодные зоны'),  
( 'жаркие страны');
```

```
INSERT INTO job (job_title, description, salary)  
VALUES  
( 'специалист', 'по динозаврам', 10),  
( 'палеонтолог', NULL, 5000);
```

```
INSERT INTO human (job_id, location_id, first_name)  
VALUES  
(2, 1, 'Грант'),  
(1, 2, 'Человек');
```

```
INSERT INTO item (item_name)  
VALUES  
( 'карты'),  
( 'скелет динозавра');
```

```
INSERT INTO action_table (human_id, item_id, description, action_name)  
VALUES  
(1, 1, 'быстро', 'посмотреть'),  
(2, NULL, NULL, 'работать');
```

```
(2, 2, NULL, 'искать');
```

```
INSERT INTO dino (item_id, age)
```

```
VALUES
```

```
(2, 1500);
```

```
INSERT INTO human_job (human_id, job_id)
```

```
VALUES
```

```
(1, 2),
```

```
(2, 1);
```

Вывод

В таблицах можно удобно проверять значения, например возраст больше 0 и тд.

Foreign Key помогают нам связать разные таблицы между собой. Many-to-many связь надо отображать в дополнительной таблице с primary key, каждого из таблиц.