**Задание**

**Операционные системы и виртуализация (Linux)**

1. Использование команды cat в Linux

- Создать два текстовых файла: "З"(Домашние животные) и "Pack animals"(вьючные животные), используя команду `cat` в терминале Linux. В первом файле перечислить собак, кошек и хомяков. Во втором — лошадей, верблюдов и ослов.

- Объединить содержимое этих двух файлов в один и просмотреть его содержимое.

- Переименовать получившийся файл в "Human Friends"(.

Пример конечного вывода после команды “ls” :

Desktop Documents Downloads HumanFriends.txt Music PackAnimals.txt Pets.txt Pictures Videos

Выполнение:

cat > Pets.txt

cat Pets.txt

cat Pets.txt PackAnimals.txt > Animals.txt

mv Animals.txt HumanFriends.txt

ls

2. Работа с директориями в Linux

- Создать новую директорию и переместить туда файл "Human Friends".

**Выполнение:**

*mkdir HumanAnimals*

*mv HumanFriends.txt HumanAnimals/*

*ls HumanAnimals*

3. Работа с MySQL в Linux. “Установить MySQL на вашу вычислительную машину ”

Выполнение:

Следуя описанию <https://dev.mysql.com/doc/mysql-apt-repo-quick-guide/en/#repo-qg-apt-repo-manual-setup> подключим дополнительный репозиторий MySQL вручную

В файл с именем /etc/apt/sources.list.d/mysql.list  поместим записи репозитория:

*sudo nano /etc/apt/sources.list.d/mysql.list deb http://repo.mysql.com/apt/ubuntu/ jammy mysql-8.0*

Создаем файл ключа

*sudo nano mysql\_pubkey.asc*

-----BEGIN PGP PUBLIC KEY BLOCK-----

Version: SKS 1.1.6

Comment: Hostname: pgp.mit.edu

mQINBGU2rNoBEACSi5t0nL6/Hj3d0PwsbdnbY+SqLUIZ3uWZQm6tsNhvTnahvPPZBGdl99iW

YTt2KmXp0KeN2s9pmLKkGAbacQP1RqzMFnoHawSMf0qTUVjAvhnI4+qzMDjTNSBq9fa3nHmO

YxownnrRkpiQUM/yD7/JmVENgwWb6akZeGYrXch9jd4XV3t8OD6TGzTedTki0TDNr6YZYhC7

jUm9fK9Zs299pzOXSxRRNGd+3H9gbXizrBu4L/3lUrNf//rM7OvV9Ho7u9YYyAQ3L3+OABK9

FKHNhrpi8Q0cbhvWkD4oCKJ+YZ54XrOG0YTg/YUAs5/3//FATI1sWdtLjJ5pSb0onV3LIbar

RTN8lC4Le/5kd3lcot9J8b3EMXL5p9OGW7wBfmNVRSUI74Vmwt+v9gyp0Hd0keRCUn8lo/1V

0YD9i92KsE+/IqoYTjnya/5kX41jB8vr1ebkHFuJ404+G6ETd0owwxq64jLIcsp/GBZHGU0R

KKAo9DRLH7rpQ7PVlnw8TDNlOtWt5EJlBXFcPL+NgWbqkADAyA/XSNeWlqonvPlYfmasnAHA

pMd9NhPQhC7hJTjCiAwG8UyWpV8Dj07DHFQ5xBbkTnKH2OrJtguPqSNYtTASbsWz09S8ujoT

DXFT17NbFM2dMIiq0a4VQB3SzH13H2io9Cbg/TzJrJGmwgoXgwARAQABtDZNeVNRTCBSZWxl

YXNlIEVuZ2luZWVyaW5nIDxteXNxbC1idWlsZEBvc3Mub3JhY2xlLmNvbT6JAlQEEwEIAD4W

IQS8pDQXw7SF3RKOxtS3s7eIqNN4XAUCZTas2gIbAwUJA8JnAAULCQgHAgYVCgkICwIEFgID

AQIeAQIXgAAKCRC3s7eIqNN4XLzoD/9PlpWtfHlI8eQTHwGsGIwFA+fgipyDElapHw3MO+K9

VOEYRZCZSuBXHJe9kjGEVCGUDrfImvgTuNuqYmVUV+wyhP+w46W/cWVkqZKAW0hNp0TTvu3e

Dwap7gdk80VF24Y2Wo0bbiGkpPiPmB59oybGKaJ756JlKXIL4hTtK3/hjIPFnb64Ewe4YLZy

oJu0fQOyA8gXuBoalHhUQTbRpXI0XI3tpZiQemNbfBfJqXo6LP3/LgChAuOfHIQ8alvnhCwx

hNUSYGIRqx+BEbJw1X99Az8XvGcZ36VOQAZztkW7mEfH9NDPz7MXwoEvduc61xwlMvEsUIaS

fn6SGLFzWPClA98UMSJgF6sKb+JNoNbzKaZ8V5w13msLb/pq7hab72HH99XJbyKNliYj3+KA

3q0YLf+Hgt4Y4EhIJ8x2+g690Np7zJF4KXNFbi1BGloLGm78akY1rQlzpndKSpZq5KWw8FY/

1PEXORezg/BPD3Etp0AVKff4YdrDlOkNB7zoHRfFHAvEuuqti8aMBrbRnRSG0xunMUOEhbYS

/wOOTl0g3bF9NpAkfU1Fun57N96Us2T9gKo9AiOY5DxMe+IrBg4zaydEOovgqNi2wbU0MOBQ

b23Puhj7ZCIXcpILvcx9ygjkONr75w+XQrFDNeux4Znzay3ibXtAPqEykPMZHsZ2sbkCDQRl

NqzaARAAsdvBo8WRqZ5WVVk6lReD8b6Zx83eJUkV254YX9zn5t8KDRjYOySwS75mJIaZLsv0

YQjJk+5rt10tejyCrJIFo9CMvCmjUKtVbgmhfS5+fUDRrYCEZBBSa0Dvn68EBLiHugr+SPXF

6o1hXEUqdMCpB6oVp6X45JVQroCKIH5vsCtw2jU8S2/IjjV0V+E/zitGCiZaoZ1f6NG7ozyF

ep1CSAReZu/sssk0pCLlfCebRd9Rz3QjSrQhWYuJa+eJmiF4oahnpUGktxMD632I9aG+IMfj

tNJNtX32MbO+Se+cCtVc3cxSa/pR+89a3cb9IBA5tFF2Qoekhqo/1mmLi93Xn6uDUhl5tVxT

nB217dBT27tw+p0hjd9hXZRQbrIZUTyh3+8EMfmAjNSIeR+th86xRd9XFRr9EOqrydnALOUr

9cT7TfXWGEkFvn6ljQX7f4RvjJOTbc4jJgVFyu8K+VU6u1NnFJgDiNGsWvnYxAf7gDDbUSXE

uC2anhWvxPvpLGmsspngge4yl+3nv+UqZ9sm6LCebR/7UZ67tYz3p6xzAOVgYsYcxoIUuEZX

jHQtsYfTZZhrjUWBJ09jrMvlKUHLnS437SLbgoXVYZmcqwAWpVNOLZf+fFm4IE5aGBG5Dho2

CZ6ujngW9Zkn98T1d4N0MEwwXa2V6T1ijzcqD7GApZUAEQEAAYkCPAQYAQgAJhYhBLykNBfD

tIXdEo7G1Lezt4io03hcBQJlNqzaAhsMBQkDwmcAAAoJELezt4io03hcXqMP/01aPT3A3Sg7

oTQoHdCxj04ELkzrezNWGM+YwbSKrR2LoXR8zf2tBFzc2/Tl98V0+68f/eCvkvqCuOtq4392

Ps23j9W3r5XG+GDOwDsx0gl0E+Qkw07pwdJctA6efsmnRkjF2YVO0N9MiJA1tc8NbNXpEEHJ

Z7F8Ri5cpQrGUz/AY0eae2b7QefyP4rpUELpMZPjc8Px39Fe1DzRbT+5E19TZbrpbwlSYs1i

CzS5YGFmpCRyZcLKXo3zS6N22+82cnRBSPPipiO6WaQawcVMlQO1SX0giB+3/DryfN9VuIYd

1EWCGQa3O0MVu6o5KVHwPgl9R1P6xPZhurkDpAd0b1s4fFxin+MdxwmG7RslZA9CXRPpzo7/

fCMW8sYOH15DP+YfUckoEreBt+zezBxbIX2CGGWEV9v3UBXadRtwxYQ6sN9bqW4jm1b41vNA

17b6CVH6sVgtU3eN+5Y9an1e5jLD6kFYx+OIeqIIId/TEqwS61csY9aav4j4KLOZFCGNU0FV

ji7NQewSpepTcJwfJDOzmtiDP4vol1ApJGLRwZZZ9PB6wsOgDOoP6sr0YrDI/NNX2RyXXbgl

nQ1yJZVSH3/3eo6knG2qTthUKHCRDNKdy9Qqc1x4WWWtSRjh+zX8AvJK2q1rVLH2/3ilxe9w

cAZUlaj3id3TxquAlud4lWDz

=h5nH

-----END PGP PUBLIC KEY BLOCK-----

Чтобы импортировать ключ сборки в нашу личную общедоступную связку ключей GPG, используем **gpg –import:**

*gpg --import mysql\_pubkey.asc*

При попытке выполнить команду sudo apt-get update выводится сообщение об устаревшем значении ключа. Для устранения ошибки выводим список ключей:

*sudo apt-key list*

Экспортируем ключ в текстовый файл, используя последние 8 символов

sudo apt-key export A8D3785C | sudo gpg --dearmour -o /etc/apt/trusted.gpg.d/mysql.gpg

Установка MySQL:

*sudo apt-get update*

*sudo apt-get install mysql-server*

проверка состояния сервера MySQL:

*systemctl status mysql*

4. Управление deb-пакетами

- Установить и затем удалить deb-пакет, используя команду `dpkg`.

Используя инструкцию <https://dev.mysql.com/doc/mysql-apt-repo-quick-guide/en/>, скачиваем архив mysql-apt-config\_0.8.29-1\_all.deb.

Устанавливаем загруженный пакет:

*sudo dpkg -i ~/Загрузки/mysql-apt-config\_0.8.29-1\_all.deb*Во время установки пакета предлагается выбрать версии сервера MySQL и других компонентов.

*sudo apt-get update*

*sudo apt-get install mysql-server*

Имена пакетов, которые установили из Хранилища данных MySQL apt:

*dpkg -l | grep mysql | grep ii*

Удаляем deb-пакет:

*sudo dpkg -r mysql-apt-config mysql-common mysql-server-core-8.0 mysql-client-8.0 mysql-server mysql-client-core-8.0 mysql-server-8.0*

Выводим список команд терминала:

*cat .bash\_history*

6. Диаграмма классов

- Создать диаграмму классов с родительским классом "Животные", и двумя подклассами: "Pets" и "Pack animals".

В составы классов которых в случае Pets войдут классы: собаки, кошки, хомяки, а в класс Pack animals войдут: Лошади, верблюды и ослы).

Каждый тип животных будет характеризоваться (например, имена, даты рождения, выполняемые команды и т.д)

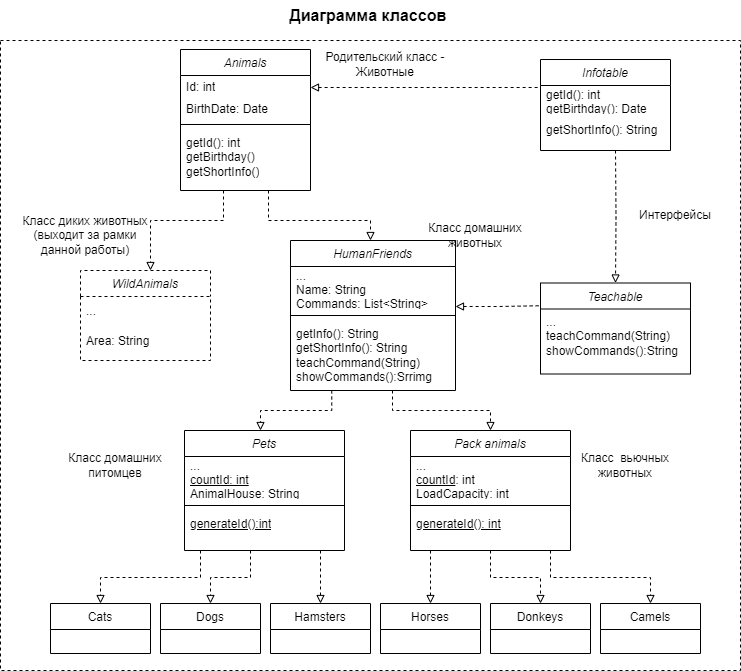
Диаграмму можно нарисовать в любом редакторе, такими как Lucidchart, Draw.io, Microsoft Visio и других.

**Выполнение:**

Строим иерархическую структуру классов, где родительским является абстрактный класс Animals, реализующий интерфейс Infotable.

Прямым наследником класса Animals является абстрактный класс HumanFriends, реализующий интерфейс Teachable, который расширяет интерфейс Infotable.

Наследниками класса HumanFriends являются абстрактные классы Pets и PackAnimals, потомками которых являются классы животных (Dog, Cat, Hamster, Camel, Horse и Donkey) согласно схемы:

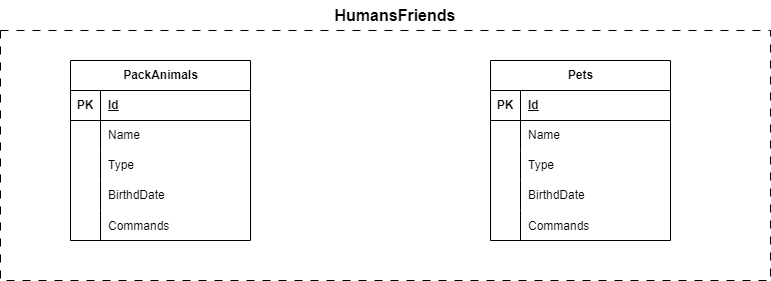


7. Работа с MySQL (Задача выполняется в случае успешного выполнения задачи “Работа с MySQL в Linux. “Установить MySQL на вашу машину”

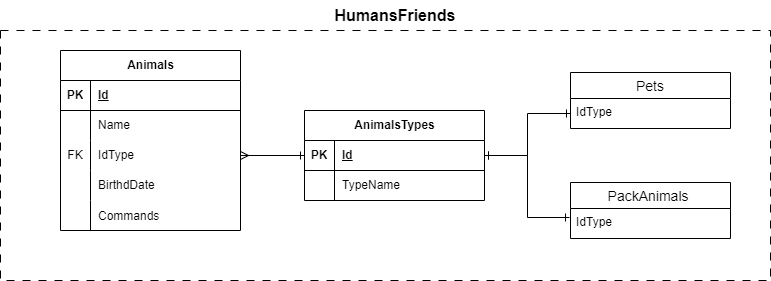
7.1. После создания диаграммы классов в 6 пункте, в 7 пункте база данных "Human Friends" должна быть структурирована в соответствии с этой диаграммой. Например, можно создать таблицы, которые будут соответствовать классам "Pets" и "Pack animals", и в этих таблицах будут поля, которые характеризуют каждый тип животных (например, имена, даты рождения, выполняемые команды и т.д.).

**Выполнение:**

Согласно описания условия задания схема БД не нормализована:



Вариант структуры нормализованной БД:



7.2 - В ранее подключенном MySQL создать базу данных с названием "Human Friends".

- Создать таблицы, соответствующие иерархии из вашей диаграммы классов.

- Заполнить таблицы данными о животных, их командах и датами рождения.

- Удалить записи о верблюдах и объединить таблицы лошадей и ослов.

- Создать новую таблицу для животных в возрасте от 1 до 3 лет и вычислить их возраст с точностью до месяца.

- Объединить все созданные таблицы в одну, сохраняя информацию о принадлежности к исходным таблицам.

Пример заполненной таблицы для теста:

Лист "Pets"

| ID | Name | Type | BirthDate | Commands |
| --- | --- | --- | --- | --- |
| 1 | Fido | Dog | 2020-01-01 | Sit, Stay, Fetch |
| 2 | Whiskers | Cat | 2019-05-15 | Sit, Pounce |
| 3 | Hammy | Hamster | 2021-03-10 | Roll, Hide |
| 4 | Buddy | Dog | 2018-12-10 | Sit, Paw, Bark |
| 5 | Smudge | Cat | 2020-02-20 | Sit, Pounce, Scratch |
| 6 | Peanut | Hamster | 2021-08-01 | Roll, Spin |
| 7 | Bella | Dog | 2019-11-11 | Sit, Stay, Roll |
| 8 | Oliver | Cat | 2020-06-30 | Meow, Scratch, Jump |

Лист "PackAnimals"

| ID | Name | Type | BirthDate | Commands |
| --- | --- | --- | --- | --- |
| 1 | Thunder | Horse | 2015-07-21 | Trot, Canter, Gallop |
| 2 | Sandy | Camel | 2016-11-03 | Walk, Carry Load |
| 3 | Eeyore | Donkey | 2017-09-18 | Walk, Carry Load, Bray |
| 4 | Storm | Horse | 2014-05-05 | Trot, Canter |
| 5 | Dune | Camel | 2018-12-12 | Walk, Sit |
| 6 | Burro | Donkey | 2019-01-23 | Walk, Bray, Kick |
| 7 | Blaze | Horse | 2016-02-29 | Trot, Jump, Gallop |
| 8 | Sahara | Camel | 2015-08-14 | Walk, Run |

**Выполнение:**

Запуск MySQL:

sudo mysql -p;

-- Создаем БД и выводим список БД:

CREATE DATABASE IF NOT EXISTS HumanFriends;

SHOW DATABASES;

USE HumanFriends;

-- DELIMITER //

--

-- DROP FUNCTION IF EXISTS getId//

-- -- Функция переводит секунды в строку: дни-часы-минуты-секунды

-- CREATE FUNCTION getId (i INT)

-- RETURNS INT DETERMINISTIC

-- BEGIN

-- SET @i := i + 1;

-- RETURN @i; #days;

-- END//

--

-- DELIMITER ;

-- Создаем таблицу Dog и заполняем ее: , InventoryValue AS QtyAvailable \* UnitPrice

DROP TABLE IF EXISTS Dogs;

CREATE TABLE Dogs (

id INT UNSIGNED PRIMARY KEY NOT NULL,

name VARCHAR(20) NOT NULL,

birthday Date,

commands VARCHAR(50)

);

-- SELECT MAX(id) FROM (SELECT id FROM Dogs UNION SELECT id FROM Cats UNION SELECT id FROM Hamsters) b;

-- Выполняем поверхностное клонирование (копирование структуры) из Dogs в Cats, Hamsters, Camel, Horses, Donkeys:

DROP TABLE IF EXISTS Cats, Hamsters, Camels, Horses, Donkeys;

CREATE TABLE Cats LIKE Dogs;

CREATE TABLE Hamsters LIKE Dogs;

CREATE TABLE Camels LIKE Dogs;

CREATE TABLE Horses LIKE Dogs;

CREATE TABLE Donkeys LIKE Dogs;

-- Заполняем таблицы данными:

-- Создаем первый счетчик:

SET @i = (SELECT MAX(id) FROM (SELECT id FROM Dogs UNION SELECT id FROM Cats UNION SELECT id FROM Hamsters) b);

SET @i := IF(@i IS NULL, 0, @i);

INSERT INTO Dogs

(id, name, birthday, commands)

VALUES

((@i := @i + 1), 'Fido', '2020-01-01','Sit, Stay, Fetch'),

((@i := @i + 1), 'Buddy', '2018-12-10', 'Sit, Paw, Bark'),

((@i := @i + 1), 'Bella', '2019-11-11', 'Sit, Stay, Roll')

;

select \* FROM Dogs;

-- DELETE FROM Dogs;

INSERT INTO Cats

(id, name, birthday, commands)

VALUES

((@i := @i + 1), 'Whiskers', '2019-05-15', 'Sit, Pounce'),

((@i := @i + 1), 'Smudge', '2020-02-20', 'Sit, Pounce, Scratch'),

((@i := @i + 1), 'Oliver', '2020-06-30', 'Meow, Scratch, Jump')

;

select \* FROM Cats;

INSERT INTO Hamsters

(id, name, birthday, commands)

VALUES

((@i := @i + 1), 'Hammy', '2021-03-10', 'Roll, Hide'),

((@i := @i + 1), 'Peanut', '2021-08-01', 'Roll, Spin')

;

SELECT \* FROM Hamsters;

-- Создаем второй счетчик:

SET @j = (SELECT MAX(id) FROM (SELECT id FROM Camels UNION SELECT id FROM Horses UNION SELECT id FROM Donkeys) b);

SET @j := IF(@j IS NULL, 0, @j);

INSERT INTO Horses

(id, name, birthday, commands)

VALUES

((@j := @j + 1), 'Thunder', '2015-07-21', 'Trot, Canter, Gallop'),

((@j := @j + 1), 'Blaze', '2016-02-29', 'Trot, Jump, Gallop'),

((@j := @j + 1), 'Storm', '2014-05-05', 'Trot, Canter')

;

SELECT \* FROM Horses;

INSERT INTO Camels

(id, name, birthday, commands)

VALUES

((@j := @j + 1), 'Sandy', '2016-11-03', 'Walk, Carry Load'),

((@j := @j + 1), 'Sahara', '2015-08-14', 'Walk, Run'),

((@j := @j + 1), 'Dune', '2018-12-12', 'Walk, Sit')

;

SELECT \* FROM Camels;

INSERT INTO Donkeys

(id, name, birthday, commands)

VALUES

((@j := @j + 1), 'Burro', '2019-01-23', 'Walk, Bray, Kick'),

((@j := @j + 1), 'Eeyore', '2017-09-18', 'Walk, Carry Load, Bray')

;

SELECT \* FROM Donkeys;

-- Создадим также виды Pets и PackAnimals и посмотрим данные в них:

DROP VIEW IF EXISTS Pets, PackAnimals;

CREATE VIEW Pets AS

SELECT id, name, 'Dog' AS type, birthday, commands FROM Dogs

UNION

SELECT id, name, 'Cat' AS type, birthday, commands FROM Cats

UNION

SELECT id, name, 'Hamster' AS type, birthday, commands FROM Hamsters

;

SELECT \* FROM Pets;

CREATE VIEW PackAnimals AS

SELECT id, name, 'Camel' AS type, birthday, commands FROM Camels

UNION

SELECT id, name, 'Horse' AS type, birthday, commands FROM Horses

UNION

SELECT id, name, 'Donkey' AS type, birthday, commands FROM Donkeys

;

SElECT \* FROM PackAnimals;

-- Удаляем данные о верблюдах:

DELETE FROM Camels;

SELECT \* FROM Camels;

-- Oбъединяем таблицы лошадей и ослов во временную таблицу HorseAndDonkey:

/\*CREATE TEMPORARY TABLE HorseAndDonkey

SELECT \* FROM PackAnimals

WHERE type = 'Horse' OR type = 'Donkey';\*/

-- т.к. мы удалили ослов, сделаем проще:

CREATE TEMPORARY TABLE HorseAndDonkey

SELECT \* FROM PackAnimals;

SELECT \* FROM HorseAndDonkey;

-- Создаем новую таблицу для животных в возрасте от 1 до 3 лет и вычисляем их возраст с точностью до месяца.

DROP TABLE IF EXISTS UnderThree;

CREATE TABLE UnderThree (

SELECT \*,

concat(DATEDIFF(curdate(), birthday) div 365, ' year '

, (DATEDIFF(curdate(), birthday) mod 365) div 30, ' month') AS Age

FROM Pets

UNION

SELECT \*,

concat(DATEDIFF(curdate(), birthday) div 365, ' year '

, (DATEDIFF(curdate(), birthday) mod 365) div 30, ' month') AS Age

FROM PackAnimals

);

SELECT \* FROM UnderThree;

-- Объединяем все таблицы животных в одну, используя виды PackAnimals и Pets в одну, сохраняя информацию о принадлежности к исходным таблицам.

CREATE TABLE Animals

SELECT \*, 'Pets' AS ParentTable

FROM Pets

UNION

SELECT \*, 'PackAnimals' AS ParentTable

FROM PackAnimals;

SELECT \* FROM Animals;