**Описание предметной области**

Небольшой зоопарк постепенно развивался, и теперь его инфраструктура довольно широка и рискует стать запутанной. Поэтому руководство зоопарка задумалось о том, чтобы обзавестись базой данных, для быстрого получения и обновления всей информации о сотрудниках и животных зоопарка, а также составлении расписаний сотрудников.

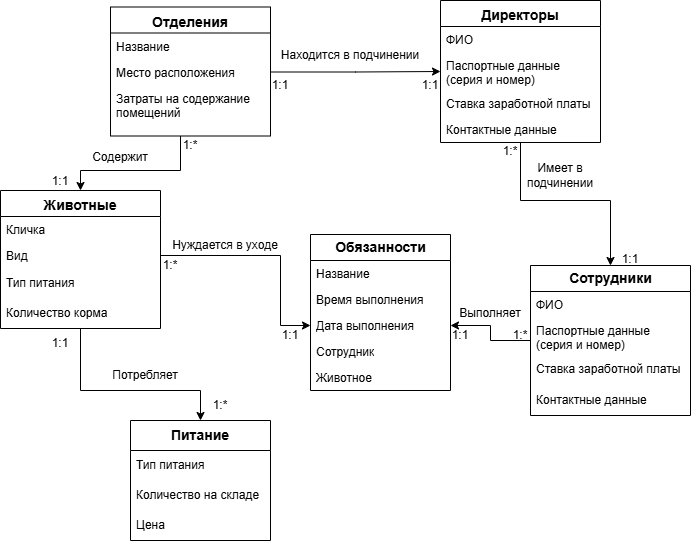
**Преимущества введения базы данных:**

* Быстрое получение объемлющей информации о животных
* Быстрое получение и обновление информации о сотрудниках
* Обновление расписания сотрудников, в случае болезни сотрудника или обновления штата
* Учет необходимых инструментов и пропитания для разных видов, удобный сбор и обновление статистик (например, расход корма и медикаментов)

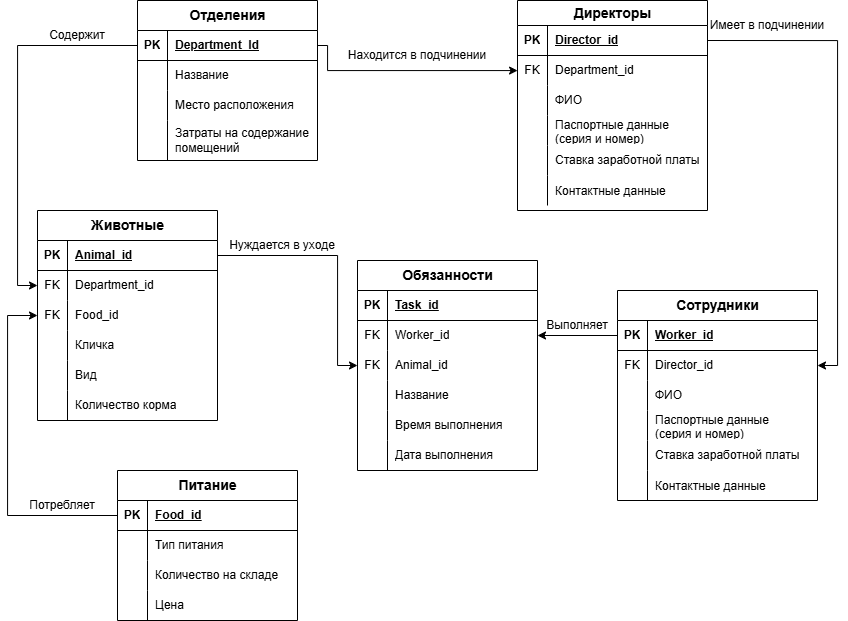
**Функции и устройство зоопарка:**

* Зоопарк состоит из нескольких отделов (например, отдел крупных млекопитающих, отдел пресмыкающихся, отдел птиц и т.п.)
* Над каждым отделом назначен свой директор, директор руководит работой отдела и занимается подбором сотрудников
* У каждого сотрудника есть различные обязанности, связанные с животными из своих отделов
* У каждого животного есть свой рацион, продукты для питания животных поставляются со склада

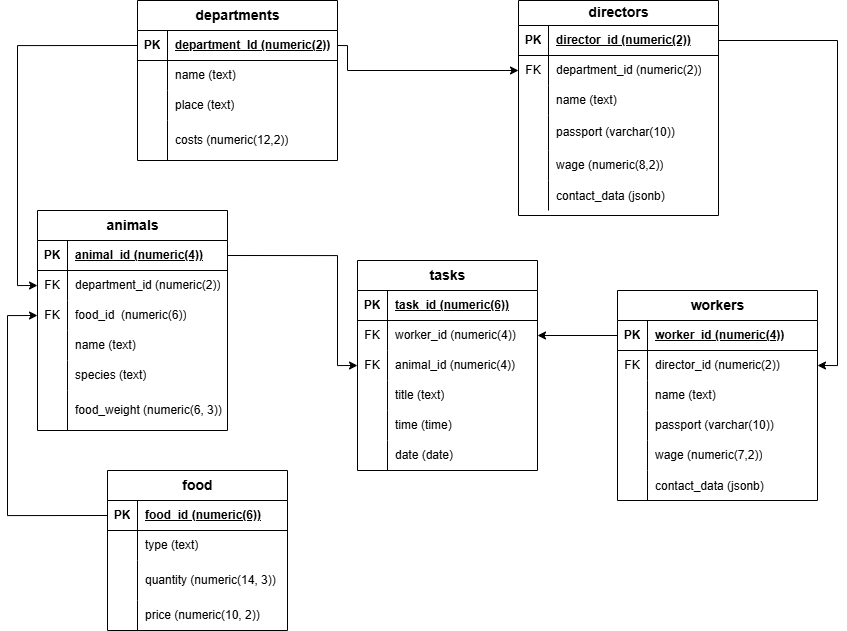
**ER-модель базы данных**



**Логическая модель базы данных**

****

**Физическая модель базы данных**



**Требования к данным**

1. **Отделение**

Отделения имеют номера (id), названия, место расположения, затраты на содержание помещений. Над каждым отделением поставлен один директор.

1. **Директор**

Директор связан с единственным отделением, имеет в качестве уникального идентификатора id. О каждом директоре также есть персональная информация (имя, паспортные данные (уникальны), контактные данные (телефон, email), ставка заработной платы). У директора в прямом подчинении есть несколько сотрудников.

1. **Сотрудники**

У каждого сотрудника также есть уникальный идентификатор (id), персональная информация (имя, паспортные данные (уникальны), ставка заработной платы, контактные данные (телефон, email), а также круг обязанностей.

1. **Обязанности**

Обязанности также имеют номер (id), название (из списка), привязаны к определенному животному из обслуживаемого сотрудником отдела, а также время и дату выполнения.

1. **Животные**

Животные имеют уникальный идентификатор (id), кличку и породу (сочетание кличка и порода уникальны), рацион (тип корма и количество), каждое животное расположено в единственном отделе, может обслуживаться несколькими сотрудниками.

1. **Питание**

На складе расположено несколько видов корма, для каждого имеются такие характеристики как тип корма (это id), его количество на складе, цена за кг.

**Требования к транзакциям**

1. Обновление/удаление данных (об отделении, директоре, сотруднике, животном, поступлении корма)
2. Ввод данных (о новом животном, сотруднике, отделении (одновременно с сотрудником), новом типе корма)
3. Получение информации из базы данных (получить информацию о животных из конкретного отделения, узнать обязанности сотрудника, узнать о работниках некоторого отделения, узнать о наличии необходимого количества корма на данный момент)

**Физическое проектирование**

DROP TABLE IF EXISTS departments CASCADE;

DROP TABLE IF EXISTS directors CASCADE;

DROP TABLE IF EXISTS workers CASCADE;

DROP TABLE IF EXISTS food CASCADE;

DROP TABLE IF EXISTS animals CASCADE;

DROP TABLE IF EXISTS tasks CASCADE;

DROP TYPE IF EXISTS data;

CREATE TYPE data AS (telephone text, email text);

CREATE TABLE departments (

department\_id numeric(2) NOT NULL,

name text NOT NULL,

place text NOT NULL,

costs numeric(12,2) NOT NULL,

PRIMARY KEY (department\_id),

UNIQUE (name),

CHECK (costs > 0) );

INSERT INTO departments (department\_id, name, place, costs)

VALUES (1, 'Пресмыкающиеся', 'Северная часть', 452638.49),

(2, 'Птицы', 'Северо-западная часть', 683636.37),

(3, 'Мелкие млекопитающие', 'Восточная часть', 316194.80),

(4, 'Крупные травоядные', 'Южная часть', 361842.37),

(5, 'Кошачьи', 'Юго-восточная часть', 984378.65),

(6, 'Приматы', 'Центральная часть', 461939.76),

(7, 'Насекомые', 'Северо-западная часть', 136327.74),

(8, 'Ночные животные', 'Северо-восточная часть', 686336.37);

CREATE TABLE directors (

director\_id numeric(2) NOT NULL,

department\_id numeric(2) NOT NULL,

name text NOT NULL,

passport varchar(10) NOT NULL,

wage numeric(8,2) NOT NULL,

contact\_data jsonb,

PRIMARY KEY (director\_id),

UNIQUE (passport),

CHECK (wage > 0),

FOREIGN KEY (department\_id)

REFERENCES departments (department\_id)

ON DELETE CASCADE );

INSERT INTO directors (director\_id, department\_id, name, passport, wage, contact\_data)

VALUES (1, 1, 'Иванов Петр Сергеевич', 4519367427, 80182.00, '{ "telephone": 89563784658, "email": "ivanov\_1@mail.ru"}'::jsonb ),

(2, 2, 'Иванов Валерий Сергеевич', 4519364820, 79689.00, '{ "telephone": 89759204858, "email": "ivanov\_2@mail.ru"}'::jsonb ),

(3, 3, 'Иванова Анна Сергеевна', 4565489247, 85378.00, '{ "telephone": 89853297583, "email": "ivanova\_3@mail.ru"}'::jsonb ),

(4, 4, 'Иванов Сергей Сергеевич', 4530490389, 90320.00, '{ "telephone": 89424782375, "email": "ivanov\_4@mail.ru"}'::jsonb ),

(5, 5, 'Иванов Сергей Петрович', 4520482048, 99389.00, '{ "telephone": 89609303558, "email": "ivanov\_5@mail.ru"}'::jsonb ),

(6, 6, 'Иванова Елена Сергеевна', 3942340590, 953820.00, '{ "telephone": 89305820458, "email": "ivanov\_6@mail.ru"}'::jsonb ),

(7, 7, 'Иванов Иван Сергеевич', 4892489253, 78520.00, '{ "telephone": 85783294829, "email": "ivanov\_7@mail.ru"}'::jsonb ),

(8, 8, 'Иванова Марина Сергеевна', 4538941034, 83490.00, '{ "telephone": 89203941937, "email": "ivanov\_8@mail.ru"}'::jsonb );

CREATE TABLE workers (

worker\_id numeric(4) NOT NULL,

director\_id numeric(2) NOT NULL,

name text NOT NULL,

passport varchar(10) NOT NULL,

wage numeric(7,2) NOT NULL,

contact\_data jsonb,

PRIMARY KEY (worker\_id),

UNIQUE (passport),

CHECK (wage > 0),

FOREIGN KEY (director\_id)

REFERENCES directors (director\_id)

ON DELETE CASCADE );

INSERT INTO workers (worker\_id, director\_id, name, passport, wage, contact\_data)

VALUES (1, 1, 'Сидоренко Галина Ивановна', 4528497247, 45402.48, '{ "telephone": 89346284636, "email": "sidorenko@mail.ru"}'::jsonb),

(2, 1, 'Федоров Иван Петрович', 4639127837, 46203.40, '{ "telephone": 89381193728, "email": "fedorov@mail.ru"}'::jsonb),

(3, 1, 'Петров Иван Петрович', 4478247319, 44580.43, '{ "telephone": 89468294738, "email": "petrov@mail.ru"}'::jsonb),

(4, 1, 'Никифорова Анна Ильинична', 4639128058, 45402.48, '{ "telephone": 89479247374, "email": "nikiforova@mail.ru"}'::jsonb),

(5, 2, 'Никифоров Антон Васильевич', 4452837492, 39173.00, '{ "telephone": 89583487283, "email": "nikiforov@mail.ru"}'::jsonb),

(6, 2, 'Сергеева Алина Михайловна', 4361368483, 35481.73, '{ "telephone": 89585620344, "email": "sergeeva@mail.ru"}'::jsonb),

(7, 2, 'Тарасова Елена Васильевна', 194279447, 36379.82, '{ "telephone": 89792747283, "email": "tarasova@mail.ru"}'::jsonb),

(8, 3, 'Баранов Кирилл Иванович', 4478294623, 46823.48, '{ "telephone": 89368419373, "email": "baranov@mail.ru"}'::jsonb),

(9, 3, 'Хайруллин Евгений Петрович', 4394713899, 47293.84, '{ "telephone": 89322253645, "email": "khairullin@mail.ru"}'::jsonb),

(10, 3, 'Риманов Федор Степанович', 1937947274, 49379.73, '{ "telephone": 89294729473, "email": "rimanov@mail.ru"}'::jsonb),

(11, 3, 'Степаненко Надежда Антоновна', 1947294463, 42938.03, '{ "telephone": 89927438923, "email": "ctepanenko@mail.ru"}'::jsonb),

(12, 4, 'Щербакова Карина Евгеньевна', 1937493478, 50138.00, '{ "telephone": 89892738925, "email": "sherbakova@mail.ru"}'::jsonb),

(13, 4, 'Мельников Степан Михайлович', 1963825571, 50389.38, '{ "telephone": 89672847827, "email": "melnikov@mail.ru"}'::jsonb),

(14, 4, 'Корнев Максим Михайлович', 1937483740, 22440.96, '{ "telephone": 89331802830, "email": "kornev@mail.ru"}'::jsonb),

(15, 4, 'Сергеева Галина Петровна', 1963882631, 58310.00, '{ "telephone": 89592374509, "email": "sergeeva\_2@mail.ru"}'::jsonb),

(16, 4, 'Карпов Дмитрий Анатольевич', 1956826464, 53947.85, '{ "telephone": 89480241804, "email": "karpov@mail.ru"}'::jsonb),

(17, 5, 'Хохлова Валерия Кирилловна', 4368394724, 52839.83, '{ "telephone": 89407274913, "email": "hohlov@mail.ru"}'::jsonb),

(18, 5, 'Блинов Петр Михайлович', 1939361836, 51738.00, '{ "telephone": 89391638183, "email": "petrov@mail.ru"}'::jsonb),

(19, 5, 'Аксенова Марина Кирилловна', 1936816386, 60371.78, '{ "telephone": 89510389191, "email": "aksenova@mail.ru"}'::jsonb),

(20, 5, 'Гаврилов Родион Юрьевич', 1962585490, 58268.98, '{ "telephone": 89820172865, "email": "gavrilov@mail.ru"}'::jsonb),

(21, 6, 'Жуков Антон Петрович', 1930580299, 69220.33, '{ "telephone": 89923892890, "email": "jukov@mail.ru"}'::jsonb),

(22, 6, 'Фролов Виталий Данилович', 1843028044, 63204.30, '{ "telephone": 89482040399, "email": "frolov@mail.ru"}'::jsonb),

(23, 6, 'Котов Петр Петрович', 1940358300, 65481.00, '{ "telephone": 85829048010, "email": "kotov@mail.ru"}'::jsonb),

(24, 6, 'Журавлева Елизавета Егоровна', 1940284930, 69205.32, '{ "telephone": 8930284820, "email": "juravleva@mail.ru"}'::jsonb),

(25, 6, 'Шешуков Леонид Сергеевич', 4364801840, 66309.73, '{ "telephone": 89515620344, "email": "sheshukovmail.ru"}'::jsonb),

(26, 6, 'Быкова Василиса Родионовна', 1903480848, 63564.22, '{ "telephone": 89492482984, "email": "bykov@mail.ru"}'::jsonb),

(27, 8, 'Кошелев Давид Иосифович', 4340148183, 69020.54, '{ "telephone": 89910348299, "email": "koshelev@mail.ru"}'::jsonb),

(28, 8, 'Логинов Елисей Петрович', 4301445792, 63048.73, '{ "telephone": 89585620344, "email": "sergeev@mail.ru"}'::jsonb);

CREATE TABLE food (

food\_id numeric(6) NOT NULL,

type text NOT NULL,

quantity numeric(14, 3) NOT NULL,

price numeric(10,2) NOT NULL,

PRIMARY KEY (food\_id),

CHECK (quantity >= 0),

CHECK (price > 0) );

INSERT INTO food (food\_id, type, quantity, price)

VALUES (1, 'комбикорм', 903339, 30.00),

(2, 'мясо (говядина)', 399008, 800.04),

(3, 'мясо (курица)', 420573, 320.67),

(4, 'фрукты', 48204, 200.45),

(5, 'рыба (сардины)', 310480, 254.63),

(6, 'кукуруза', 859302, 25.07),

(7, 'бамбук', 39401, 4002.49),

(8, 'сверчки', 80310, 2791.03);

CREATE TABLE animals (

animal\_id numeric(4) NOT NULL,

department\_id numeric(2) NOT NULL,

food\_id numeric(6) NOT NULL,

name text NOT NULL,

species text NOT NULL,

food\_weight numeric(6,3) NOT NULL,

PRIMARY KEY (animal\_id),

UNIQUE (name, species),

CHECK (food\_weight > 0),

FOREIGN KEY (department\_id)

REFERENCES departments (department\_id)

ON DELETE CASCADE,

FOREIGN KEY (food\_id)

REFERENCES food (food\_id)

ON DELETE CASCADE);

INSERT INTO animals (animal\_id, department\_id, food\_id, name, species, food\_weight)

VALUES (1, 1, 3, 'Беляш', 'Нильский крокодил', 1.204),

(2, 1, 2, 'Каа', 'Удав', 1.050),

(3, 1, 8, 'Октавий', 'Бородатая агама', 0.230),

(4, 1, 2, 'Монти', 'Питон', 0.950),

(5, 2, 5, 'Порро', 'Пингвин', 2.570),

(6, 2, 4, 'Император', 'Казуар', 4.250),

(7, 2, 1, 'Вася', 'Большой белохвостый какаду', 0.100),

(8, 2, 1, 'Жасмин', 'Гиацинтовый ара', 0.070),

(9, 2, 1, 'Красавчик', 'Павлин', 0.350),

(10, 3, 1, 'Сцинтия', 'Мешотчатая крыса', 0.010),

(11, 3, 4, 'Соня', 'Новозеландский красный кролик', 0.120),

(12, 3, 4, 'Роза', 'Капибара', 4.500),

(13, 3, 4, 'Гибби', 'Агути', 0.250),

(14, 3, 1, 'Лариса', 'Тростниковая крыса', 0.050),

(15, 4, 4, 'Иннокентий', 'Африканский слон', 300.050),

(16, 4, 1, 'Самсон', 'Жираф', 30.500),

(17, 4, 1, 'Маруся', 'Як', 6.000),

(18, 4, 1, 'Финн', 'Зебра', 8.750),

(19, 4, 1, 'Кристо', 'Винторогий козел', 6.000),

(20, 5, 2, 'Агат', 'Амурский тигр', 9.600),

(21, 5, 2, 'Кира', 'Лев', 7.000),

(22, 5, 3, 'Алла', 'Пума', 3.260),

(23, 5, 3, 'Коралл', 'Обыкновенная рысь', 2.000),

(24, 6, 4, 'Банши', 'Карликовый шимпанзе', 2.150),

(25, 6, 4, 'Минна', 'Карликовый шимпанзе', 1.850),

(26, 6, 4, 'Айзек', 'Карликовый шимпанзе', 0.850),

(27, 6, 4, 'Антонина', 'Горилла', 32.850),

(28, 6, 4, 'Грета', 'Белорукий гиббон', 3.450),

(29, 8, 4, 'Сережа', 'Азарская мирикина', 0.250),

(30, 8, 3, 'Макар', 'Обыкновенный еж', 0.030);

CREATE TABLE tasks (

task\_id numeric(6) NOT NULL,

worker\_id numeric(4) NOT NULL,

animal\_id numeric(4) NOT NULL,

title text NOT NULL,

time time NOT NULL,

date date NOT NULL,

PRIMARY KEY (task\_id),

UNIQUE (worker\_id, time, date),

UNIQUE (animal\_id, time, date),

CHECK (title IN ('Уборка', 'Кормление', 'Медосмотр', 'Взвешивание')),

FOREIGN KEY (worker\_id)

REFERENCES workers (worker\_id)

ON DELETE CASCADE,

FOREIGN KEY (animal\_id)

REFERENCES animals (animal\_id)

ON DELETE CASCADE );

INSERT INTO tasks (task\_id, worker\_id, animal\_id, title, time, date)

VALUES (1, 1, 1, 'Кормление', '16:10', '2020-03-20'),

(2, 6, 5, 'Медосмотр', '12:00', '2020-03-10'),

(3, 14, 15, 'Взвешивание', '15:45', '2020-03-05'),

(4, 16, 15, 'Уборка', '9:00', '2020-03-17'),

(5, 10, 12, 'Кормление', '19:00', '2020-03-22'),

(6, 26, 30, 'Кормление', '3:00', '2020-03-12'),

(7, 4, 2, 'Кормление', '19:00', '2020-03-20'),

(8, 6, 7, 'Уборка', '14:00', '2020-03-13'),

(9, 17, 21, 'Кормление', '14:00', '2020-03-12'),

(10, 11, 16, 'Кормление', '16:30', '2020-03-16'),

(11, 11, 17, 'Кормление', '18:00', '2020-03-10'),

(12, 12, 15, 'Кормление', '14:50', '2020-03-15'),

(13, 6, 7, 'Уборка', '19:40', '2020-03-11'),

(14, 19, 26, 'Медосмотр', '12:25', '2020-03-18'),

(15, 15, 16, 'Кормление', '14:20', '2020-03-23');

**Типичные запросы к базе данных**

**Найти всех животных с именем начинающимся на А**

SELECT name, species

FROM animals

WHERE name LIKE 'А%';

**Посчитать количество животных в отделе «Приматы»**

SELECT d.name, count(\*)

FROM departments AS d

INNER JOIN animals AS a ON d.department\_id = a.department\_id

GROUP BY d.name

HAVING d.name = 'Приматы';

**Выяснить все рабочие задачи работников отдела «Крупные травоядные»**

WITH herbivore\_workers AS (

SELECT worker\_id, w.name

FROM departments AS dep

INNER JOIN directors AS dir ON dep.department\_id = dir.department\_id

LEFT JOIN workers AS w ON dir.director\_id = w.director\_id

WHERE dep.name = 'Крупные травоядные' )

SELECT hw.name, t.title, a.name, a.species

FROM herbivore\_workers AS hw

INNER JOIN tasks AS t ON hw.worker\_id = t.worker\_id

INNER JOIN animals AS a ON t.animal\_id = a.animal\_id;

**Выяснить количество работников в каждом отделе**

CREATE OR REPLACE VIEW number\_workers AS (

SELECT dep.name AS name, wd.number AS number

FROM departments AS dep INNER JOIN

(SELECT d.department\_id, count(\*) AS number

FROM directors AS d

INNER JOIN workers AS w ON d.director\_id = w.director\_id

GROUP BY d.director\_id ) AS wd

ON dep.department\_id = wd.department\_id );

SELECT \* FROM number\_workers;

**Выяснить в каком отделе животных больше 'Мелкие млекопитающие' или 'Крупные травоядные'**

SELECT

CASE WHEN (SELECT number FROM number\_workers AS nw WHERE nw.name = 'Мелкие млекопитающие') > (SELECT number FROM number\_workers AS nw WHERE nw.name = 'Крупные травоядные') THEN 'Мелкие млекопитающие'

ELSE 'Крупные травоядные'

END as department;

**Посчитать количество издержек в каждом отделе**

SELECT dep.name, dep.costs AS fixed\_costs,

(CASE WHEN ww.workers\_wage IS NULL AND fc.food\_costs IS NULL THEN 0.00

WHEN ww.workers\_wage IS NULL THEN fc.food\_costs

WHEN fc.food\_costs IS NULL THEN ww.workers\_wage

ELSE ww.workers\_wage + fc.food\_costs

END) AS variable\_costs,

(CASE WHEN ww.workers\_wage IS NULL AND fc.food\_costs IS NULL THEN dep.costs

WHEN ww.workers\_wage IS NULL THEN dep.costs + fc.food\_costs

WHEN fc.food\_costs IS NULL THEN dep.costs + ww.workers\_wage

ELSE dep.costs + ww.workers\_wage + fc.food\_costs

END) AS total\_costs

FROM

(SELECT d.department\_id, sum(w.wage) AS workers\_wage

FROM directors AS d

LEFT JOIN workers AS w ON d.director\_id = w.director\_id

GROUP BY d.department\_id) AS ww

RIGHT JOIN departments AS dep

ON dep.department\_id = ww.department\_id

LEFT JOIN

(SELECT a.department\_id, sum(f.price \* a.food\_weight) AS food\_costs

FROM animals AS a

INNER JOIN food AS f ON a.food\_id = f.food\_id

GROUP BY a.department\_id) AS fc

ON dep.department\_id = fc.department\_id;

**Функции и триггеры**

**Выяснить контактные данные рабочего по его имени**

CREATE OR REPLACE FUNCTION worker\_contact\_data(current\_name text) RETURNS TABLE (telephone text, email text)

AS $$

SELECT \* FROM jsonb\_populate\_record(null::data, (

SELECT contact\_data FROM workers WHERE name = current\_name) );

$$ LANGUAGE SQL;

SELECT \* FROM worker\_contact\_data('Жуков Антон Петрович');

**При кормлении животного изменить количество корма на складе**

CREATE OR REPLACE FUNCTION feed()

RETURNS trigger AS

$$

DECLARE

delta numeric(10):= 0;

number\_of\_food numeric(6):= 0;

BEGIN

IF NEW.title != 'Кормление' THEN

RETURN NEW;

END IF;

delta = (

SELECT a.food\_weight

FROM animals AS a

WHERE a.animal\_id = NEW.animal\_id );

number\_of\_food = (

SELECT f.food\_id

FROM animals AS a INNER JOIN

food AS f ON a.food\_id = f.food\_id

WHERE a.animal\_id = NEW.animal\_id );

IF (SELECT quantity FROM food WHERE food\_id = number\_of\_food) >= delta THEN

UPDATE food

SET quantity = quantity - delta

WHERE food\_id = number\_of\_food;

RETURN NEW;

END IF;

RAISE NOTICE 'Еды не хватает!';

RETURN NULL;

END;

$$ LANGUAGE plpgsql;

DROP TRIGGER IF EXISTS feed\_trigger ON tasks;

CREATE TRIGGER feed\_trigger BEFORE INSERT ON tasks

FOR EACH ROW EXECUTE PROCEDURE feed();

INSERT INTO tasks

VALUES (16, 15, 15, 'Кормление', '16:45', '2020-03-21');