МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

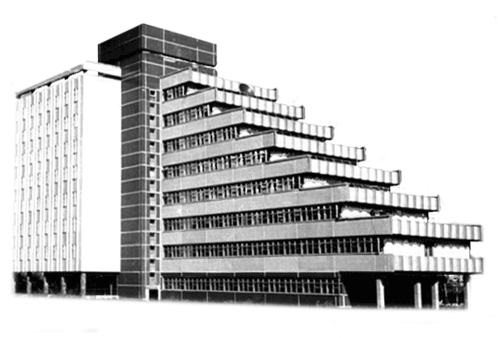
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Учебный Центр Информационных Технологий «Информатика»



Лабораторная работа

по дисциплине «Системы управления базами данных I часть»

Направление подготовки: 230105 - «Программное обеспечение вычислительной техники и автоматизированных систем»

Выполнил слушатель: Аксенов И. Г.

Вариант: 29

Дата сдачи: 14.06.2021

Преподаватель: Медведев М. А.

Новосибирск, 2021 г.

# Задание

Вариант 29. ОТДЕЛ КАДРОВ

База данных должна содержать сведения о следующих объектах:

Сотрудники - фамилия, имя, отчество, адрес, дата рождения, номер паспорта, дата выдачи, учреждение, выдавшее паспорт, должность, отдел, оклад, сведения о перемещении (отдел, должность, причина перевода, номер и дата приказа), семейное положение, состав семьи (иждивенцы), отношение к военной службе.

Командировки – сотрудник, город, цель командировки, срок командировки, дата начала, аванс, авансовый отчет (назначение платежа, получатель платежа, сумма).

Отделы - название, штатное расписание (наименование должности, разряд по ЕТС, количество ставок, количество занятых ставок).

Выходные документы:

1. Распределение среднего оклада по отделам.

2. Распределение суммы командировочных по отделам.

Бизнес-правила

1. Каждый сотрудник работает только в одной должности и только в одном отделе. Оклад сотрудника определяется должностью, но некоторые сотрудники имеют персональную надбавку.

2. В каждом отделе более одного сотрудника.

3. Некоторые сотрудники неоднократно ездят в командировки в различные города России.

4. Сведения о командировках сохраняются в течение года.

5. Сведения об уволенных сотрудниках сохраняются в течение 5 лет.

# Инфологическое проектирование.

***Анализ предметной области:***

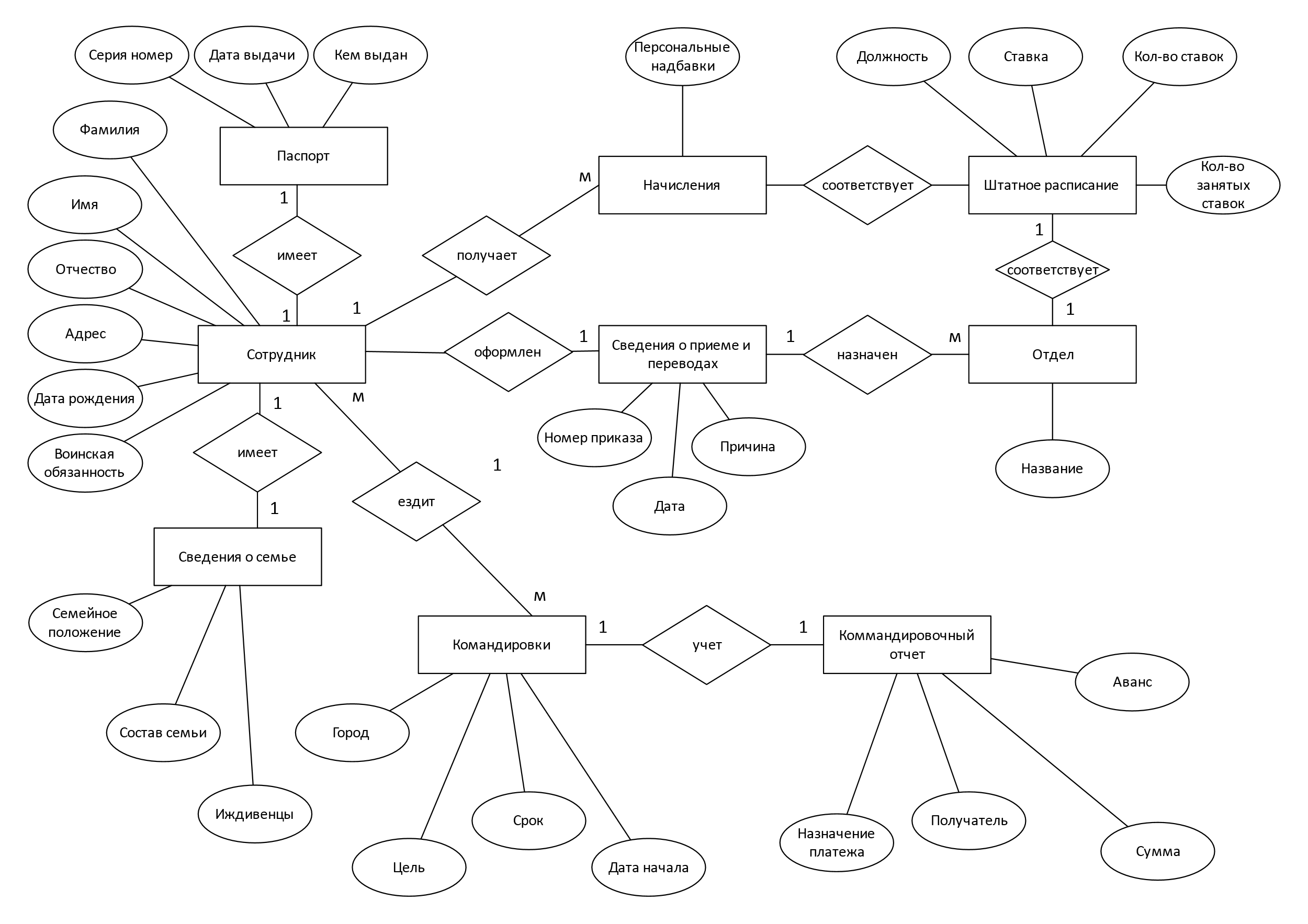
База данных создаётся для информационного обслуживания отдела кадров предприятия.

Для инфологического проектирования – самого абстрактного уровня проектирования баз данных, воспользуемся методом «сущность-связь». Будем так-же учитывать что составленная модель предметной области будет изменяться и уточняться в процессе проектирования.

Исходя из сути предметной области необходимо выделить сущности – объекты базы данных, о которых необходимо хранить информацию, а так-же определить набор хранимых в базе данных атрибутов сущности. В нашей предметной области такими сущностями могут быть:

|  |  |
| --- | --- |
| **Наименование сущности** | **Назначение сущности** |
| Сотрудник | Данные о сотрудниках |
| Паспорт | Данные о паспортах сотрудников |
| Сведения о семье | Данные о семье сотрудников |
| Командировки | Данные о командировках |
| Командировочный отчет | Отчеты о командировках и расходах |
| Сведения о приемах и переводах | Данные о кадровых перемещениях |
| Отдел | Данные о отделах предприятия |
| Штатное расписание | Данные о должностях и ставках |
| Начисления | Данные о начислениях сотрудникам |

На следующем рисунке ER-диаграммы в нотации Чена представлены предполагаемые сущности, их атрибуты и связи между ними:



*Рис.1 ER-диаграмма*

Определение связи между сущностями:

|  |  |  |  |
| --- | --- | --- | --- |
| **Тип сущности** | **Тип связи** | **Кратность** | **Тип сущности** |
| Сотрудник | имеет | 1 : 1 | Паспорт |
| Сотрудник | имеет | 1 : 1 | Сведения о семье |
| Сотрудник | ездит в | 1 : М | Командировки |
| Сотрудник | оформлен | 1 : М | Сведения о переводах |
| Сотрудник | получает | 1 : М | Начисления |
| Командировки | учитываются | 1 : М | Отчет о командировках |
| Сведения о переводах | назначен | 1 : М | Отдел |
| Отдел | соответствует | 1 : М | Штатное расписание |
| Штатное расписание | соответствует | М : М | Начисление |

**3. Логическое проектирование**

Цель логического проектирования – преобразование концептуальной модели в логическую, не зависящую от конкретной СУБД, модель. Для этапа логического проектирования существуют следующие стадии:

– выбор модели данных – для выполнения лабораторной работы выбрана реляционная модель данных

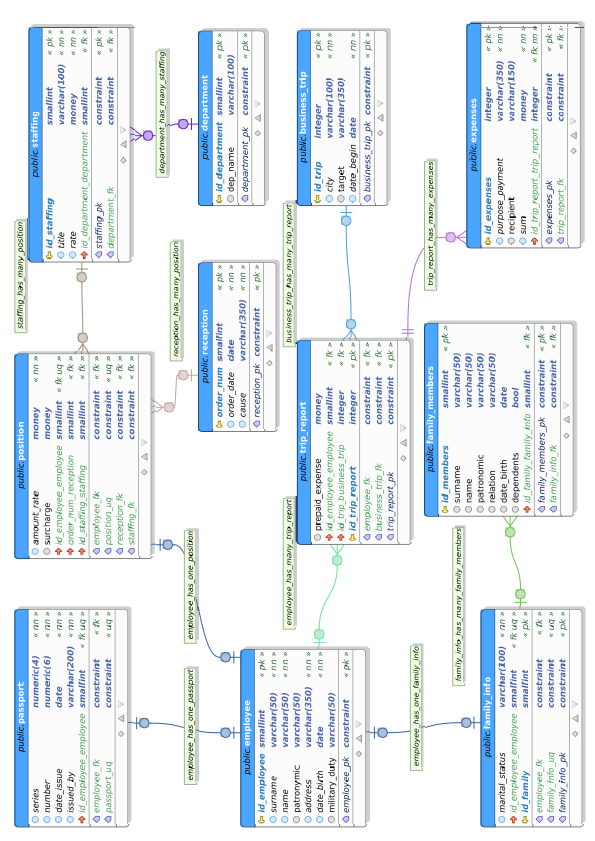
– определение таблиц из ER-диаграммы – таблица создается на основе сущностей, определенных в процессе инфологического проектирования, определяются связи между таблицами и ключи, по которым эти таблицы связываются

– нормализация таблиц – проверяется корректность структуры таблиц, исключается избыточность и коллизии обновленя, удаления, вставки данных

– проверка логической модели на возможность осуществления всех тразакций необходимых пользователям БД.

На этапе логического проектирования учитывается специфика конкретной модели данных.

В результате этапа логического проектирования были изменены некоторые связи между таблицами и введены в схему новые таблицы, позволяющие нормализовать таблицы БД. Выделены ключевые отрибуты, по которым осуществляется связь между таблицами. Так-же предварительно определены типы данных и ограничения для атрибутов таблиц. В результате логического проектирования схема приняла следующий вид:

*Рис.2 Логическая схема БД*

**3. Физическое проектирование**

1.1 Проектировние средствами выбранной СУБД.

Для выполнения данной лабораторной работы была выбрана СУБД PostgreSQL – свободная объектно-реляционная система управления базами данных.

Создание таблиц, определение ключей:

CREATE TABLE public.employee (

id\_employee smallint NOT NULL GENERATED ALWAYS AS IDENTITY ,

surname varchar(50) NOT NULL,

name varchar(50) NOT NULL,

patronymic varchar(50),

address varchar(350) NOT NULL,

date\_birth date NOT NULL,

military\_duty varchar(50),

CONSTRAINT employee\_pk PRIMARY KEY (id\_employee));

CREATE TABLE public.passport (

series numeric(4) NOT NULL,

number numeric(6) NOT NULL,

date\_issue date NOT NULL,

issued\_by varchar(200) NOT NULL,

id\_employee\_employee smallint);

ALTER TABLE public.passport ADD CONSTRAINT employee\_fk FOREIGN KEY (id\_employee\_employee)

REFERENCES public.employee (id\_employee) MATCH FULL

ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE public.passport ADD CONSTRAINT passport\_uq UNIQUE (id\_employee\_employee);

CREATE TABLE public.family\_info (

marital\_status varchar(100) NOT NULL,

id\_employee\_employee smallint,

id\_family smallint NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT BY 1 MINVALUE 0 MAXVALUE 32767 START WITH 1 CACHE 1 ),

CONSTRAINT family\_info\_pk PRIMARY KEY (id\_family));

ALTER TABLE public.family\_info ADD CONSTRAINT employee\_fk FOREIGN KEY (id\_employee\_employee)

REFERENCES public.employee (id\_employee) MATCH FULL

ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE public.family\_info ADD CONSTRAINT family\_info\_uq UNIQUE (id\_employee\_employee);

CREATE TABLE public.business\_trip (

id\_trip integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT BY 1 MINVALUE 0 MAXVALUE 2147483647 START WITH 1 CACHE 1 ),

city varchar(100) NOT NULL,

target varchar(350),

date\_begin date NOT NULL,

CONSTRAINT business\_trip\_pk PRIMARY KEY (id\_trip));

CREATE TABLE public.trip\_report (

prepaid\_expense money,

id\_employee\_employee smallint,

id\_trip\_business\_trip integer,

id\_trip\_report integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT BY 1 MINVALUE 0 MAXVALUE 2147483647 START WITH 1 CACHE 1 ),

CONSTRAINT trip\_report\_pk PRIMARY KEY (id\_trip\_report));

ALTER TABLE public.trip\_report ADD CONSTRAINT employee\_fk FOREIGN KEY (id\_employee\_employee)

REFERENCES public.employee (id\_employee) MATCH FULL

ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE public.trip\_report ADD CONSTRAINT business\_trip\_fk FOREIGN KEY (id\_trip\_business\_trip)

REFERENCES public.business\_trip (id\_trip) MATCH FULL

ON DELETE SET NULL ON UPDATE CASCADE;

CREATE TABLE public.department (

id\_department smallint NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT BY 1 MINVALUE 1 MAXVALUE 32767 START WITH 1 CACHE 1 ),

dep\_name varchar(100),

CONSTRAINT department\_pk PRIMARY KEY (id\_department));

CREATE TABLE public.staffing (

id\_staffing smallint NOT NULL GENERATED ALWAYS AS IDENTITY ,

title varchar(100) NOT NULL,

rate money NOT NULL,

salary money NOT NULL,

id\_department\_department smallint,

CONSTRAINT staffing\_pk PRIMARY KEY (id\_staffing));

ALTER TABLE public.staffing ADD CONSTRAINT department\_fk FOREIGN KEY (id\_department\_department)

REFERENCES public.department (id\_department) MATCH FULL

ON DELETE SET NULL ON UPDATE CASCADE;

CREATE TABLE public.reception (

order\_num smallint NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT BY 1 MINVALUE 0 MAXVALUE 32767 START WITH 1 CACHE 1 ),

order\_date date NOT NULL,

cause varchar(350) NOT NULL,

CONSTRAINT reception\_pk PRIMARY KEY (order\_num));;

CREATE TABLE public."position" (

amount\_rate money NOT NULL,

surcharge money,

id\_employee\_employee smallint,

order\_num\_reception smallint,

id\_staffing\_staffing smallint);

ALTER TABLE public."position" ADD CONSTRAINT employee\_fk FOREIGN KEY (id\_employee\_employee)

REFERENCES public.employee (id\_employee) MATCH FULL

ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE public."position" ADD CONSTRAINT position\_uq UNIQUE (id\_employee\_employee);

ALTER TABLE public."position" ADD CONSTRAINT reception\_fk FOREIGN KEY (order\_num\_reception)

REFERENCES public.reception (order\_num) MATCH FULL

ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE public."position" ADD CONSTRAINT staffing\_fk FOREIGN KEY (id\_staffing\_staffing)

REFERENCES public.staffing (id\_staffing) MATCH FULL

ON DELETE SET NULL ON UPDATE CASCADE;

CREATE TABLE public.expenses (

id\_expenses integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT BY 1 MINVALUE 0 MAXVALUE 2147483647 START WITH 1 CACHE 1 ),

purpose\_payment varchar(350) NOT NULL,

recipient varchar(150),

sum money NOT NULL,

id\_trip\_report\_trip\_report integer NOT NULL,

CONSTRAINT expenses\_pk PRIMARY KEY (id\_expenses));

ALTER TABLE public.expenses ADD CONSTRAINT trip\_report\_fk FOREIGN KEY (id\_trip\_report\_trip\_report)

REFERENCES public.trip\_report (id\_trip\_report) MATCH FULL

ON DELETE RESTRICT ON UPDATE CASCADE;

CREATE TABLE public.family\_members (

id\_members smallint NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT BY 1 MINVALUE 0 MAXVALUE 32767 START WITH 1 CACHE 1 ),

surname varchar(50),

name varchar(50),

patronomic varchar(50),

relation varchar(50),

date\_birth date,

dependents bool,

id\_family\_family\_info smallint,

CONSTRAINT family\_members\_pk PRIMARY KEY (id\_members));

ALTER TABLE public.family\_members ADD CONSTRAINT family\_info\_fk FOREIGN KEY (id\_family\_family\_info)

REFERENCES public.family\_info (id\_family) MATCH FULL

ON DELETE SET NULL ON UPDATE CASCADE;

Заполняем БД тестовыми данными:

INSERT INTO employee (surname, name, patronymic, address, date\_birth, military\_duty)

VALUES ('Иванов', 'Иван', 'Иванович', 'Новосибирск, ул. Красный проспект, 74 -99', '1981.12.25', 'военнообязанный'),

('Сидорова', 'Агафья', 'Павловна', 'Новосибирск, ул. Красноармейская, 7 -35', '1978.01.23', 'в запасе'),

('Трамп', 'Игорь', 'Васильевич', 'Новосибирск, ул. Маркса, 56 - 159', '1990.08.12', 'военнообязанный'),

('Агилера', 'Кристина', 'Марковна', 'Новосибирск, ул. Петухова, 71 -98', '1999.05.25', 'невоеннообязанный'),

('Васильев', 'Василий', 'Васильевич', 'Новосибирск, ул. Маркса, 32 - 9', '1975.04.09', 'в запасе'),

('Сидоров', 'Игорь', 'Павлович', 'Новосибирск, ул. Красноармейская, 7 - 35', '1976.11.10', 'в запасе'),

('Столптман', 'Марк', 'Генрихович', 'Новосибирск, ул. Октябрьская, 98 - 82', '1999.09.18', 'военнообязанный'),

('Кузнецов', 'Андрей', 'Афанасьевич', 'Новосибирск, ул. Танкистов, 56 - 32', '1984.10.06', 'в запасе'),

('Байден', 'Роберт', 'Степанович', 'НСО, п. Юный Ленинец, ул. Солнечная, 58', '1963.03.18', 'в запасе');

INSERT INTO passport (series, number, date\_issue, issued\_by, id\_employee\_employee)

VALUES (1234, 123456, '1995-12-27', 'УФМС России в Центральном районе г. Новосибирска', 1),

(4525, 952347, '1992.02.21', 'УФМС России в Центральном районе г. Новосибирска', 2),

(1235, 364972, '2004.09.08', 'УФМС России в Ленинском районе г. Новосибирска', 3),

(9547, 246791, '2013.06.27', 'УФМС России в Кировском районе г. Новосибирска', 4),

(6521, 924371, '1989.05.15', 'УФМС России в Ленинском районе г. Новосибирска', 5),

(1326, 748513, '1990.12.04', 'УФМС России в Центральном районе г. Новосибирска', 6),

(7813, 932647, '2013.10.09', 'УФМС России в Октябрьском районе г. Новосибирска', 7),

(9537, 329413, '2003.04.19', 'УФМС России в Дзержинском районе г. Новосибирска', 8),

(5423, 326718, '2012.11.03', 'УФМС России в Кировском районе г. Новосибирска', 9);

INSERT INTO department (dep\_name)

VALUES ('Администрация'),

('Инженерный'),

('Склад'),

('Бухгалтерия');

INSERT INTO staffing (title, rate, id\_department\_department, salary)

VALUES ('Директор', 1, 1, 150000),

('Главный бухгалтер', 1, 4, 95000),

('Технический директор', 1, 2, 95000),

('сервис-инженер', 4, 2, 80000),

('Начальник склада', 1, 3, 50000),

('Грузчик', 2, 3, 35000),

('Бухгалтер', 2, 4, 40000);

INSERT INTO reception (order\_date, cause)

VALUES ('2018.05.31', 'Назначить Иванова И.И. директором - исполнение решения единственного учредителя'),

('2018.05.31', 'Принять Сидорову А.П. гл. бухгалтером'),

('2018.05.31', 'Принять Трамп И.В. техническим директором'),

('2018.06.18', 'Принять Васильева В.В. сервис-инженером'),

('2018.06.18', 'Принять Байдена Р.С. начальником склада'),

('2018.08.03', 'Принять Сидорова И.П. сервис-инженером'),

('2018.09.22', 'Принять Столтмана М.Г. сервис-инженером'),

('2018.10.04', 'Принять Агилеру К.М. бухгалтером'),

('2018.09.22', 'Принять Кузнецова А.А. грузчиком');

INSERT INTO position (amount\_rate, surcharge, id\_employee\_employee,order\_num\_reception, id\_staffing\_staffing)

VALUES (1, 1, 1, 1, 1),

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

(1, 1, 3, 3, 3),

(1.25, 1, 5, 4, 4),

(1, 1.25, 9, 5, 5),

(1, 1, 6, 6, 4),

(1, 1, 7, 7, 4),

(1, 1, 4, 8, 7),

(1, 1, 8, 9, 6);

INSERT INTO business\_trip (city, target, date\_begin)

VALUES ('Норильск', 'Запуск, установка', '2020.01.22'),

('Владивосток', 'Запуск, установка', '2020.02.17'),

('Самара', 'Сервисное обслуживание', '2020.02.27'),

('Екатеринбург', 'Сервисное обслуживание', '2020.03.22'),

('Ижевск', 'Демонтаж, перевозка, запуск', '2020.04.08'),

('Норильск', 'Уже сломали?!...', '2020.06.24'),

('Владимир', 'Сервисное обслуживание', '2020.06.24'),

('Калуга', 'Семинар бухучет форевер', '2020.07.06');

INSERT INTO trip\_report (prepaid\_expense, id\_employee\_employee, id\_trip\_business\_trip)

VALUES (15000, 5, 1),

(15000, 6, 1),

(25000, 7, 2),

(20000, 5, 3),

(20000, 6, 4),

(30000, 3, 5),

(30000, 7, 5),

(16000, 6, 6),

(12000, 6, 7),

(16000, 2, 8),

(12000, 4, 8);

INSERT INTO expenses (purpose\_payment, recipient, sum, id\_trip\_report\_trip\_report)

VALUES ('Авиабилеты', 'Норавиа', 18000, 1),

('Гостиница', 'Норильск', 5400, 1),

('Суточные', 'Васильев', 2500, 1),

('Авиабилеты', 'Норавиа', 18000, 2),

('Гостиница', 'Норильск', 5400, 2),

('Суточные', 'Сидоров', 2500, 2),

('Авиабилеты', 'Владивосток аэро спейс', 28000, 3),

('Суточные', 'Столптман', 2000, 3),

('Авиабилеты', 'Самара аиркрафт', 12000, 4),

('Суточные', 'Васильев', 2500, 4),

('Гостиница', 'Самара-городок', 5400, 4),

('Авиабилеты', 'ЕКБ-авиа', 10000, 5),

('Суточные', 'Сидоров', 2500, 5),

('Билеты', 'РЖД', 4300, 6),

('Гостиница', 'Калашникофф', 10000, 6),

('Суточные', 'Трамп', 2500, 6),

('Билеты', 'РЖД', 4300, 7),

('Гостиница', 'Калашникофф', 10000, 7),

('Суточные', 'Столптман', 2500, 7),

('Авиабилеты', 'Норавиа', 19800, 8),

('Гостиница', 'Северное сияние', 9000, 8),

('Суточные', 'Сидоров', 4320, 8),

('Билеты', 'РЖД', 5200, 9),

('Суточные', 'Сидоров', 3200, 9),

('Суточные', 'Сидорова', 8600, 10),

('Билеты', 'РЖД', 8230, 10),

('Гостиница', 'Калуга хелс энд каре спа ресерч', 33200, 10),

('Суточные', 'Агилера', 8600, 11),

('Билеты', 'РЖД', 8230, 11),

('Гостиница', 'Калуга хелс энд каре спа ресерч', 33200, 11);

INSERT INTO family\_info (marital\_status, id\_employee\_employee)

VALUES ('женат', 1),

('замужем', 2),

('не женат', 3),

('не замужем', 4),

('женат', 5),

('женат', 6),

('не женат', 7),

('не женат', 8),

('женат', 9);

INSERT INTO family\_members (surname, name, patronomic, relation, date\_birth, dependents, id\_family\_family\_info)

VALUES ('Иванова', 'Мария', 'Владимировна', 'супруга', '1985.12.12', FALSE, 1),

('Иванова', 'Дарья', 'Ивановна', 'дочь', '2007.07.24', TRUE, 1),

('Сидоров', 'Игорь', 'Павлович', 'супруг', '1976.11.10', FALSE, 2),

('Сидоров', 'Денис', 'Игоревич', 'сын', '2000.04.17', FALSE, 2),

('Васильева', 'Василина', 'Васильевна', 'супруга', '1974.03.07', FALSE, 5),

('Васильева', 'Наталья', 'Васильевна', 'дочь', '1998.08.16', FALSE, 5),

('Сидорова', 'Агафья', 'Павловна', 'супруга', '1978.01.23', FALSE, 6),

('Сидоров', 'Денис', 'Игоревич', 'сын', '2000.04.17', FALSE, 6),

('Байден', 'Иван', 'Робертович', 'сын', '1989.11.22', FALSE, 9);

Составляем SQL-запросы (выходные документы) и проверяем их на тестовой базе данных:

1. Распределение среднего оклада по отделам.

SELECT id\_department, dep\_name, ROUND(SUM(salary::numeric \* rate::numeric) / SUM(rate::numeric), 2)::money AS average FROM staffing

JOIN department ON department.id\_department = staffing.id\_department\_department

GROUP BY id\_department ORDER BY id\_department;

id\_department | dep\_name | average

----------------+--------------------------+-------------

1 | Администрация | $150,000.00

2 | Инженерный | $83,000.00

3 | Склад | $40,000.00

4 | Бухгалтерия | $58,333.33

(4 rows)

2. Распределение суммы командировочных по отделам.

SELECT id\_department, SUM(sum), dep\_name FROM expenses

JOIN trip\_report ON trip\_report.id\_trip\_report = expenses.id\_trip\_report\_trip\_report

JOIN employee ON employee.id\_employee = trip\_report.id\_employee\_employee

JOIN position ON position.id\_employee\_employee = employee.id\_employee

JOIN staffing ON staffing.id\_staffing = position.id\_staffing\_staffing

JOIN department ON department.id\_department = staffing.id\_department\_department

GROUP BY id\_department ORDER BY id\_department;

id\_department | sum | dep\_name

------------------------+-------------------+-------------

2 | $189,320.00 | Инженерный

4 | $100,060.00 | Бухгалтерия

(2 rows)

Важной задачей по обеспечению безопасности базы данных становится разграничение прав доступа и определение привилегий, позволяющих системным администраторам осуществлять управление, а пользователям получать доступ к данным. Самый частый подход – это создание ролей и назначение этих ролей пользователям. Роль в PostgreSQL – это совокупность разрешений и запретов на доступ к базе данных и ее объектов. В Postgres пользователь -- это такая-же роль, только имеющая пароль.

Может быть организовано несколько уровней безопасности:

– экземпляра (сервера, кластера, инсталяции) аутетификация, управление безопасностью, создание БД

– базы данных: подключение к конкретной БД, создание в ней ролей и т. д.

– схемы: управление схемами (создание, удаление)

– таблицы: CRUD-операции над таблицами

– колонки: операции над конкретной колонкой конкретной таблицы

– строки таблицы

Для нашей базы данных организуем три дополнительные роли и трех пользователей базы данных.

PostgreSQL по умолчанию назначает группе public определённые права для некоторых типов объектов. Поэтому сначала необходимо снять права со схемы public:

REVOKE CREATE ON SCHEMA public FROM public

REVOKE ALL ON DATABASE ax\_postgres FROM public

Теперь можно создавать роли и пользователей.

1. Администратор базы данных.

Роль admin\_db – не является superuser, может использовать и содавать в схеме public, подключаться к базе данных, создавать базы данных, и имеет все права на все таблицы:

CREATE ROLE admin\_db

GRANT CONNECT ON DATABASE ax\_postgres TO admin\_db

GRANT USAGE ON SCHEMA public TO admin\_db

GRANT CREATE ON SCHEMA public TO admin\_db

GRANT CREATE ON DATABASE ax\_postgres TO admin\_db

GRANT SELECT, INSERT, UPDATE, DELETE, REFERENCES, TRIGGER ON ALL TABLES IN SCHEMA public TO admin\_db

Пользователь с ролью admin\_db:

CREATE USER sidorov WITH PASSWORD '1122'

GRANT admin\_db TO sidorov

2. Директор.

Роль director – может подключатся к БД, может использовать схему public, делать выборку, изменять, удалять, обновлять все таблицы в БД:

CREATE ROLE director

GRANT CONNECT ON DATABASE ax\_postgres TO director

GRANT USAGE ON SCHEMA public TO director

GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA public TO director

Пользователь с ролью director:

CREATE USER ivanov WITH PASSWORD '1111'

GRANT director TO ivanov

3. Инженер.

Роль engineer – может подключатся к БД, может использовать схему public, делать выборку, изменять, удалять, обновлять только таблицы employee, trip\_report, business\_trip, expenses в БД:

CREATE ROLE engineer

GRANT CONNECT ON DATABASE ax\_postgres TO engineer

GRANT USAGE ON SCHEMA public TO engineer

GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE

public.employee, public.trip\_report, public.business\_trip, public.expenses

TO engineer

Пользователь с ролью engineer:

CREATE USER trump WITH PASSWORD '2222'

GRANT engineer TO trump

Можно подключиться к БД и проверить созданные роли:

ax\_postgres=# SELECT rolname, rolsuper, rolcreaterole, rolcreatedb, rolcanlogin, oid FROM pg\_roles;

rolname | rolsuper | rolcreaterole | rolcreatedb | rolcanlogin | oid

--------------------------------------------+--------------+--------------------+-------------------+-----------------+-------

pg\_monitor | f | f | f | f | 3373

pg\_read\_all\_settings | f | f | f | f | 3374

pg\_read\_all\_stats | f | f | f | f | 3375

pg\_stat\_scan\_tables | f | f | f | f | 3377

pg\_read\_server\_files | f | f | f | f | 4569

pg\_write\_server\_files | f | f | f | f | 4570

pg\_execute\_server\_program | f | f | f | f | 4571

pg\_signal\_backend | f | f | f | f | 4200

ax\_postgres | t | t | t | t | 10

admin\_db | f | f | f | f | 16518

sidorov | f | f | f | t | 16519

director | f | f | f | f | 16520

ivanov | f | f | f | t | 16521

engineer | f | f | f | f | 16522

trump | f | f | f | t | 16523

(15 rows)