Міністерство освіти і науки України

Харківський національний університет імені В.Н. Каразіна
Кафедра теоретичної та прикладної інформатики

Звіт по дисципліні
Вступ до SQL баз даних

Індивідуальне завдання № 4

Студента: Чистякова Артема
Групи: МФ-31

Необхідний термін здачі завдання: 12.11.2020

Фактичний термін здачі завдання: _____

Кількість балів: _____

Харків 2020

Постановка задачи

Разработать базу данных для хранения и обработки информации о внутреннем устройстве некоторой небольшой аутсорс IT-компании.

Компания хоть и небольшая, но может располагать несколькими филиалами. В каждом из офисов работают: менеджеры, программисты и дизайнеры. У каждого менеджера в подчинении могут находится другие менеджеры, а также программисты и дизайнеры. У каждого программиста может быть программист ментор высшей должности.

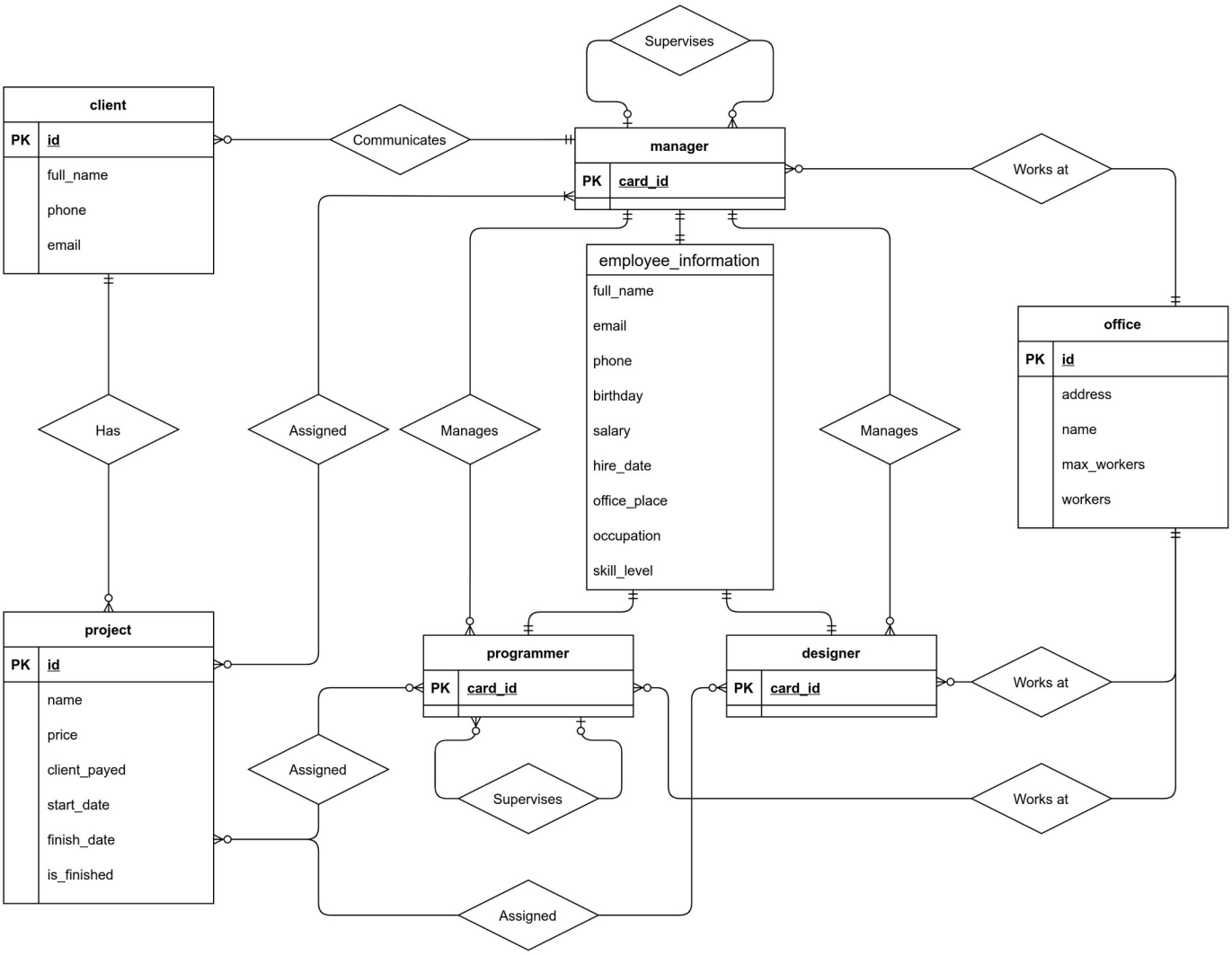
В компанию поступают заказы от клиентов в качестве проектов. Каждому клиенту предоставляется менеджер для общения, и над поступившими проектами начинается работа. В проекте могут быть задействованы менеджеры, программисты и дизайнеры.

В базе данных должна храниться информация о каждом сотруднике, это личные данные, зарплата, место в офисе и направление работы (для программиста, например, Java или C++).

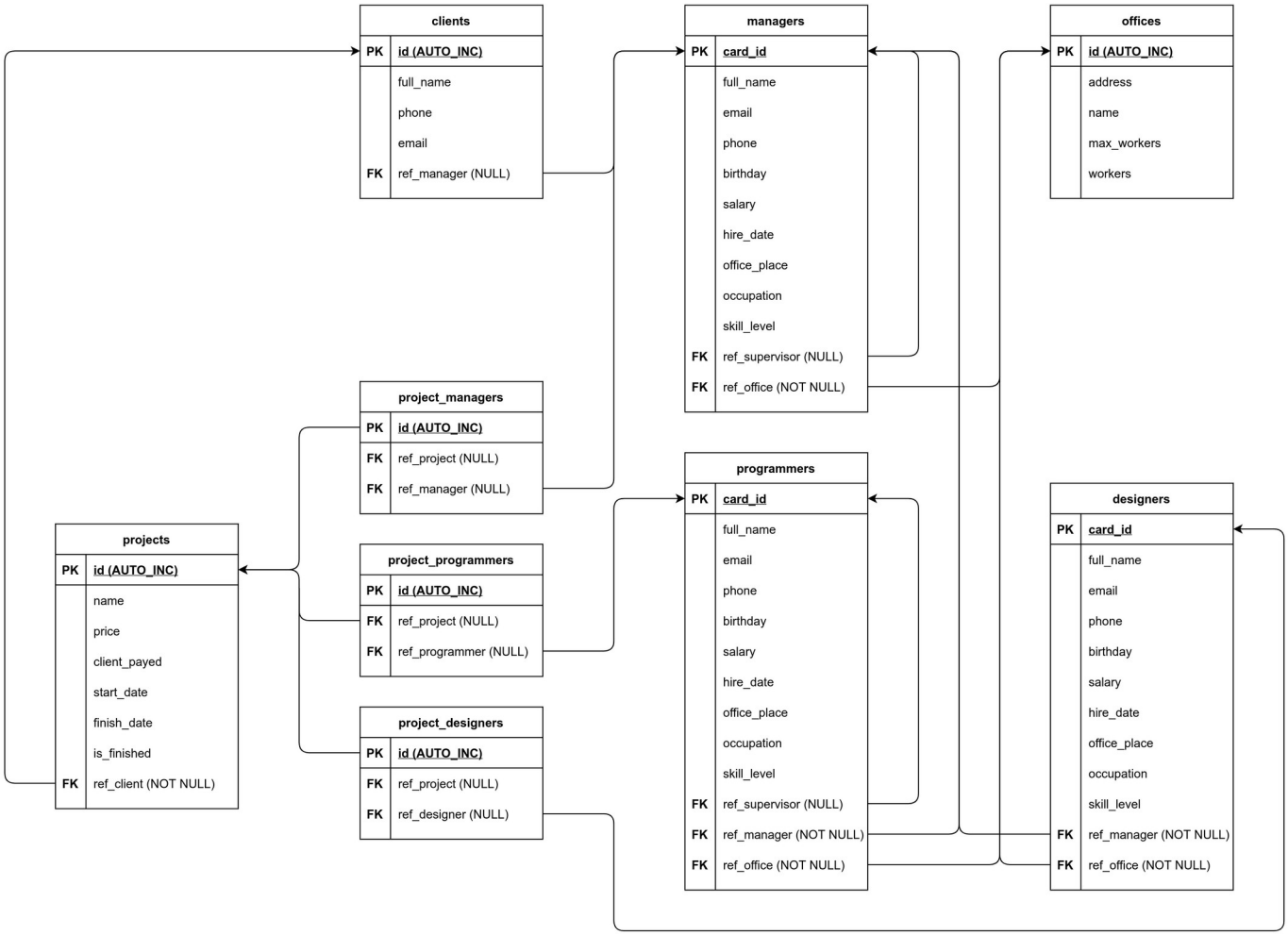
О проектах в базе данных тоже должна храниться информация, это стоимость проекта, текущая оплата клинта, время начала работы и дедлайн.

В базе данных также должна храниться информация о офисе.

Концептуальная модель БД



Представление БД в качестве таблиц и связей



Перечень запросов на выборку

1. Вывести имена всех менеджеров, работающих в «Sweaty office».

```
// объединение таблиц

SELECT full_name FROM managers
INNER JOIN offices ON ref_office = id
WHERE name = "Sweaty office";

// подчиненный запрос

SELECT full_name FROM managers
WHERE ref_office =
(SELECT id FROM offices
WHERE name = "Sweaty office");
```

2. Вывести имена клиентов, у которых менеджер – Abraham Lincoln.

```
// объединение таблиц

SELECT clients.full_name FROM clients
INNER JOIN managers ON ref_manager = card_id
WHERE managers.full_name = "Abraham Lincoln";

// подчиненный запрос

SELECT full_name FROM clients
WHERE ref_manager =
(SELECT card_id FROM managers
WHERE full_name = "Abraham Lincoln");
```

3. Вывести названия проектов, над которыми работает более одного программиста.

```
// объединение таблиц

SELECT projects.name FROM project_programmers
INNER JOIN projects ON project_programmers.ref_project =
projects.id
GROUP BY project_programmers.ref_project
HAVING COUNT(*) > 1;

// подчиненный запрос

SELECT name FROM projects
```

```
WHERE id IN
(SELECT ref_project FROM project_programmers
GROUP BY ref_project
HAVING COUNT(*) > 1);
```

4. Вывести имена менеджеров, у которых в подчинении более одного менеджера.

```
// объединение таблиц

SELECT sup.full_name FROM managers
INNER JOIN managers as sup ON managers.ref_supervisor =
sup.card_id
GROUP BY sup.full_name
HAVING COUNT(*) > 1;

// подчиненный запрос

SELECT full_name FROM managers
WHERE card_id IN
(SELECT ref_supervisor FROM managers
GROUP BY ref_supervisor
HAVING COUNT(*) > 1);
```

5. Вывести имена всех программистов, у которых менеджер – Senior.

```
// объединение таблиц

SELECT programmers.full_name FROM programmers
INNER JOIN managers ON programmers.ref_manager =
managers.card_id
WHERE managers.skill_level = "Senior";

// подчиненный запрос

SELECT full_name FROM programmers
WHERE ref_manager IN
(SELECT card_id FROM managers
WHERE skill_level = "Senior");
```

6. Вывести имена всех дизайнеров с зарплатой выше среднего.

```
// нельзя выполнить в помощью объединения таблиц

SELECT full_name FROM designers
WHERE salary >
```

```
(SELECT avg(salary) FROM designers);
```

7. Вывести суммарную зарплату всех сотрудников.

```
// нельзя выполнить с помощью объединения таблиц

SELECT sum(
(SELECT sum(salary) FROM managers) +
(SELECT sum(salary) FROM programmers) +
(SELECT sum(salary) FROM designers)) AS "total salary";
```

8. Вывести имена всех менеджеров, у которых зарплата больше каждого из программистов.

```
// нельзя выполнить с помощью объединения таблиц

SELECT full_name FROM managers
WHERE salary > ALL
(SELECT salary FROM programmers);
```

9. Вывести названия офисов, в которых не работают менеджеры.

```
// нельзя выполнить с помощью объединения таблиц

SELECT name FROM offices
WHERE NOT EXISTS
(SELECT * FROM managers WHERE ref_office = id);
```

10. Вывести количество Junior в компании.

```
// нельзя выполнить с помощью объединения таблиц

SELECT sum(
(SELECT count(*) FROM managers WHERE skill_level = "Junior") +
(SELECT count(*) FROM programmers WHERE skill_level =
"Junior") +
(SELECT count(*) FROM designers WHERE skill_level = "Junior"))
AS "total juniors";
```