МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Модульний контроль

з дисципліни «Бази даних та інформаційні системи»

варіант 1

Студента групи КБ-01 _____ Борща Д.О.
підпис

Перевірив Кузіков Б.О.

СУМИ 2022

Постановка задачі

Ми розробляємо інформаційну систему (ІС) «Дистанційне навчання». Підчас співбесіди із представниками замовника виявлено наступні дані.

- ІС «Дистанційне навчання» містить інформацію про:
  - студентів та дисципліни, які вони вивчають
  - викладачів, що ведуть дисципліни
  - завдання, які необхідно виконати студентам в рамках дисципліни.
- Деканат підключає студентів та викладачів до дисциплін.
- Студенти надсилають завдання, а викладачі їх перевіряють.
- Деканат має можливість продивитись підсумковий бал студента за дисципліною.

Необхідно виконати моделювання ІС, та представити результати у вигляді DF та ER діаграм та сценарію створення таблиць мовою SQL.

1 Data Flow Diagram

Результати моделювання ІС представлені у вигляді DF-діаграм 0-го та 1-го рівня на рисунках 1.1, 1.2.

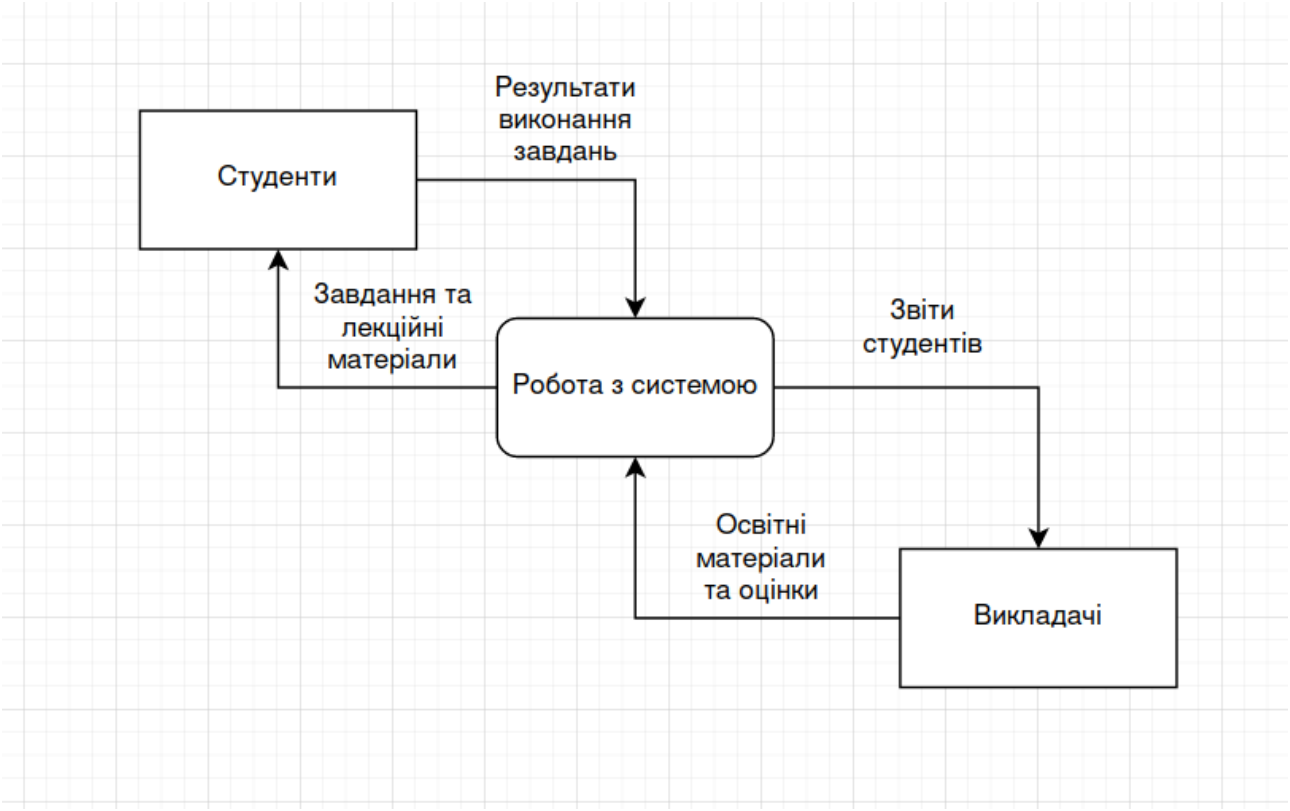

Рисунок 1.1. DF-діаграма 0-го рівня ІС «Дистанціне навчання»

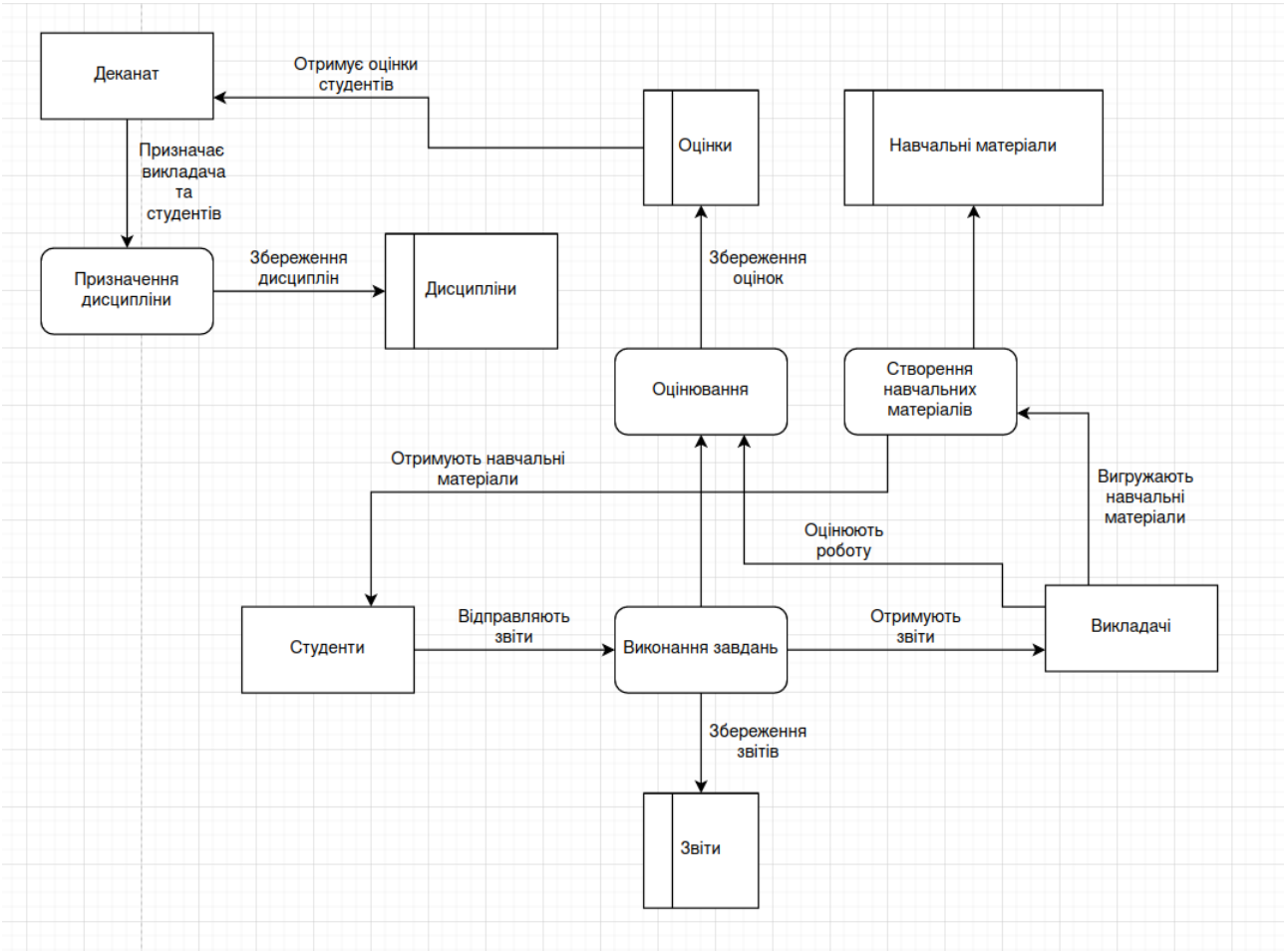

Рисунок 1.2. DF-діаграма 1-го рівня ІС «Дистанціне навчання»

При аналізі вхідних даних щодо ІС будо виділені наступні елементи:

Сутності:

1) Людина — містить в собі відомості про осіб, що беруть участь в навчальному процесі;
2) Дисципліна — містить в собі відомості про дисципліну, викладача та студентів, що навчаються;
3) Звіти — сховище звітів;
4) Навчальні матеріали — сховище навчальних матеріалів;
5) Оцінки — сховище оцінок.

Процеси:

1) Призначення дисципліни — процес створення нової дисципліни та призначення учасників навчального процесу;

2) Створення навчальних матеріалів — процес вигружання навчальних матеріалів викладачем на ресурс;
3) Виконання завдань — процес виконання завдань студентом;
4) Оцінювання — процес оцінювання викладачем після Виконання студентом.

2 Entity Relation Diagram

На основі моделювання процесів виділені основні сховища даних, які у подальшому будуть перетворені у відповідні таблиці бази даних. Структура сховища даних, що проектується представлена на рисунку 2.1.

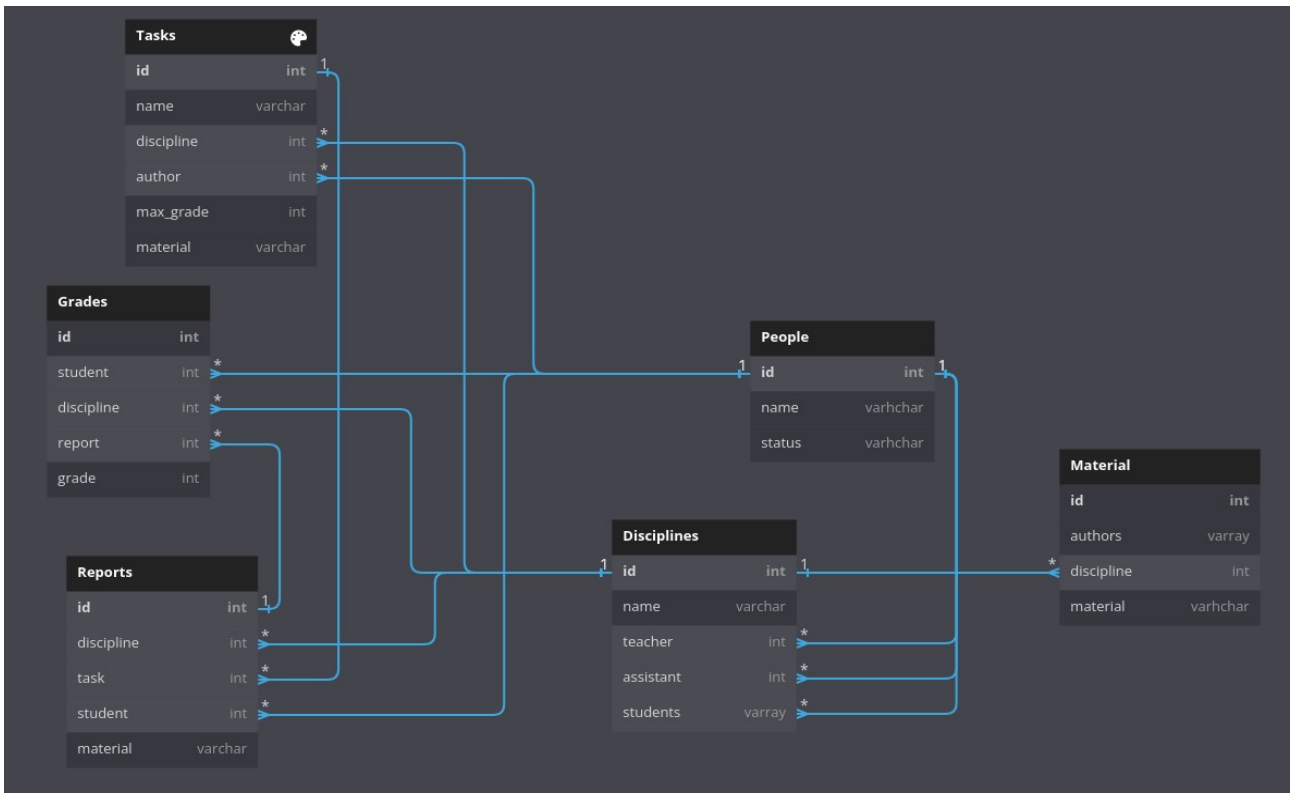

Рисунок 2.1. ER-діаграма ІС «Дистанціне навчання»

Опис представлених на діаграмі сутностей та їх атрибутів наведено у таблиці 2.1. Назви атрибутів та сутностей у таблиці 2.1 збігаються із позначеннями на рисунку 2.1. У дужках позначені назви відповідних елементів, що використані у сценарії створення сховища даних, якщо є відмінності.

Таблиція 2.1 Опис сутностей та атрибутів ER-діаграми

Атрибут	**Обов'-язковий**	**Тип**	**Опис**	**Обмеження**
Дисципліна (Discipline) — зберігає відомості щодо дисциплін, які вивчаються студентами				
id	+	Number	ID дисципліни	PK
name	+	Number	Ім'я дисципліни	
teacher	+	Number	ID викладача	

assistant	-	Number	ID асистента	
students	+	Varray of Number	ID студентів	
Люди (People) — зберігає всіх людей, що приймають участь в навчальному процесі				
id	+	Number	ID людини	PK
name	+	Varchar	Ім’я людини	
status	+	Varchar	Посада (викладач/студент)	
Навчальні матеріали (Material) — сховище навчальних матеріалів				
id	+	Number	ID матеріалу	PK
author	+	Number	ID автору	
discipline	+	Number	ID дисципліни	
material	+	Varchar	Посилання на матеріал на вебсервері	
Звіти (Reports) — сховище звітів				
id	+	Number	ID звіту	PK
discipline	+	Number	ID дисципліни	
task	+	Number	ID завдання	
student	+	Number	ID студента	
material	+	Varchar	Посилання на звіт на вебсервері	
Оцінка (Grades) — сховище оцінок				
id	+	Number	ID оцінки	PK
student	+	Number	ID студента	
discipline	+	Number	ID дисципліни	
report	+	Number	ID звіту	
grade	+	Number	Оцінка	
Завдання (Tasks) — завдання до дисциплін				
id	+	Number	ID завдання	PK
name	+	Varchar	Назва завдання	
discipline	+	Number	ID дисципліни	
author	+	Number	ID автору	
max_grade	+	Number	Максимальний бал за завдання	
material	+	Varchar	Посилання на завдання на вебсервері	

На основі ER-діаграми, розроблено сценарій створення структури сховища даних із використанням діалекта мови SQL для СКБД Oracle Database 18c Express Edition.

```
CREATE OR REPLACE TYPE people_array IS VARRAY (30) OF NUMBER;
/

CREATE TABLE People (
  id number(10) PRIMARY KEY,
  name varchar(150) NOT NULL,
```

```
  status varchar(50) NOT NULL
);

CREATE SEQUENCE People_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER People_seq_tr
 BEFORE INSERT ON People FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT People_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

CREATE TABLE Disciplines (
  id number(10) PRIMARY KEY,
  name varchar(100) NOT NULL,
  teacher number(10) NOT NULL,
  assistant number(10),
  students people_array NOT NULL
);

CREATE SEQUENCE Disciplines_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER Disciplines_seq_tr
 BEFORE INSERT ON Disciplines FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT Disciplines_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

CREATE TABLE Tasks (
  id number(10) PRIMARY KEY,
  name varchar(100) NOT NULL,
  discipline number(10) NOT NULL,
  author number(10) NOT NULL,
  max_grade number(10),
  material varchar(255) NOT NULL
);

CREATE SEQUENCE Tasks_seq START WITH 1 INCREMENT BY 1;
```

```
CREATE OR REPLACE TRIGGER Tasks_seq_tr
 BEFORE INSERT ON Tasks FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT Tasks_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

CREATE TABLE Material (
  id number(10) PRIMARY KEY,
  authors people_array NOT NULL,
  discipline number(10) NOT NULL,
  material varchar(255) NOT NULL
);

CREATE SEQUENCE Material_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER Material_seq_tr
 BEFORE INSERT ON Material FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT Material_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

CREATE TABLE Reports (
  id number(10) PRIMARY KEY,
  discipline number(10) NOT NULL,
  task number(10) NOT NULL,
  student number(10) NOT NULL,
  material varchar(255) NOT NULL
);

CREATE SEQUENCE Reports_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER Reports_seq_tr
 BEFORE INSERT ON Reports FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT Reports_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/
```

```
CREATE TABLE Grades (
  id number(10) PRIMARY KEY,
  student number(10) NOT NULL,
  discipline number(10) NOT NULL,
  report number(10) NOT NULL,
  grade number(10) NOT NULL
);


CREATE SEQUENCE Grades_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER Grades_seq_tr
 BEFORE INSERT ON Grades FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT Grades_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

ALTER TABLE Reports ADD FOREIGN KEY (student) REFERENCES People
(id);
ALTER TABLE Reports ADD FOREIGN KEY (task) REFERENCES Tasks (id);
ALTER TABLE Disciplines ADD FOREIGN KEY (teacher) REFERENCES
People (id);
ALTER TABLE Disciplines ADD FOREIGN KEY (assistant) REFERENCES
People (id);
ALTER TABLE Grades ADD FOREIGN KEY (student) REFERENCES People
(id);
ALTER TABLE Material ADD FOREIGN KEY (discipline) REFERENCES
Disciplines (id);
ALTER TABLE Reports ADD FOREIGN KEY (discipline) REFERENCES
Disciplines (id);
ALTER TABLE Grades ADD FOREIGN KEY (discipline) REFERENCES
Disciplines (id);
ALTER TABLE Grades ADD FOREIGN KEY (report) REFERENCES Reports
(id);

INSERT INTO People (name, status) VALUES ('Borys', 'teacher');
INSERT INTO People (name, status) VALUES ('Dmytro', 'student');
INSERT INTO People (name, status) VALUES ('Petro', 'student');
```

```
INSERT INTO People (name, status) VALUES ('Danil', 'student');

INSERT INTO Disciplines (name, teacher, students) VALUES ('Data
Bases', 1, people_array(2, 3, 4));

INSERT INTO Material (authors, discipline, material) VALUES
(people_array(1), 1, 'material_url');
INSERT INTO Material (authors, discipline, material) VALUES
(people_array(1), 1, 'material_url');

INSERT INTO Tasks (name, discipline, author, max_grade, material)
VALUES ('First task', 1, 1, 100, 'material_url1');
INSERT INTO Tasks (name, discipline, author, max_grade, material)
VALUES ('Second task', 1, 1, 100, 'material_url2');

INSERT INTO Reports (discipline, task, student, material) VALUES
(1, 1, 2, 'report_url1');
INSERT INTO Reports (discipline, task, student, material) VALUES
(1, 2, 2, 'report_url2');
INSERT INTO Reports (discipline, task, student, material) VALUES
(1, 1, 3, 'report_url3');
INSERT INTO Reports (discipline, task, student, material) VALUES
(1, 1, 4, 'report_url4');
INSERT INTO Reports (discipline, task, student, material) VALUES
(1, 2, 4, 'report_url5');

INSERT INTO Grades (student, discipline, report, grade) VALUES (2,
1, 1, 5);
INSERT INTO Grades (student, discipline, report, grade) VALUES (2,
1, 2, 4);
INSERT INTO Grades (student, discipline, report, grade) VALUES (3,
1, 3, 2);
INSERT INTO Grades (student, discipline, report, grade) VALUES (4,
1, 4, 3);
INSERT INTO Grades (student, discipline, report, grade) VALUES (4,
1, 5, 4);
```

Уточнюючи отримані дані, були виділені такі факти:

Завдання «на 60%» (обов'язкове)

1. Студент може вивчати кілька дисциплін.
2. Викладач може викладати кілька дисциплін.
3. В одній дисципліні може бути декілька завдань.
4. Про кожне завдання ми знаємо його назву, до якої дисципліни воно відноситься і максимальний бал.
5. Студент відправляє завдання та воно оцінюється викладачем 1 раз (зберігається, хто оцінив і на який бал).

Завдання на 74% = обов'язкове +

6. Завдання за варіантами з частини DFD.
7. В одній дисципліні може бути кілька викладачів (наприклад, лекції/практики)

Завдання «100%» = «на 74%» +

8. Студент може надсилати кілька варіантів вирішення завдання.

Перевірка правильності ERD

«на 60%»

1) Як гарантується, що до дисципліни підключено викладача (студента), який дійсно існує? (про якого в ІС є всі необхідні дані)

Бо використовуються зовнішні ключі

2) Як гарантується, що в одній дисципліні людина не може бути одночасно викладачем та студентом?

Можемо лише перевірити посаду людини на стороні контролера

На 74% та вище

3) В рамках курсів підвищення кваліфікації (ФПК) деякі викладачі можуть бути підключені до дисципліни як студенти.

Так, можуть.

4) Викладачі поскаржилися, що вони дуже завантажені. Тому у деякі дисципліни додали помічника (помічників) викладача. Вони також можуть перевіряти роботи.

Додано поле «асистент»

5) Що потрібно зробити, якщо знадобиться нова категорія користувачів (гість, відпрацювання заборгованостей...)?

Можна вписати будь яку посаду новій людині в БД

3 Нормалізація структури БД

На рисунку 3.1 наведено ER-діаграму ІС авіакомпанії. ER-діаграма сховища даних ІС авіакомпанії після нормалізації до 3-ї нормальної форми наведена на рисунку 3.2.

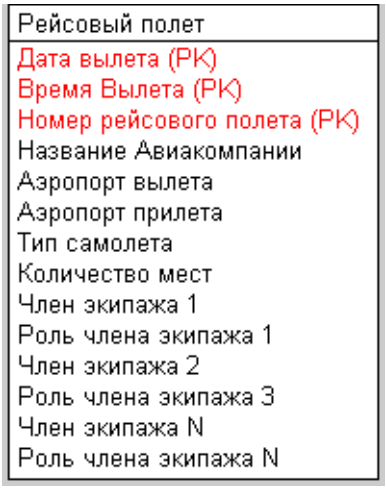

Рисунок 3.1. ER-діаграма ІС авіакомпанії (згідно завдвння)

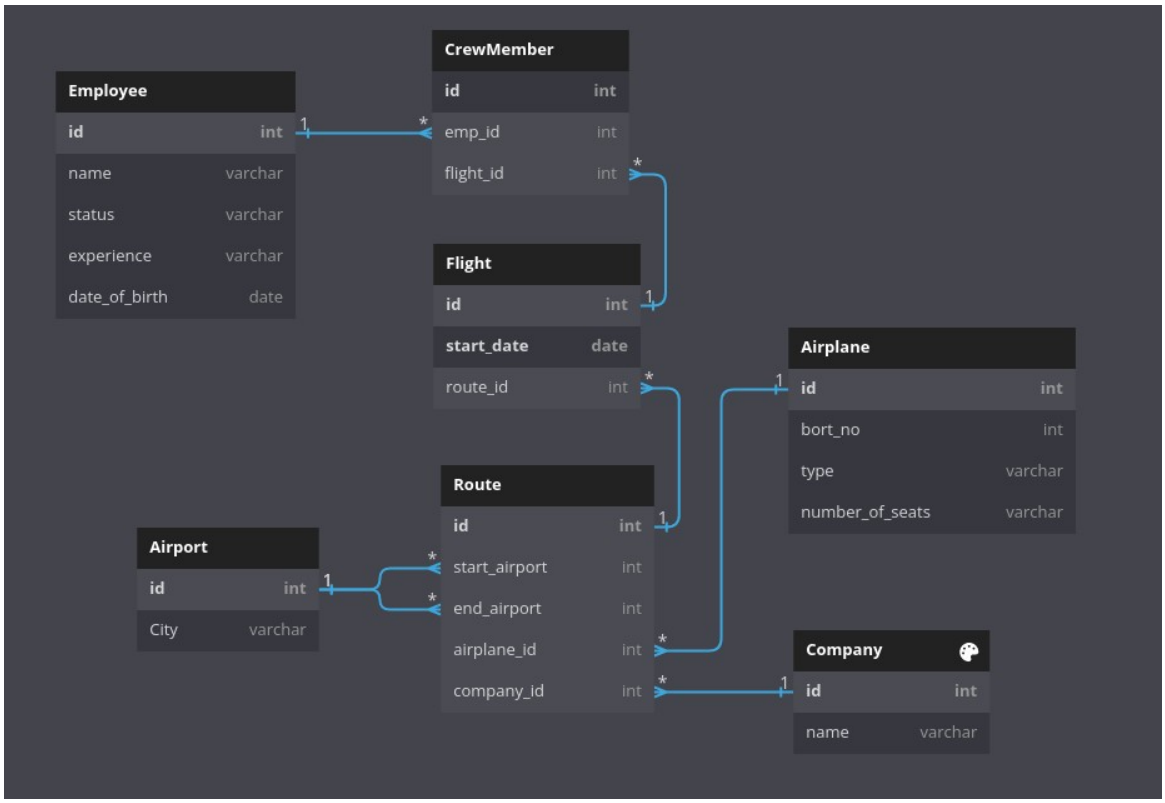

Рисунок 3.2. ER-діаграма сховища даних ІС авіакомпанії нормалізована до 3-ї нормальної форми

На основі ER-діаграми зображеної на рисунку 3.2, розроблено сценарій створення структури сховища даних із використанням діалекта мови SQL для СКБД Oracle Database 18c Express Edition.

```
CREATE TABLE Employee (
  id number(10) PRIMARY KEY,
  name varchar2(255),
  status varchar2(255),
  experience varchar2(255),
  date_of_birth date
);

CREATE SEQUENCE Employee_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER Employee_seq_tr
 BEFORE INSERT ON Employee FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT Employee_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

CREATE TABLE CrewMember (
  id number(10) PRIMARY KEY,
  emp_id number(10),
  flight_id number(10)
);

CREATE SEQUENCE CrewMember_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER CrewMember_seq_tr
 BEFORE INSERT ON CrewMember FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT CrewMember_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

CREATE TABLE Route (
  id number(10) PRIMARY KEY,
  start_airport number(10),
  end_airport number(10),
```

```
  airplane_id number(10),
  company_id number(10)
);

CREATE SEQUENCE Route_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER Route_seq_tr
 BEFORE INSERT ON Route FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT Route_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

CREATE TABLE Flight (
  id number(10),
  start_date date,
  route_id number(10),
  PRIMARY KEY (id, start_date)
);

CREATE SEQUENCE Flight_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER Flight_seq_tr
 BEFORE INSERT ON Flight FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT Flight_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

CREATE TABLE Airport (
  id number(10) PRIMARY KEY,
  City varchar2(255)
);

CREATE SEQUENCE Airport_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER Airport_seq_tr
 BEFORE INSERT ON Airport FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
```

```
 SELECT Airport_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

CREATE TABLE Company (
  id number(10) PRIMARY KEY,
  name varchar2(255)
);

CREATE SEQUENCE Company_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER Company_seq_tr
 BEFORE INSERT ON Company FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT Company_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

CREATE TABLE Airplane (
  id number(10) PRIMARY KEY,
  bort_no number(10),
  type varchar2(255),
  number_of_seats varchar2(255)
);

CREATE SEQUENCE Airplane_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER Airplane_seq_tr
 BEFORE INSERT ON Airplane FOR EACH ROW
 WHEN (NEW.id IS NULL)
BEGIN
 SELECT Airplane_seq.NEXTVAL INTO :NEW.id FROM DUAL;
END;
/

ALTER TABLE CrewMember ADD FOREIGN KEY (flight_id) REFERENCES
Flight (id);
ALTER TABLE Flight ADD FOREIGN KEY (route_id) REFERENCES Route
(id);
ALTER TABLE CrewMember ADD FOREIGN KEY (emp_id) REFERENCES
Employee (id);
```

```
ALTER TABLE Route ADD FOREIGN KEY (start_airport) REFERENCES Airport (id);
ALTER TABLE Route ADD FOREIGN KEY (end_airport) REFERENCES Airport (id);
ALTER TABLE Route ADD FOREIGN KEY (company_id) REFERENCES Company (id);
ALTER TABLE Route ADD FOREIGN KEY (airplane_id) REFERENCES Airplane (id);
```

4. Очне завдання модульного контролю

Залишить це лист порожнім. На ньому ви будете виконувати очну частину завдання модульного контролю.