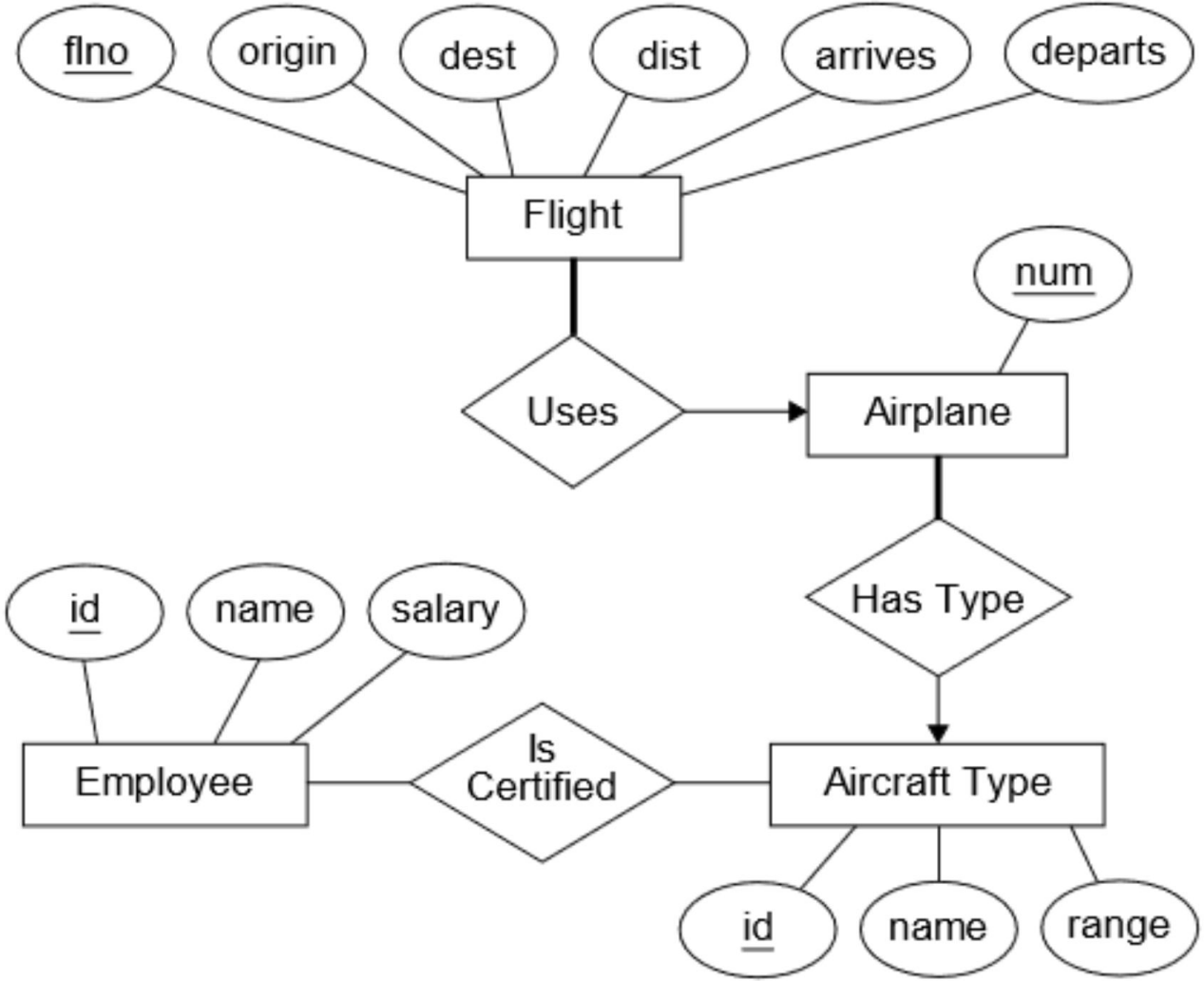
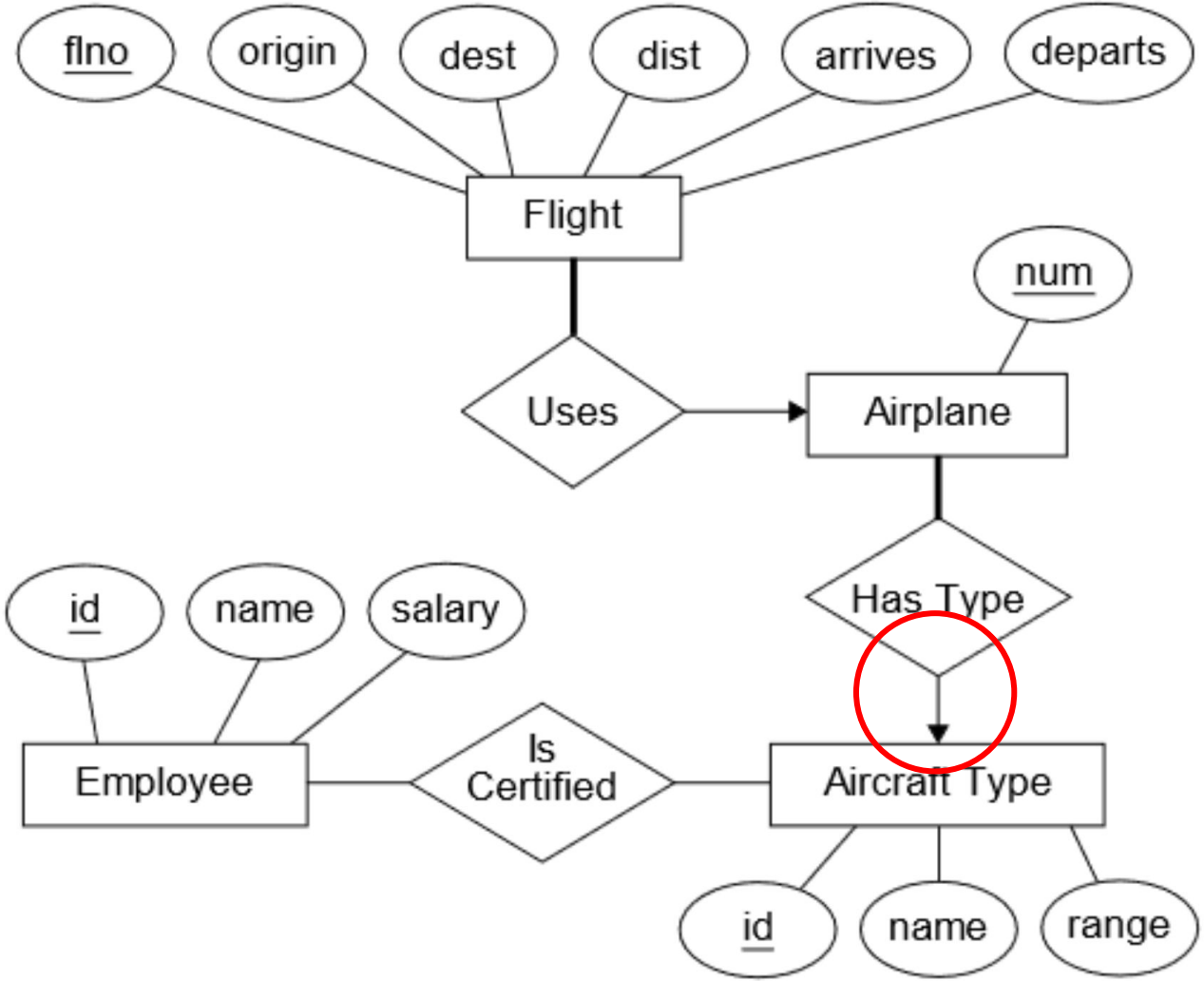
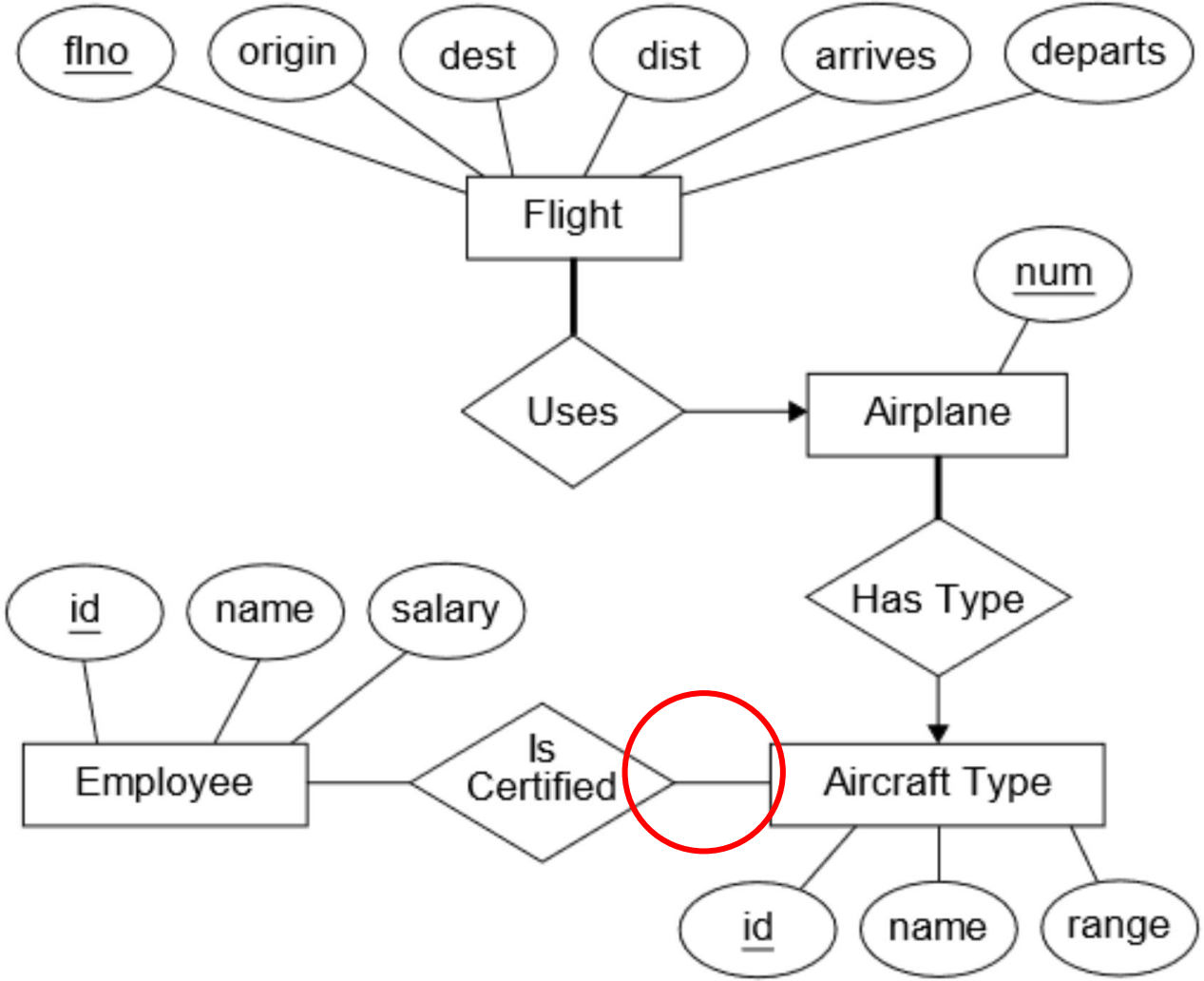


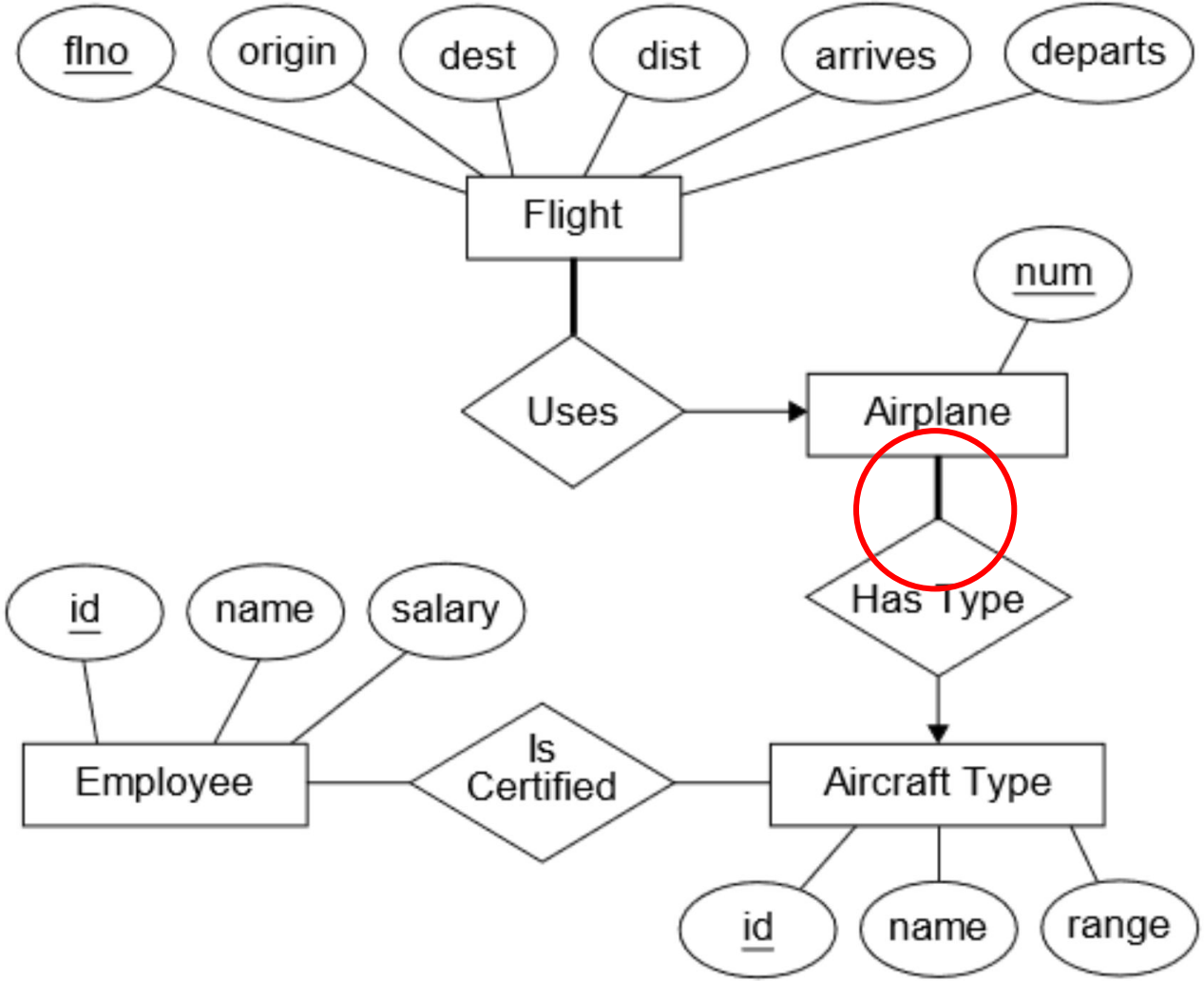
- The *Employees* entity set includes both pilots and other kinds of employees.
- The *Is Certified* relationship set specifies the types of aircraft that a pilot is certified to fly.
- The *Uses* relationship set specifies the airplane used by a particular flight.
- The *Has Type* relationship set specifies a particular airplane's aircraft type.



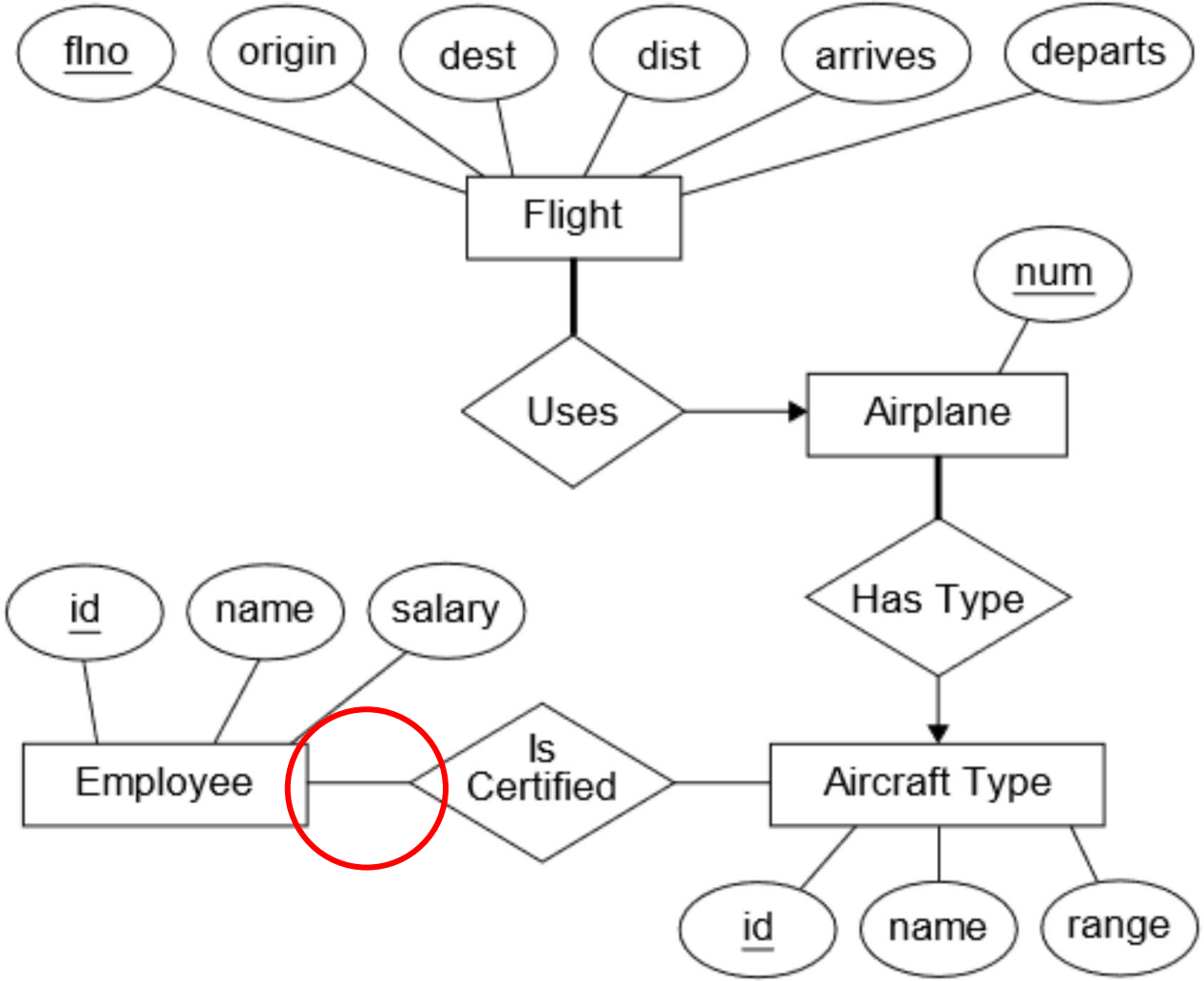
- The *Employees* entity set includes both pilots and other kinds of employees.
- The *Is Certified* relationship set specifies the types of aircraft that a pilot is certified to fly.
- The *Uses* relationship set specifies the airplane used by a particular flight.
- The *Has Type* relationship set specifies a particular airplane's aircraft type.



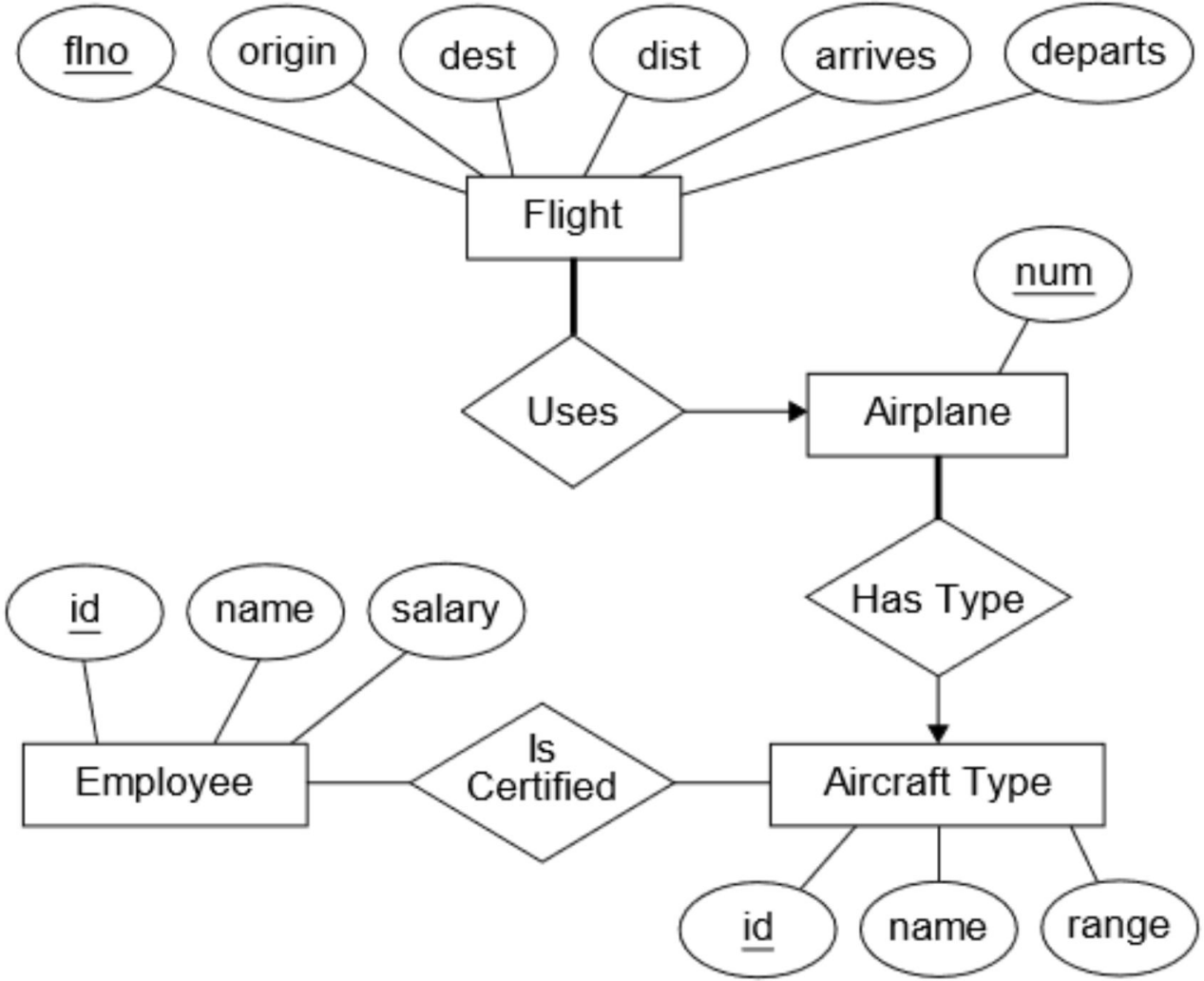
- The *Employees* entity set includes both pilots and other kinds of employees.
- The *Is Certified* relationship set specifies the types of aircraft that a pilot is certified to fly.
- The *Uses* relationship set specifies the airplane used by a particular flight.
- The *Has Type* relationship set specifies a particular airplane's aircraft type.



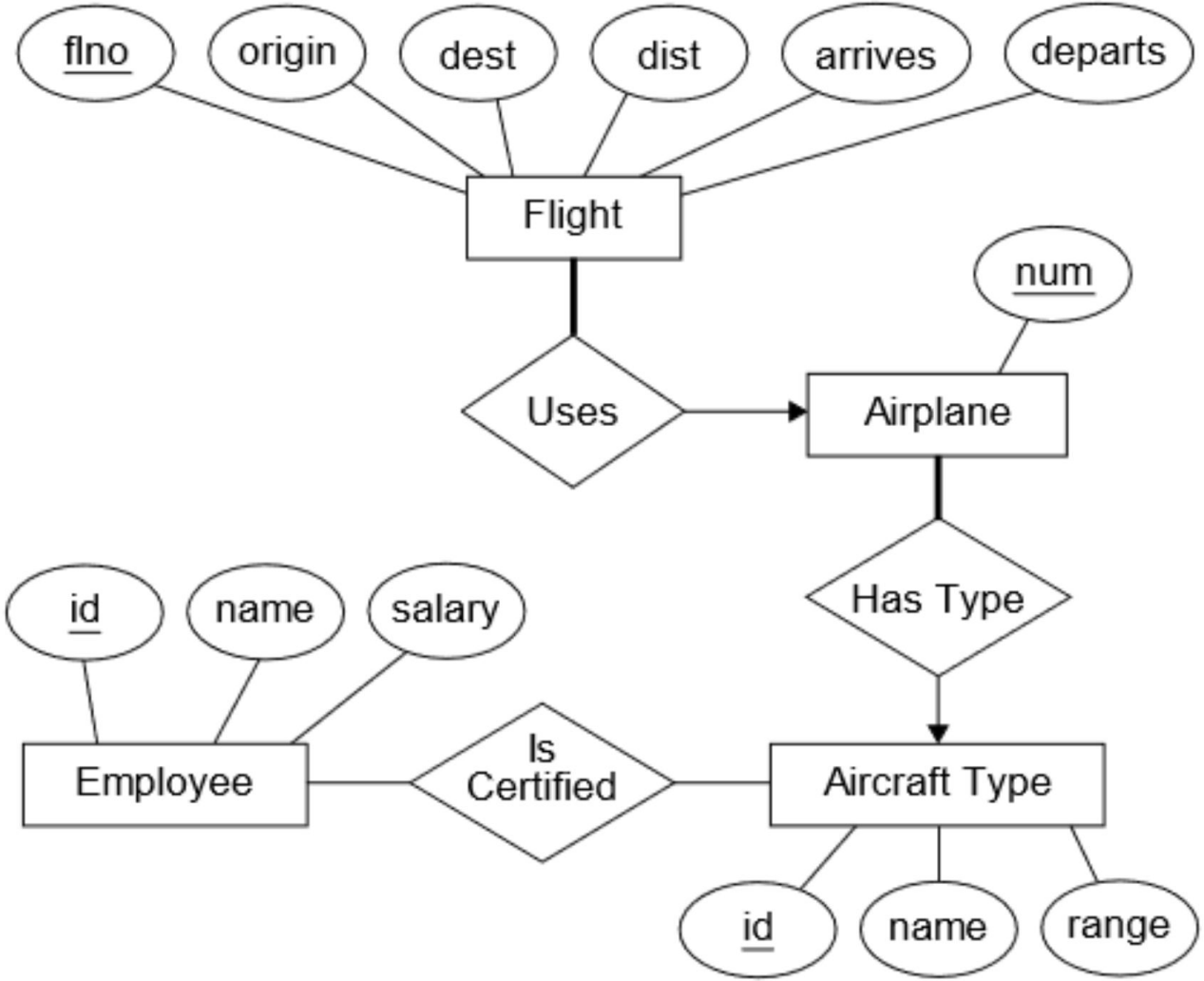
- The *Employees* entity set includes both pilots and other kinds of employees.
- The *Is Certified* relationship set specifies the types of aircraft that a pilot is certified to fly.
- The *Uses* relationship set specifies the airplane used by a particular flight.
- The *Has Type* relationship set specifies a particular airplane's aircraft type.



- The *Employees* entity set includes both pilots and other kinds of employees.
- The *Is Certified* relationship set specifies the types of aircraft that a pilot is certified to fly.
- The *Uses* relationship set specifies the airplane used by a particular flight.
- The *Has Type* relationship set specifies a particular airplane's aircraft type.

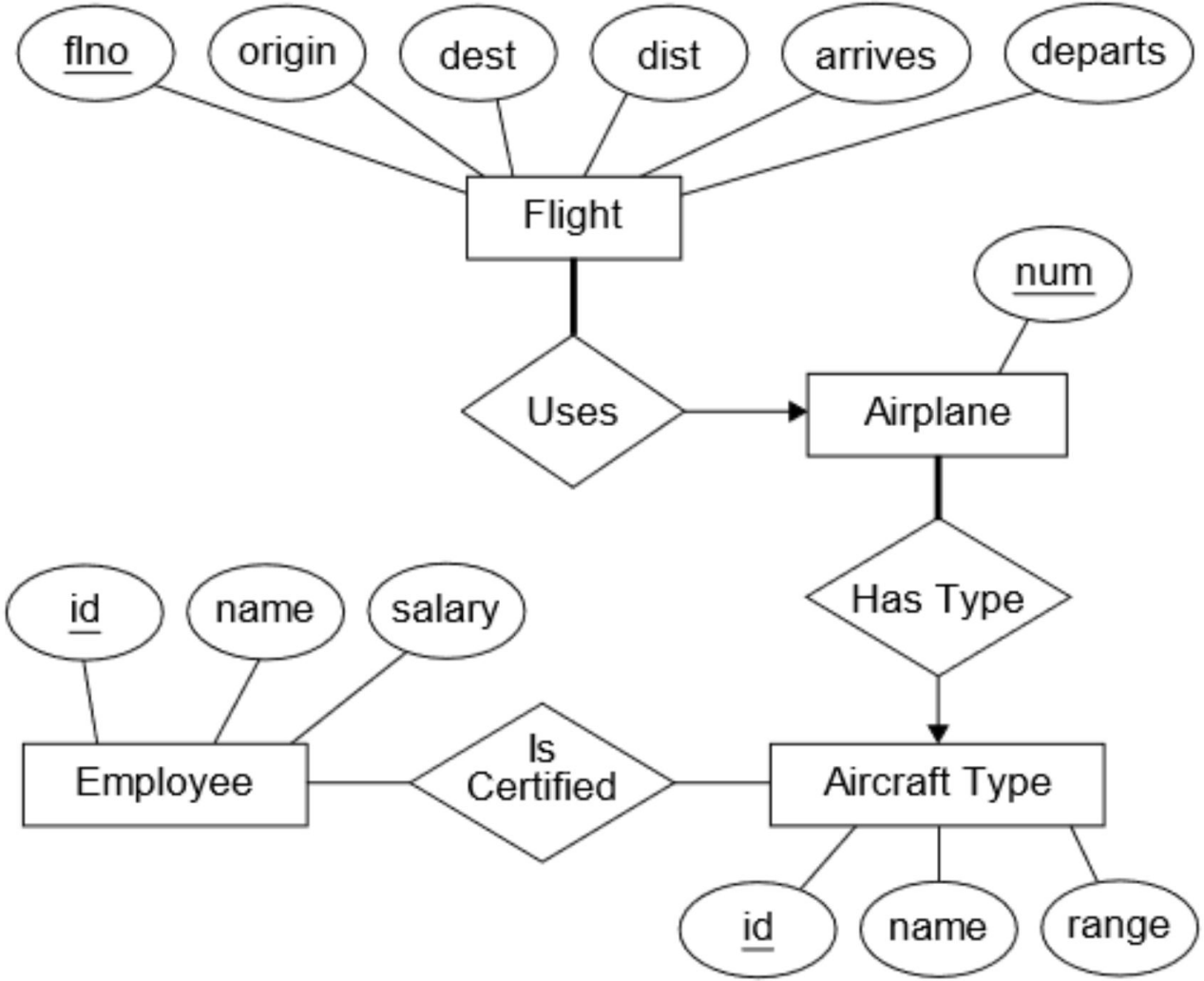


In transforming this diagram to a relational schema, would we need a separate relation to represent the *Has Type* relationship set? Why or why not?

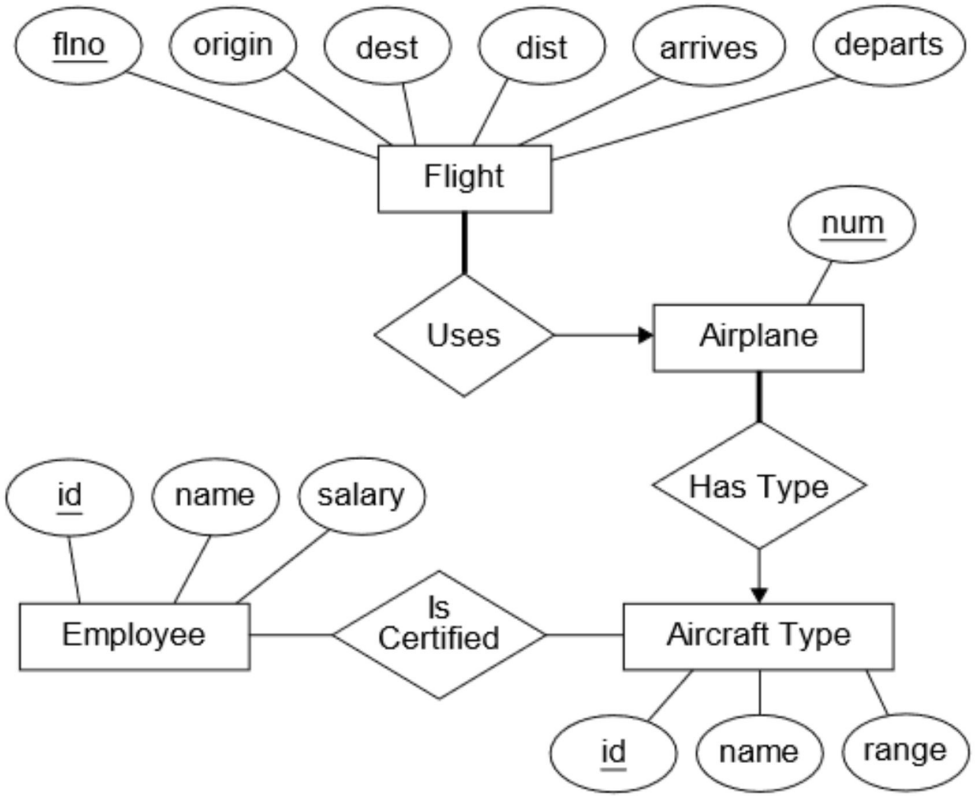


In transforming this diagram to a relational schema, would we need a separate relation to represent the *Has Type* relationship set? Why or why not?

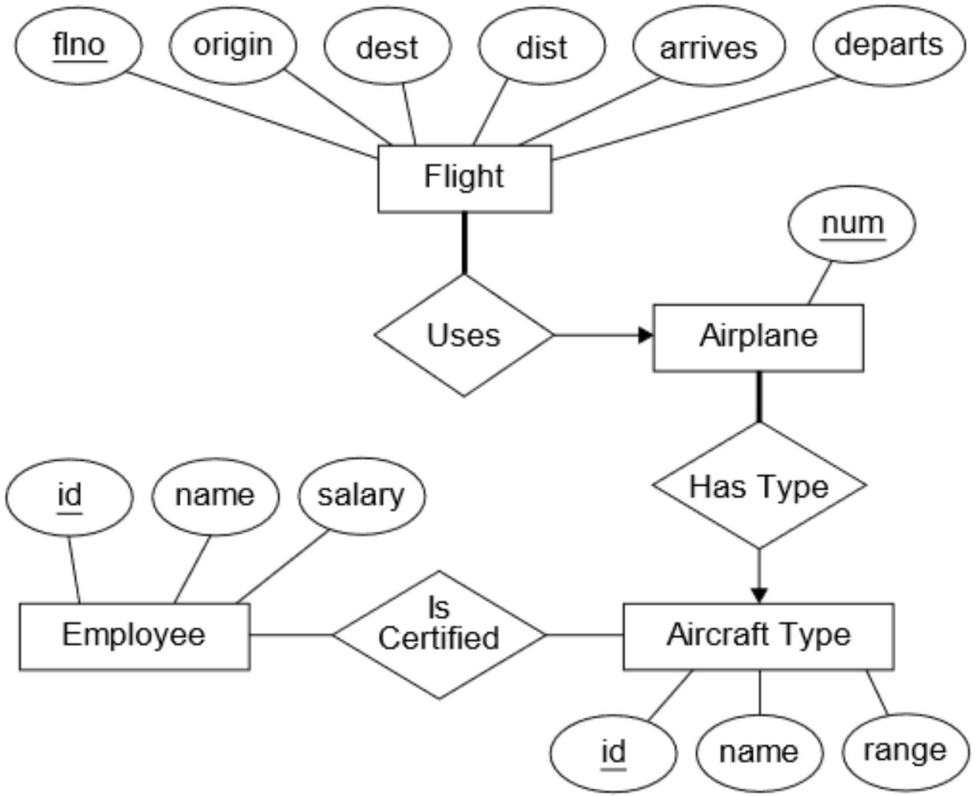
Airplane(<u>num</u>, type_id)



Transform this diagram to a relational schema

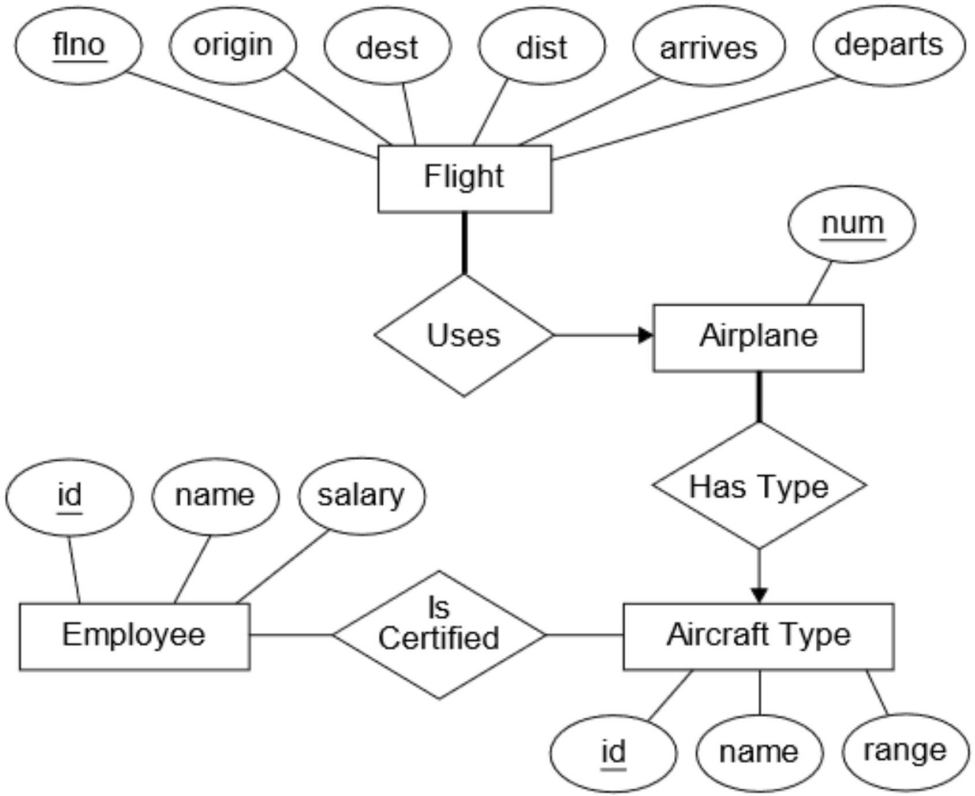


Flight(flno, origin, dest, dist, arrives, departs, airplane_num)



Flight(flno, origin, dest, dist, arrives, departs, airplane_num)

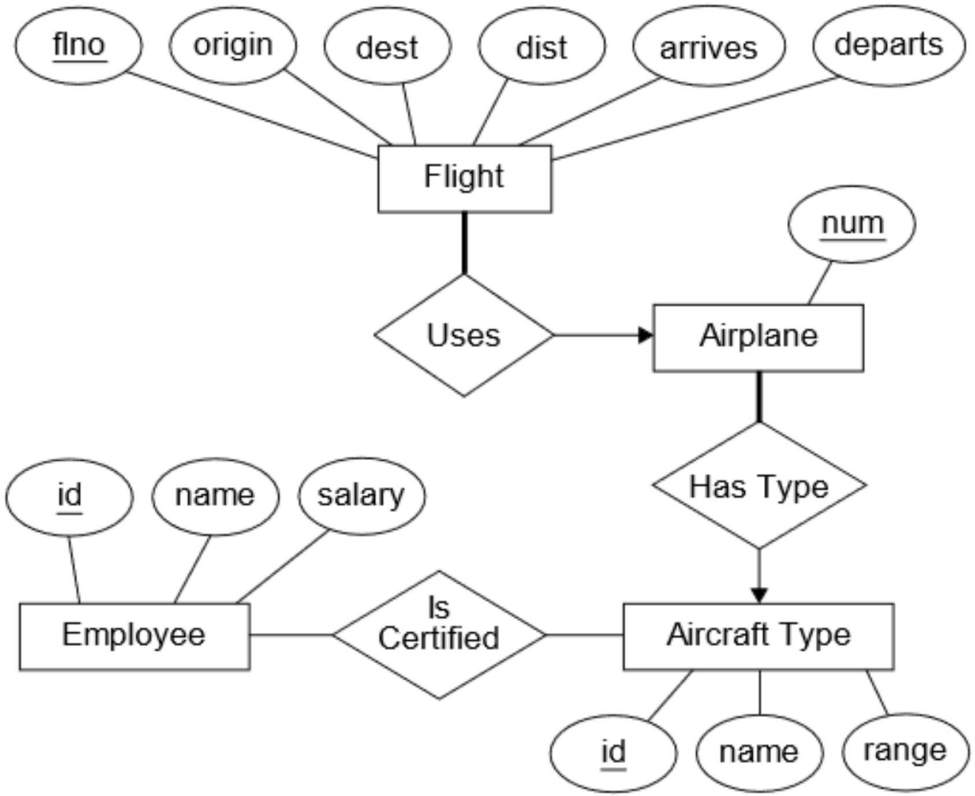
Airplane(num, type_id)



Flight(flno, origin, dest, dist, arrives, departs, airplane_num)

Airplane(num, type_id)

AircraftType(id, name, range)

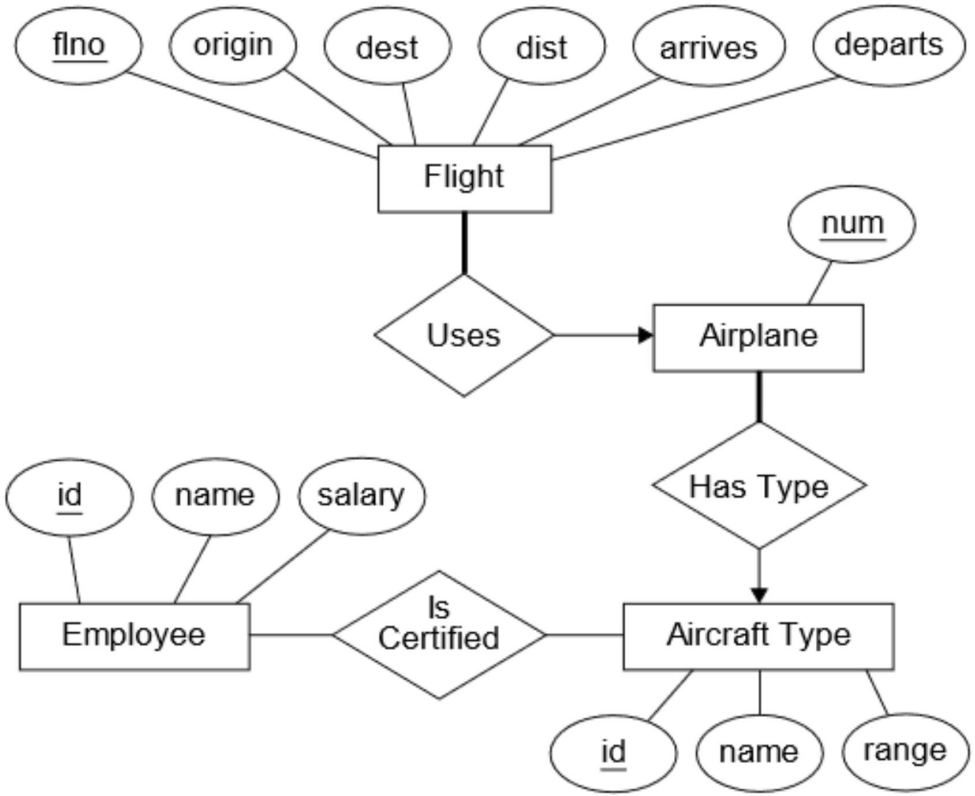


Flight(flno, origin, dest, dist, arrives, departs, airplane_num)

Airplane(num, type_id)

AircraftType(id, name, range)

Employee(id, name, salary)



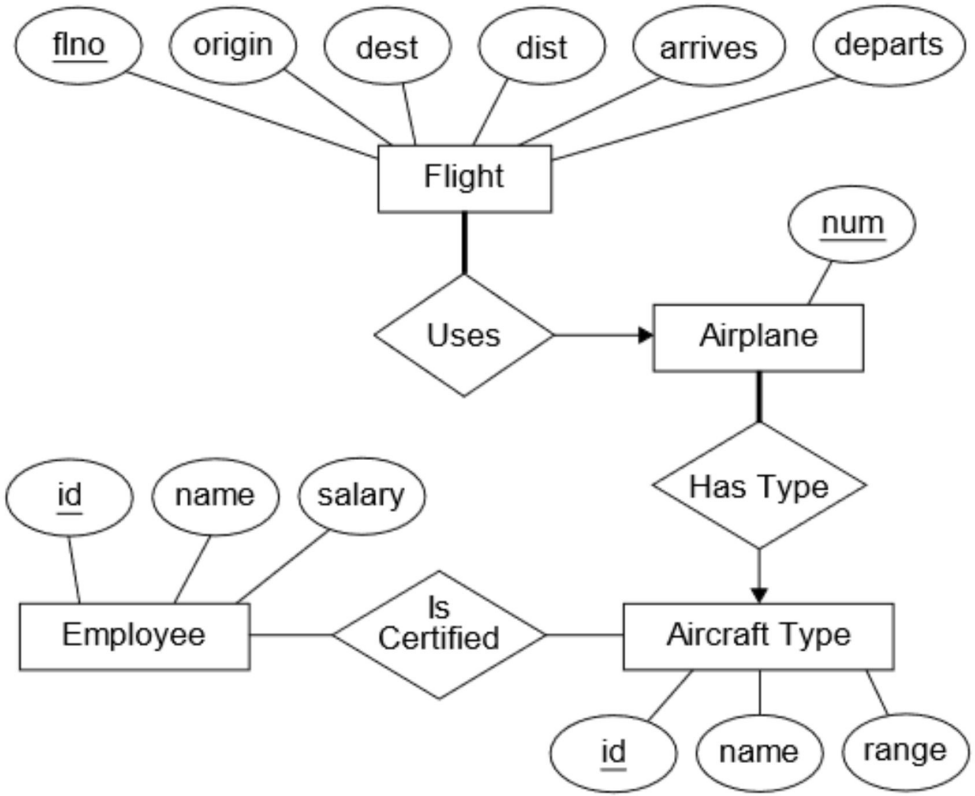
Flight(flno, origin, dest, dist, arrives, departs, airplane_num)

Airplane(num, type_id)

AircraftType(id, name, range)

Employee(id, name, salary)

IsCertified(employee_id, aircraft_type_id)



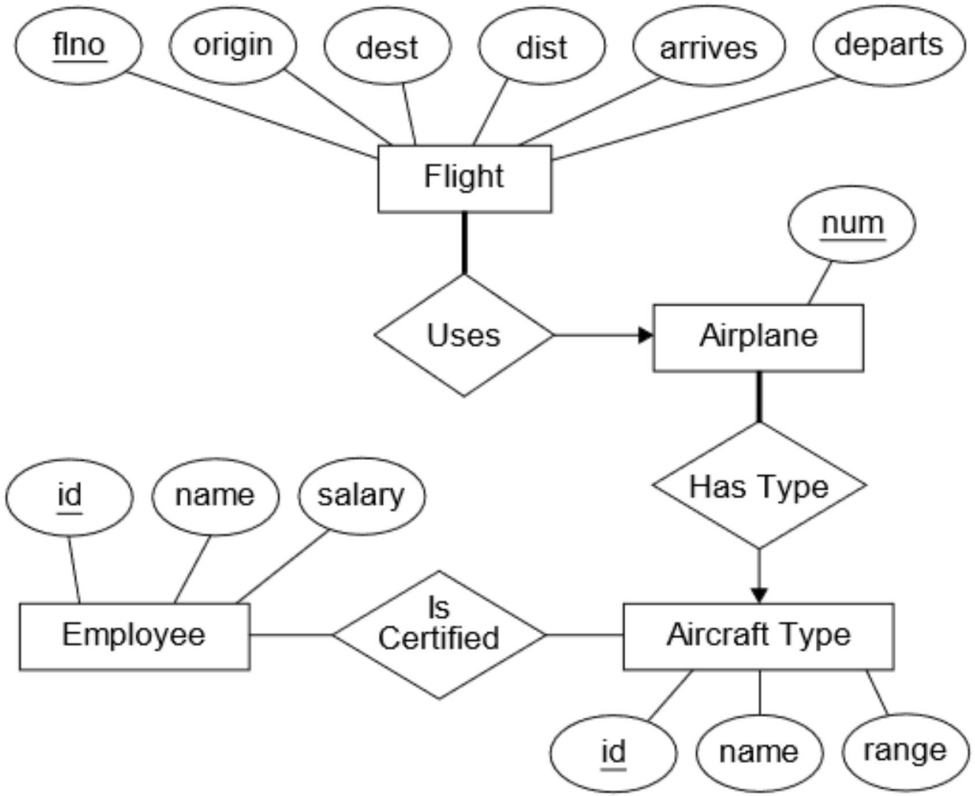
Flight(flno, origin, dest, dist, arrives, departs, airplane_num)

Airplane(num, type_id)

AircraftType(id, name, range)

Employee(id, name, salary)

IsCertified(employee_id, aircraft_type_id)



Foreign Keys

Flight(flno, origin, dest, dist, arrives, departs, airplane_num)

Airplane(num, type_id)

AircraftType(id, name, range)

Employee(id, name, salary)

IsCertified(employee_id, aircraft_type_id)

Schema of the database

Student(*id*, name)
Department(*name*, office)
Room(*id*, name, capacity)
Course(*name*, start_time, end_time, room_id)
MajorsIn(*student_id*, *dept_name*)
Enrolled(*student_id*, *course_name*, credit_status)

1. Let's say that we want to use relational alebra to find the names of all students in the database who are enrolled in CS 460 and the credit status for which they are enrolled.

   Why doesn't the following query work?
   $\pi_{\text{name, credit\_status}} (\sigma_{\text{course\_name = 'CS 460'}} (\text{Student x Enrolled}))$

2. How could we fix the query so that we get only the names of students who are actually enrolled in CS 460?

David is the only student enrolled in CS460

Student

id	name
001	David
002	Monty
003	Alex

Enrolled

Course_name	Student_id
CS460	001
CS599	003

Student x Enrolled

Student.id	Student.name	Couse_name	Student_id
001	David	CS460	001
001	David	CS599	003
002	Monty	CS460	001
002	Monty	CS599	003
003	Alex	CS460	001
003	Alex	CS599	003

$\sigma_{\text{course\_name = 'CS 460'}}$ (Student x Course)

Student.id	Student.name	Couse_name	Student_id
001	David	CS460	001
001	David	CS599	003
002	Monty	CS460	001
002	Monty	CS599	003
003	Alex	CS460	001
003	Alex	CS599	003

Π name ($\sigma_{\text{course\_name = 'CS 460'}}$ (Student x Course)

Student.id	Student.name	Couse_name	Student_id
001	David	CS460	001
002	Monty	CS460	001
003	Alex	CS460	001

π name, credit_status (σ **id = student_id** AND course_name = 'CS 460' (Student x Enrolled))

σ course_name = 'CS 460' (Student x Course)

Student.id	Student.name	Couse_name	Student_id
001	David	CS460	001
001	David	CS599	003
002	Monty	CS460	001
002	Monty	CS599	003
003	Alex	CS460	001
003	Alex	CS599	003

σ id = student_id AND course_name = 'CS 460' (Student x Course)

Student.id	Student.name	Couse_name	Student_id
001	David	CS460	001
001	David	CS599	003
002	Monty	CS460	001
002	Monty	CS599	003
003	Alex	CS460	001
003	Alex	CS599	003

Schema of the database

Student(*id*, name)
Department(*name*, office)
Room(*id*, name, capacity)
Course(*name*, start_time, end_time, room_id)
MajorsIn(*student_id*, *dept_name*)
Enrolled(*student_id*, *course_name*, credit_status)

3. Would the following relational-algebra query allow us to solve the same problem?

$\pi_{\text{name, credit\_status}} (\sigma_{\text{course\_name} = \text{'CS 460'}} (\text{Student} \bowtie \text{Enrolled}))$

Schema of the database

Student(*id*, name)
Department(*name*, office)
Room(*id*, name, capacity)
Course(*name*, start_time, end_time, room_id)
MajorsIn(*student_id*, *dept_name*)
Enrolled(*student_id*, *course_name*, credit_status)

3. Would the following relational-algebra query allow us to solve the same problem?

$\pi_{\text{name, credit\_status}} (\sigma_{\text{course\_name} = \text{'CS 460'}} (\text{Student} \bowtie \text{Enrolled}))$

No, a predicate is not specified in the join.

$\pi_{\text{name, credit\_status}} (\sigma_{\text{course\_name} = \text{'CS 460'}} (\text{Student} \bowtie_{\text{id} = \text{student\_id}} \text{Enrolled}))$

Schema of the database

Student(*id*, name)
Department(*name*, office)
Room(*id*, name, capacity)
Course(*name*, start_time, end_time, room_id)
MajorsIn(*student_id*, *dept_name*)
Enrolled(*student_id*, *course_name*, credit_status)

4. Assume that a given student can have 0 or more majors. If we want to find the *id numbers* of all students who are *not* majoring in computer science, why wouldn't the following query work?
   $\pi_{\text{id}} (\sigma_{\text{id = student\_id AND dept\_name != 'comp sci'}} (\text{Student x MajorsIn}))$

5. How *could* we use relational algebra to obtain the id numbers of all students who are *not* majoring in computer science?

Schema of the database

Student(*id*, name)
Department(*name*, office)
Room(*id*, name, capacity)
Course(*name*, start_time, end_time, room_id)
MajorsIn(*student_id*, *dept_name*)
Enrolled(*student_id*, *course_name*, credit_status)

4. Assume that a given student can have 0 or more majors. If we want to find the *id numbers* of all students who are *not* majoring in computer science, why wouldn't the following query work?
   $\pi_{\text{id}} (\sigma_{\text{id = student\_id AND dept\_name != 'comp sci'}} (\text{Student x MajorsIn}))$

   1. Doesn't include students without a major (entry not in MajorsIn)
   2. Will include students with more than 1 major (CS & Math)

$$\pi_{id}(\text{Student}) - \pi_{student\_id}(\sigma_{dept\_name = \text{'comp sci'}}(\text{MajorsIn}))$$