Exercise 2

Jasper Steinberg

February 6, 2025

1 Constraints and syntax with specialization

1.1 a)

Disjointness means that the subclasses of a given specialization must be disjoint sets, i.e. an entity may only belong to at most one of the subclasses of the specialization. Completeness can be either total or partial. In a total specialization constraint we enforce that each entity in the superclass must be a member of at least one subclass (in the specialization). In a partial specialization we we allow an entity to not belong to any of its subclasses.

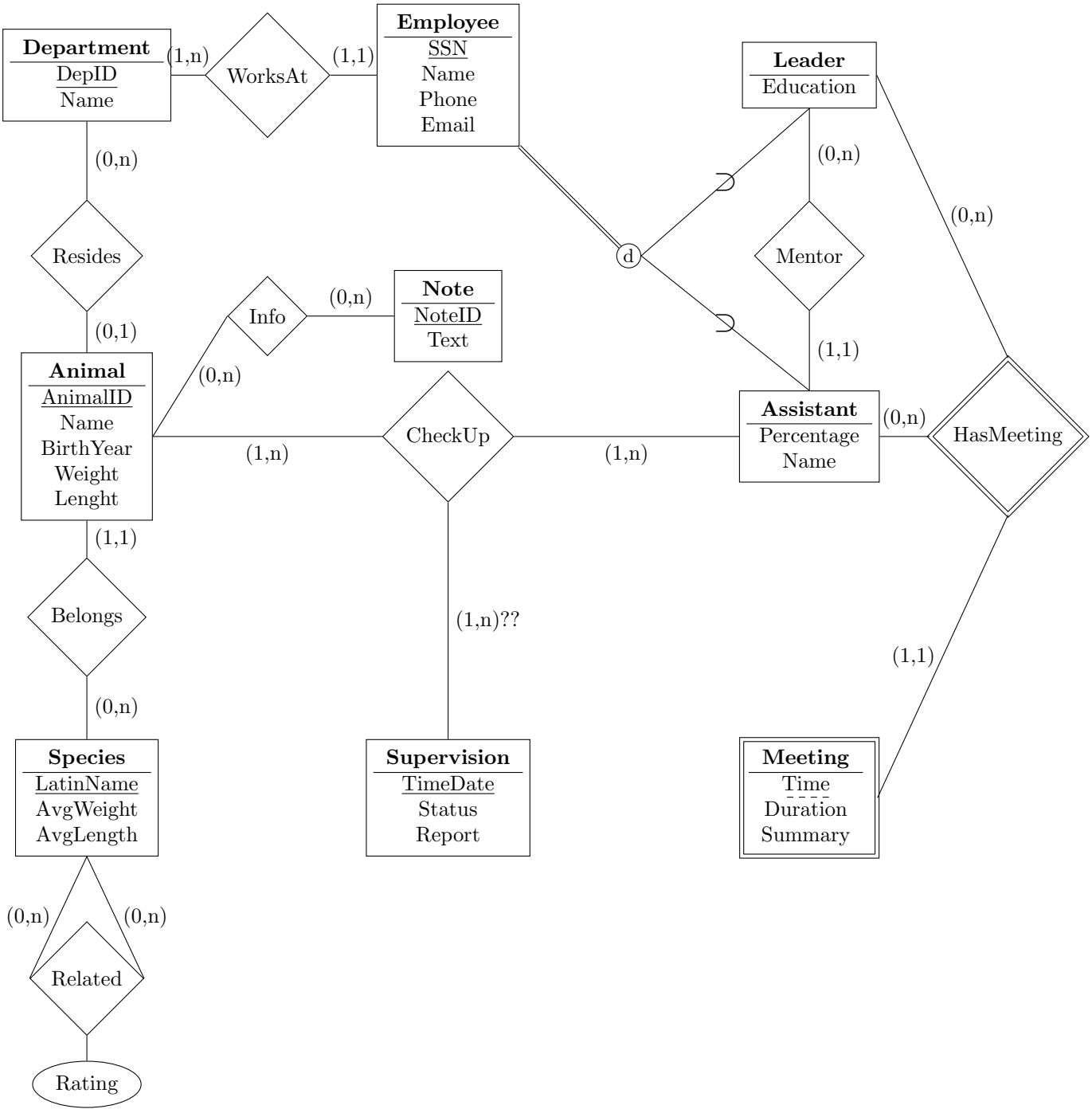
1.2 b)

Restrictions	**Example**
Disjoint, total	Biological sex as a superclass, male and female as a subclass.
Disjoint, partial	Athlete superclass with different, non-exhaustive sports as subclasses.
Overlapping, total	Student superclass with different fields like math and chemistry as subclasses.
Overlapping, partial	Student superclass with extra curricular activities as subclasses.

1.3 c)

Figure	**Validity**	**Explanation**
1	Invalid	A relation to a specialization circle is nonsensical.
2	Valid	I see no problem with the logic of the diagram.
3	Valid	No problems.
4	Invalid	Inclusions towards a specialization with no superclass is nonsensical.

2 ER model of a zoo



3 Important terms in the relational data model

Entity integrity is based on the fact that a relation is a set of tuples. We thus cannot have two tuples with the same values, as sets ignore duplicates. Such ambiguities are avoided by imposing the need for a unique identifier (primary key).

Referential integrity is the same concept just for relations. A foreign key must refer to an existent row in the relation its referencing (or only consist of NULL-values). I.e. the key must point to a row with a valid primary key, or point to nothing.

4 Conversion between models and relational algebra

4.1 a)

Exam(ExamNo, CourseCode, ExaminationAids)

Student(StudentNo, Name)

ExaminationLocation(RoomNo, Name, Capacity)

Table(TableNo, Type, TableAtRoomNo) TableAtRoomNo is foreign key for ExaminationLocation, cannot be NULL

Chair(ChairNo, Type, ChairAtRoomNo) ChairAtRoomNo is foreign key for ExaminationLocation, cannot be NULL

HasExam(StudRegNo, ExamRegNo) StudRegNo is foreign key for Student, cannot be NULL. ExamRegNo is foreign key for Exam, cannot be NULL

SetUp(SetUpStudentNo, SetUpExamNo, SetUpRoomNo, Date, StudentPlacement) SetUpStudNo is foreign key for Student, cannot be NULL. SetUpExamNo is foreign key for Exam, cannot be NULL. SetUpRoomNo is foreign key for ExaminationLocation, cannot be NULL.

4.2 b)

4.2.1 Return the hotel number and name for all hotels.

$$\Pi_{\text{HotelNo,Name}}(\text{Hotel})$$

4.2.2 Return the hotel number and name for all hotels in Barcelona

Let

$$\Sigma = \sigma_{\text{Area="Barcelone"}}(\text{Hotel}),$$

then

$$\Pi_{\text{HotelNo,Name}}(\Sigma)$$

returns the desires information.

4.2.3 Return the room number and hotel name for those hotel rooms larger than 100 square meters

First extract the the rows with the desired room specification from HotelRoom

$$\Gamma = \sigma_{\text{Area}>100}(\text{HotelRoom}).$$

Then join by matching hotels with their hotel rooms (kunne ha natural-joina),

$$\kappa = \text{Hotel} \underset{\text{Hotel.HotelNo=HotelRoom.HotelNo}}{\bowtie} \Gamma.$$

Then project with the desired information

$$\Sigma = \Pi_{\text{RoomNo,Name}}(\kappa).$$

4.2.4 Count the orders of hotel rooms smaller than 8 square meters and with a duration longer than 7 days

First natural join HotelRoom and Order,

$$\Gamma = \text{HotelRoom*Order}.$$

Then extract rows with small enough rooms and long enough duration

$$\kappa = \sigma_{\text{SquareMeterSize<8 \& Duration>7}}(\Gamma).$$

Now return the value of a count(*) operation on κ.

4.2.5 Return the full name of those customers with an order in Madrid

First we restrict to Madrid,

$$\Gamma = \sigma_{\text{Area="Madrid"}}(\text{Hotel}).$$

Now natural join Hotel with HotelRoom,

$$\Gamma^{'} = \text{Hotel*HotelRoom}.$$

Now natural join with Order,

$$\overline{\Gamma} = \Gamma^{'}\text{*Order}.$$

Now natural merge with Customer

$$\underline{\Gamma} = \overline{\Gamma}\text{*Customer}.$$

We extract the names of the customers

$$\Phi = \Pi_{\text{FirstName,LastName}}(\underline{\Gamma})$$

which is the answer.

4.2.6 Find the duration of all orders made by customers with the name "Ole Hansen". Sort the result on duration in an ascending order

We restrict Customer to the name given, and natural merge with order

$$\gamma = \text{Order*}\sigma_{\text{FirstName=Ole,LastName=Hansen}}(\text{Customer}).$$

Now we extract the durations

$$\zeta = \Pi_{\text{Order}}(\gamma).$$

Finally run a sort operator with the ASC order specification.