Exam Revision Exercises

SWEN 304

Trimester 2, 2017

Lecturer: Dr Hui Ma

Engineering and Computer Science

1. Query Processing and Optimisation

- Suppose the following relational database schema is used by *InterCity Coachlines* to manage information on their bus transportation network.
- *InterCity Coachlines* runs multiple bus lines. Each bus line is served by one or more services. On each bus line, the services are identifiable by their numbers (*ServiceNo*), but services on different bus lines may have the same service number.

- S = {

  Stop ({StopName, City, Location}, {StopName}),

  Line ({LineId, LineName}, {LineId}),

  Service ({LineId, ServiceNo, From, To}, {LineId, ServiceNo}),

  Destination ({LineId, ServiceNo, StopName, ArrTime, DepTime},

  {LineId, ServiceNo, StopName })

  }

1. Query Processing and Optimisation

- I = {
  Service [LineId] $\subseteq$ Line [LineId],
  Service [From] $\subseteq$ Stop [StopName],
  Service [To] $\subseteq$ Stop [StopName],
  Destination [(LineId, ServiceNo)] $\subseteq$ Service [(LineId, ServiceNo)],
  Destination [StopName] $\subseteq$ Stop [StopName]
  }

  Not Null Constraints = {
  Null (Service, From) = Not,
  Null (Service, To) = Not,
  }

1. Query Processing and Optimisation

- Suppose you want to know the arrival times of services on the GreatSights line that go from Auckland Central to Wellington Parliament. To answer this question you can issue the following SQL query against the database:

  SELECT ArrTime

  FROM Line NATURAL JOIN Service NATURAL JOIN Destination

  WHERE LineName = 'Great Sights' AND From = 'Auckland Central' AND To = 'Wellington Parliament' AND StopName = 'Tongariro';

1. Transfer the above SQL query in relational algebra.
2. Draw a query tree for your relational algebra query in part **a)**
3. Optimise the query tree in part **b)** using heuristic rules for algebraic query optimisation as discussed in our lectures. Explain which heuristic rules you can apply. Draw the optimised query tree.

2. Entity Relationship Model

- Suppose Wellington's most popular DVD rental shop (Rent'n'Watch) hires you as a data architect to design a new DVD rental database for them. They provide you with the following list of requirements:
  - For customers, it is required to store their customer number, first name, last name, and address.
  - For staff members, it is required to store their staff id, first name, last name, and contact phone number.
  - Rent'n'Watch rents out DVDs containing movies. Each DVD contains exactly one movie. For a movie there may be many DVD copies.
  - For each movie, it is required to store its title, production year, and length.
  - Each DVD has its unique DVD number and some features.
  - When a customer hires a DVD from a certain staff member, the hiring date and due date will be recorded.

2. Entity Relationship Model

- Draw an EER diagram of the new database for *Rent'n'Watch.* Show the entity types and relationship types, but skip the attributes on the diagram. Show structural constraints explicitly on the diagram.

  Give attributes separately in the form:

  Entity_Type_Name {$Attribute_1$,..., $Attribute_n$},

  with entity type keys underlined, or

  Relationship_Type_Name ($Entity\_Type_1$,..., $Entity\_Type_n$, {$Attribute_1$,..., $Attribute_m$}),

  where the set of a relationship type attributes can be empty.

Note: Use the notation and conventions for drawing ER diagrams that we introduced in the lectures. If you prefer to use another notation that differs from the one we used in lectures, clearly define all differences.

3. Normalization

- Consider the following set of attributes used for capturing the data about entries into Photography Competitions:

- *Entry* = { *EntrantName, EntrantEmail, Competition, NumberOfImages, Fees*}
- with the following set of functional dependencies:
- *F* = {

$$EntrantEmail \rightarrow EntrantName$$

$$Competition+NumberOfImages \rightarrow Fees$$

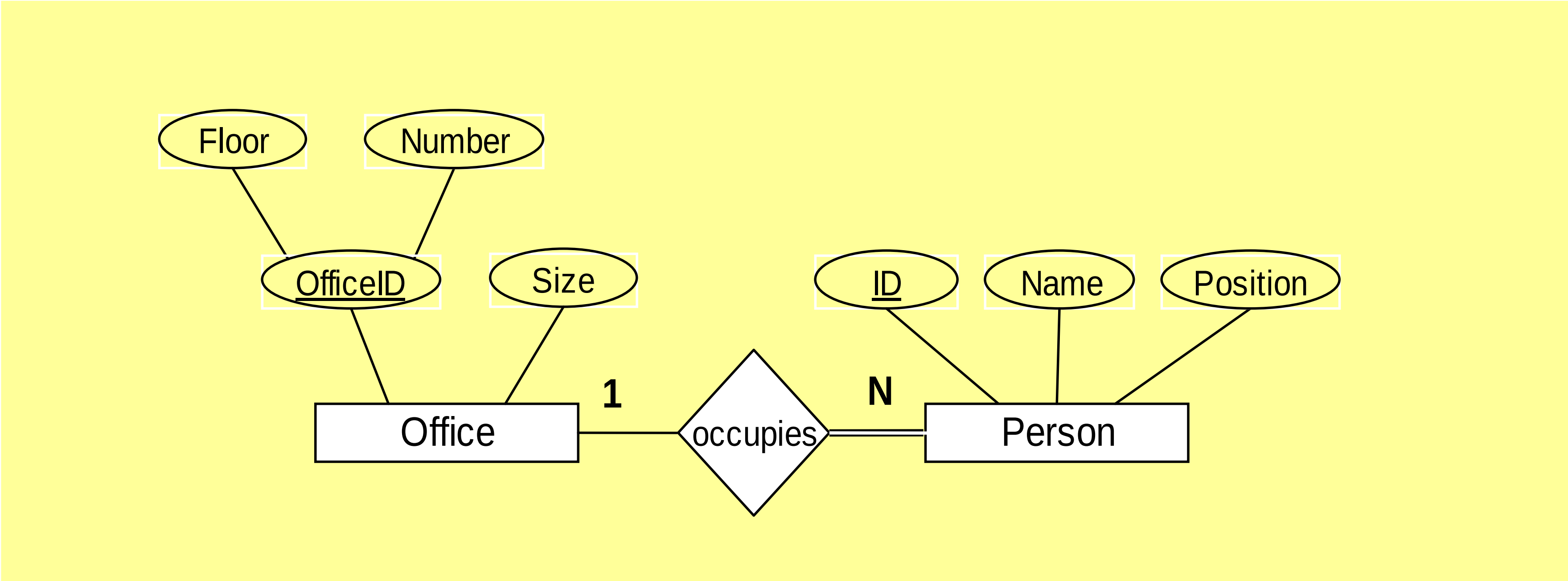
$$EntrantEmail+Competition \rightarrow NumberOfImages$$

}

3. Normalization

1. Determine all minimal keys of *Entry* with respect to *F*. Justify your answer.
2. Explain when a relation schema is in Third Normal Form (3NF) with respect to a set of functional dependencies.
3. Decide whether *Entry* is in Third Normal Form (3NF) with respect to *F*. Justify your answer. If not, determine a lossless Third Normal Form decomposition.
4. Explain when a relation schema is in Boyce-Codd Normal Form (BCNF) with respect to a set of functional dependencies.
5. Decide whether *Entry* in Boyce-Codd Normal Form (BCNF) with respect to *F*. Justify your answer. If not, determine a lossless Boyce-Codd Normal Form decomposition.

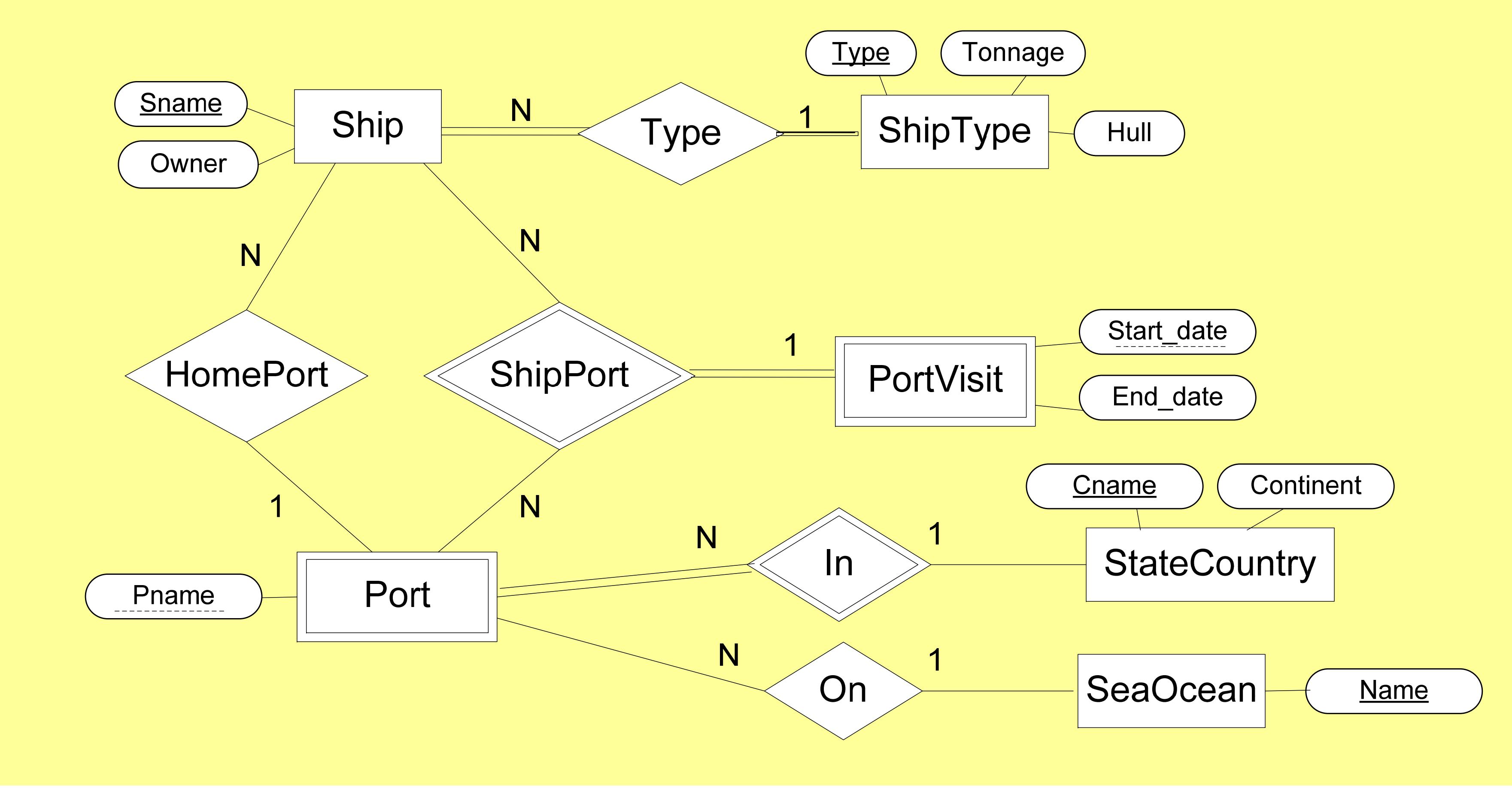
4. Transform from ER to RDM: Example 1



Office({Floor, Number, Size},{Floor+Number})
Person({ID, Name, Position, Floor, Number},{ID})

Person.Floor and Person.Number must be Not Null.
Person[(Floor, Number)] $\subseteq$Office[(Floor, Number)]

4. Transform from ER to RDM: Example 2



4. Transform from ER to RDM: Example 2

S = {
ShipType({Type, Tonnage, Hull }, {Type })
StateCountry({Cname, Continent}, {Cname})
Port({Cname, Pname, Name}, {Cname + Pname })
Ship ({Sname, Owner, Type, Cname, Pname }, {Sname})
PortVisit ({Sname, Cname, Pname, Start_date, End_date},
{Sname + Pname + Cname + Start_Date, })
}

I = {
Ship [Type] $\subseteq$ ShipType [Type]
Ship [Cname, Pname] $\subseteq$ Port [Cname, Pname]
Port[Cname] $\subseteq$ StateCountry [Cname]
PortVisit [Sname] $\subseteq$ Ship [Sname]
PortVisit [Cname, Pname] $\subseteq$ Port [Cname, Pname]
}

Null (Ship, Type) = N