CM1603 - Database Systems

Week 05| Introduction to Relational Model

Dileeka Alwis – Lecturer / Level Coordinator,

Department of Computing, IIT

Learning Outcomes

- Covers LO1 for Module - Describe and evaluate underlying theory and principles of relational database management systems (RDBMS).
- Covers LO2 for Module – Analyses and apply database design and modelling methods for a given business case study
- On completion of this lecture, students are expected to be able to:
  - Rules of relational schema
  - Convert ER diagram in to relational schema.
  - Convert E-ER diagram in to relational schema.

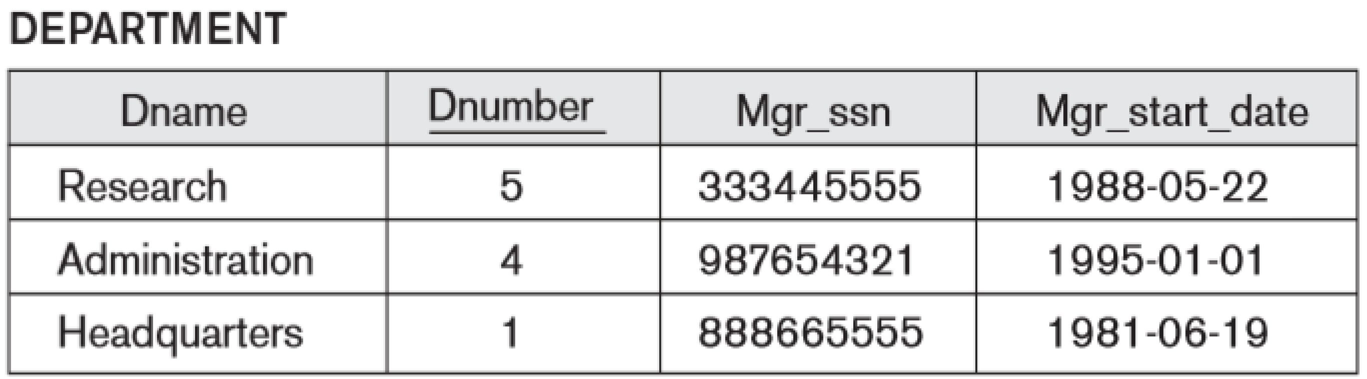
Lesson Outline

- Introduction to Relational Model
- Relational Model Terminologies
- Relational Database Schema
- Essential properties of relations
- Relational Database Keys
- Convert ER Model to Relational Model

Relational Model

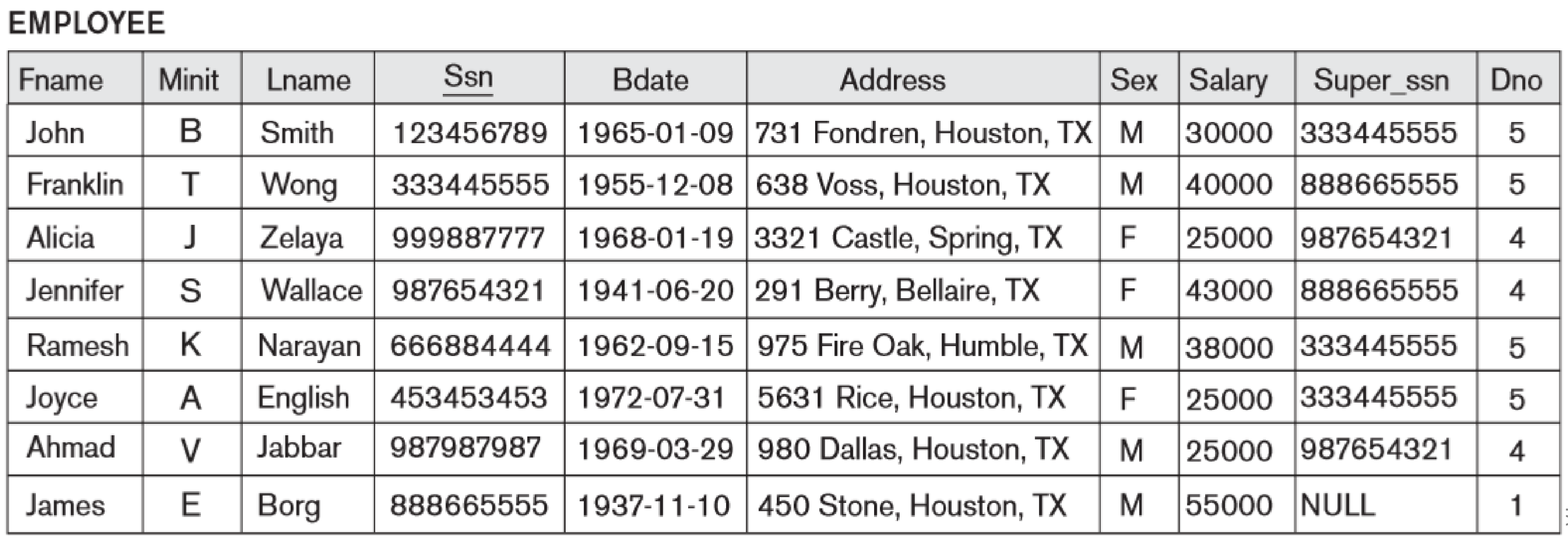
- The most widely used Data Model.
- Data is organized as a collection of **two-dimensional tables (relations)** which are interconnected via relationships.
- Uses PKs and FKs to maintain the relationships and follow a set of integrity constraints.
- Uses high level Structured Query Language (SQL) for CRUD operations.
- Also called as SQL Model.

Example for Relational Model



DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

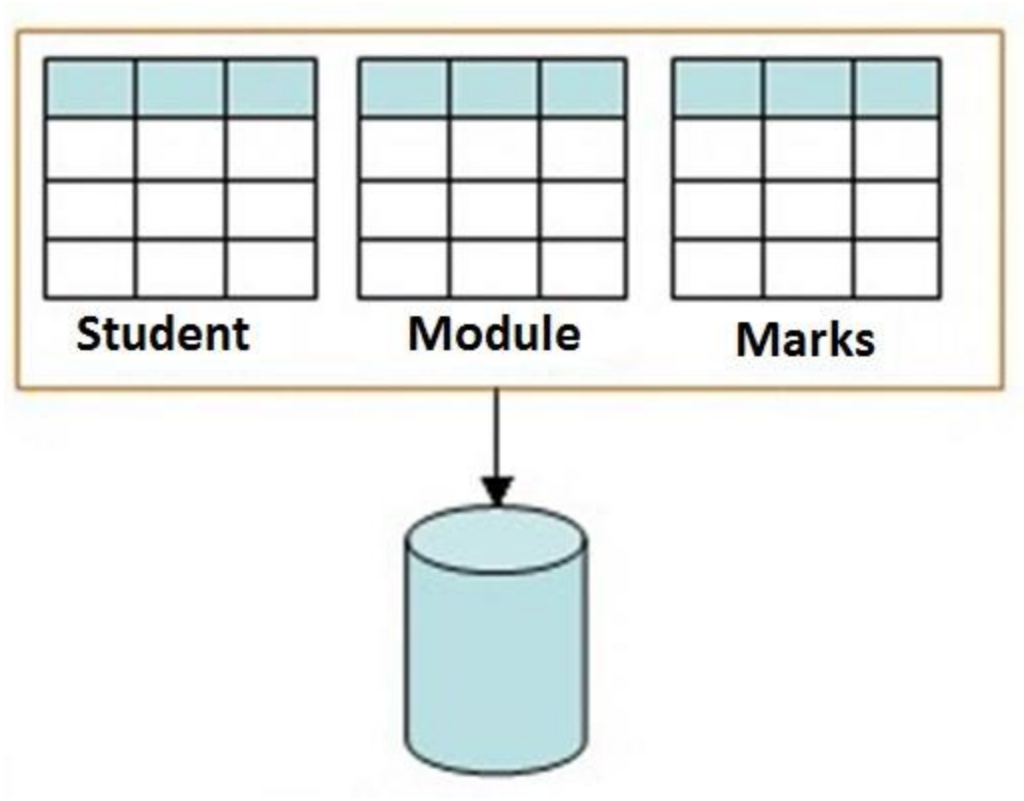


EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Relational Database Terminologies

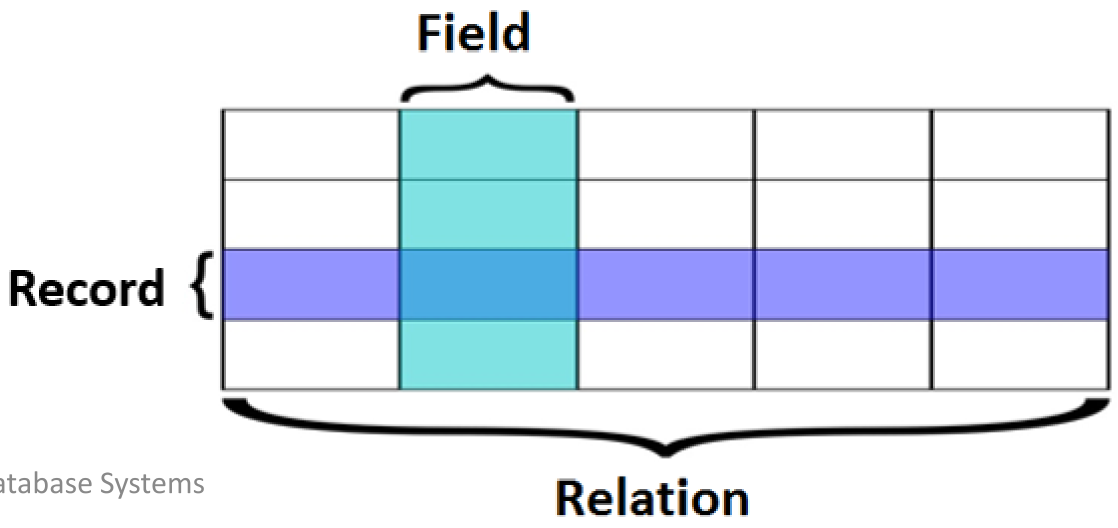
- **Relation**
  - A two-dimensional table which contains related data about a particular entity.



Relational Database Terminologies

- **Field / Attribute**
  - A column in a relation which is a characteristic of that entity.
  - Eg: StudentID, Name, DOB etc. are fields in Student relation.

- **Record / Tuple**
  - A row in a relation which contains information about different instances of an entity.



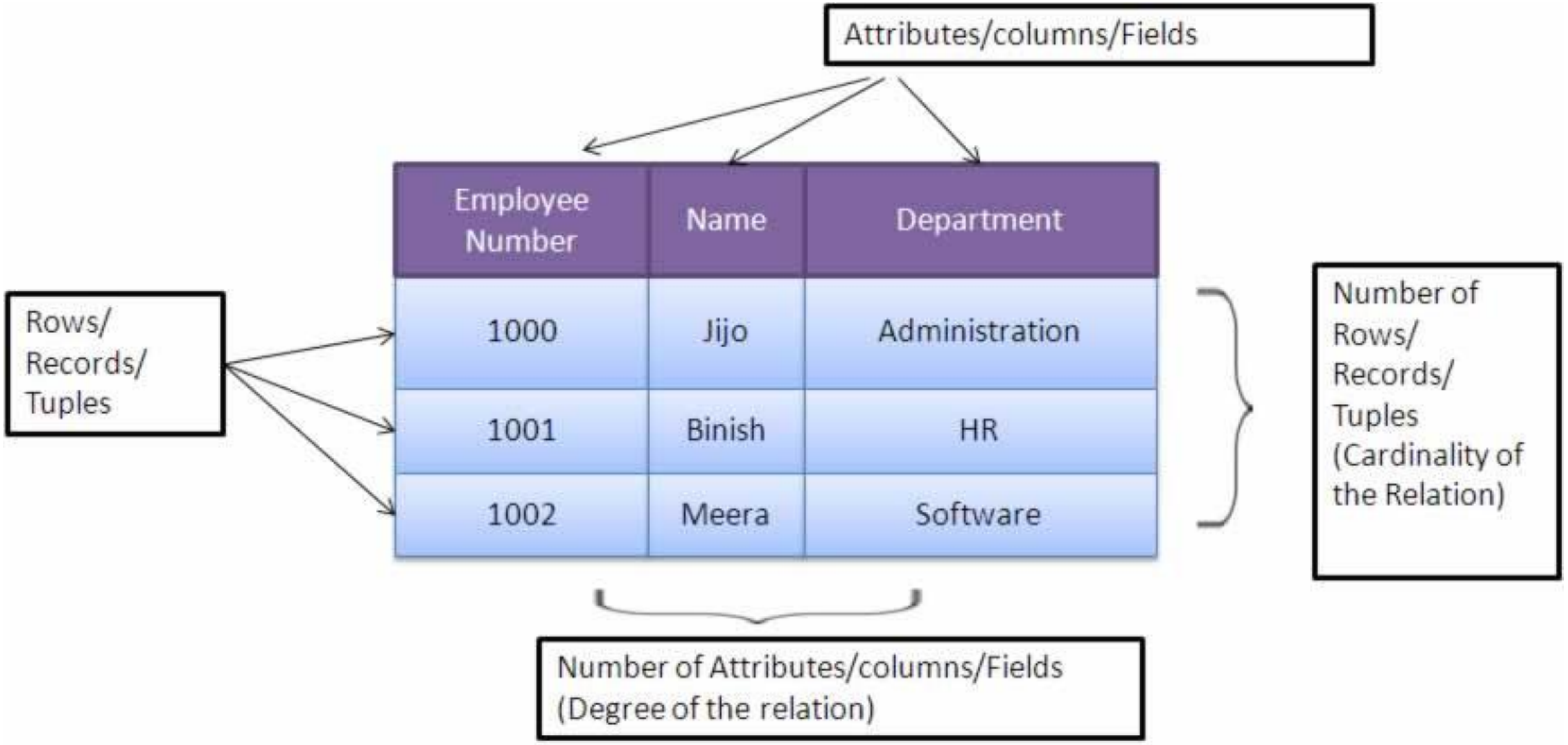
Relational Database Terminologies

- **Attribute Domain**
  - The set of acceptable values for an attribute.
  - A particular entity occurrence / instance will have a value for each of its attributes.

- **Degree of a Relation**
  - Number of attributes in a relation.

- **Cardinality of a Relation**
  - Number of tuples in a relation

Example



Employee Number	Name	Department
1000	Jijo	Administration
1001	Binish	HR
1002	Meera	Software

Attributes/columns/Fields
Rows/ Records/ Tuples
Number of Rows/ Records/ Tuples (Cardinality of the Relation)
Number of Attributes/columns/Fields (Degree of the relation)

employee_details table

Relational Database Schema

- **Table Schema**
  - A named relation defined by a set of field names.

Employee = (EmployeeNumber, Name, Department)

- **Relational Database Schema**
  - Set of table schemas, each with a distinct name which includes all the tables required to capture all the data in the database.

Essential properties of relations

- **Relation**
  - A relation name is distinct from all other relation names in a relational schema.
  - Each cell of relation contains exactly one atomic (single) value.
- **Attribute**
  - Each attribute has a distinct name.
  - Values of an attribute are from the same domain.
  - Order of attributes has no significance.
- **Tuple**
  - Each tuple is distinct; no duplicate tuples.
  - Order of tuples has no significance.

Relational Database Keys

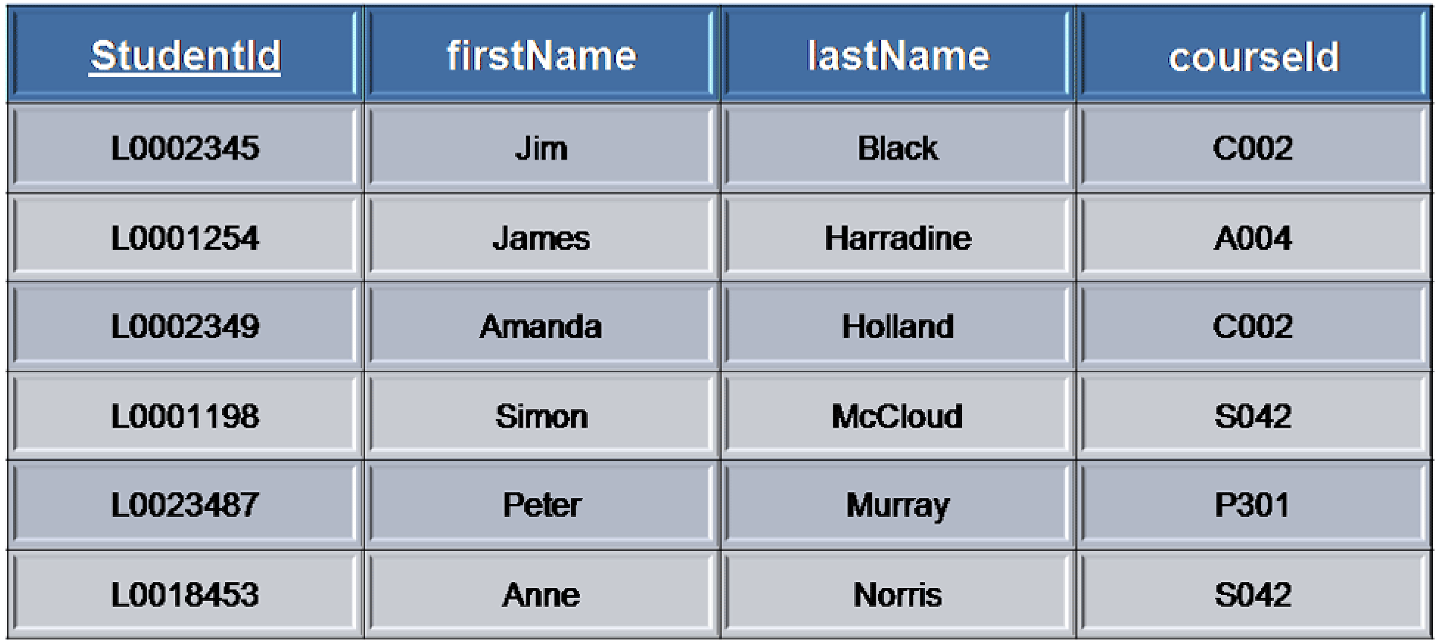
- Database key is an attribute or set of attributes which helps to identify a tuple in a relation.

- It allows to find the relationship between two relations.

- There are several Database keys:
    - **Super Key**
    - **Candidate Key**
    - **Primary Key**
    - **Foreign Key**

Super Key

- Any combination of attributes within a relation that uniquely identifies each tuple within a relation.
- The set of **all** the attributes of a relation is always a super key.



StudentId	firstName	lastName	courseId
L0002345	Jim	Black	C002
L0001254	James	Harradine	A004
L0002349	Amanda	Holland	C002
L0001198	Simon	McCloud	S042
L0023487	Peter	Murray	P301
L0018453	Anne	Norris	S042

Candidate Key

- An irreducible (cannot separate) set of attributes which uniquely identifies a tuple in a relation.
- Even if one or more attributes are excluded, the remaining attributes still uniquely identifies a tuple.
- Every table must have at least one candidate key.
- A subset of a super key.
- Each candidate key is a super key, but each super key is not a candidate key.

Primary Key (PK)

- The candidate key with minimum set of fields which can uniquely identify each record in a relation.
- A table must have **one** primary key.
- Values must be **unique and not null** for each record.

PK

StudentID	Name	Gender
1	Ann	F
2	Peter	M
3	Tom	M
4	Rose	F
5	Peter	M

Foreign Key (FK)

- A field in one relation which refers to a field in another relation.
- Always refers to a primary key of the other table.

PK FK PK

Student_ID	Name	Course
1	Ann	C1
2	Peter	C2
3	Tom	C1
4	Rose	C3
5	Peter	C2

CCode	Course Name
C1	Software Engineering
C2	Computer Science
C3	Computer Networks

Composite PK

- PK can be a combination of two or more fields when there is no unique field in the particular relation.
- The combination of two or more fields will help to identify a record uniquely.

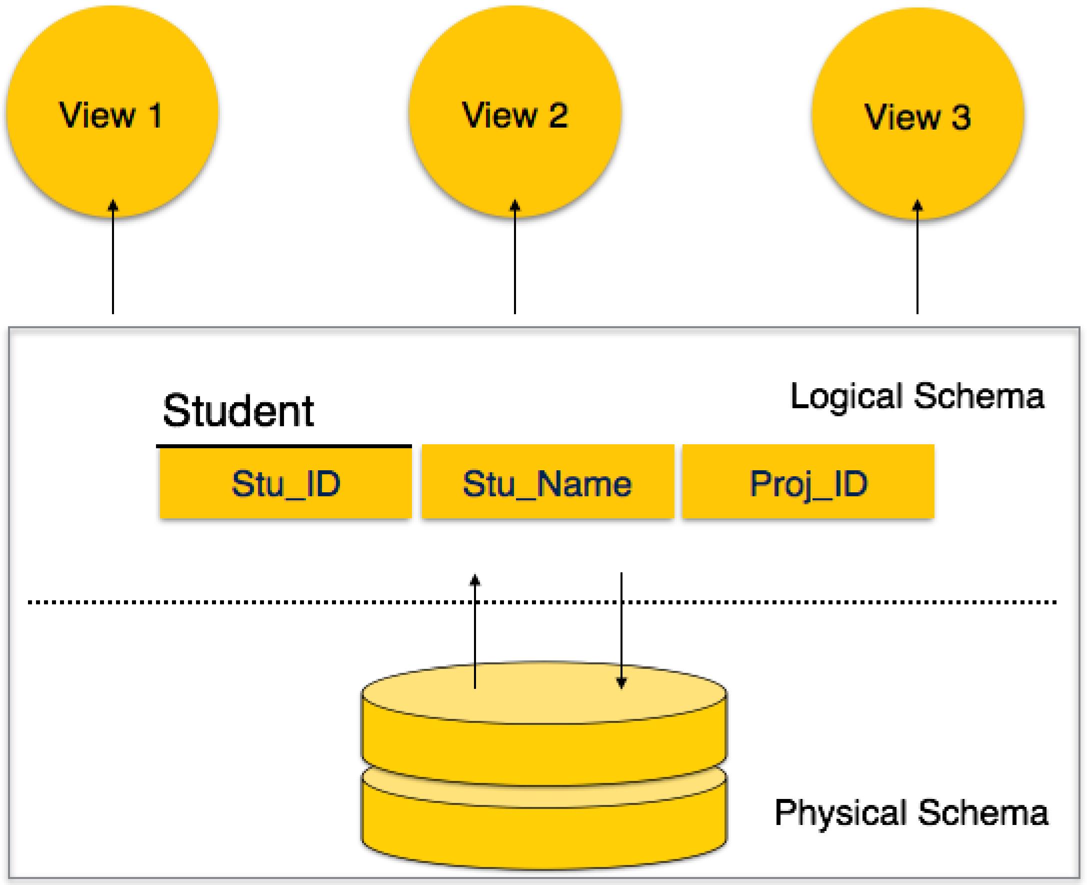
PK

Student_ID	Module	Marks
1	CS105.3	54
1	SE104.3	61
2	CS105.3	69
2	SE104.3	48
3	CS105.3	28
4	CS105.3	34

Logical Schema

- A database schema is the skeleton structure that represents the logical view of the entire database.

- It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.



Convert ER Model to Relational Model

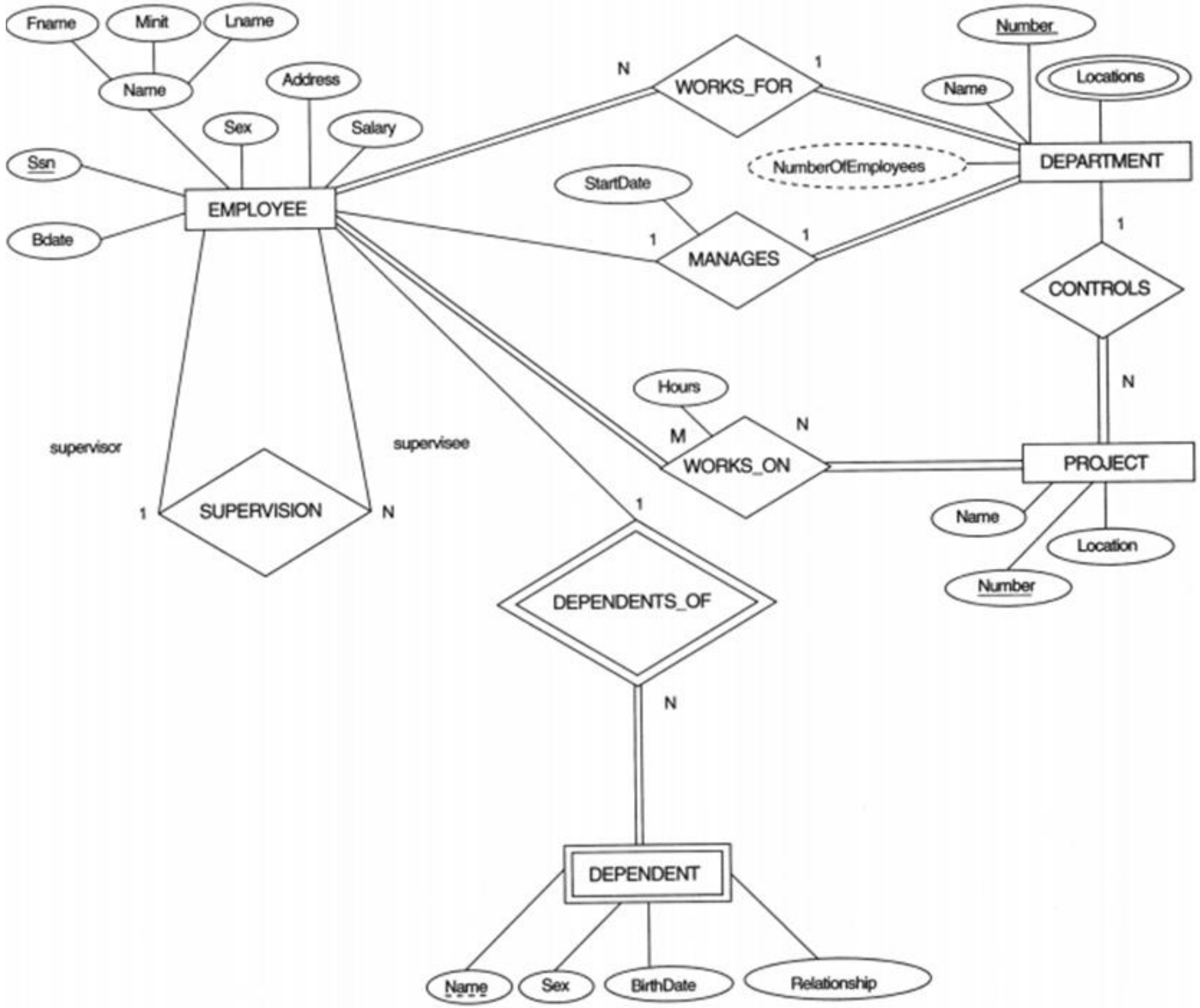
- ERD is converted into tables in Relational Model as follows:
  - Mapping entities (Strong entity, Weak entity)
  - Mapping attributes (Simple attributes, Composite attributes, Derived attributes, Multivalued attributes)
  - Mapping relationships (1:1, 1:m, m:n, Self- Referencing)

Company DB Scenario

A company database keeps track of a company's employees, departments, and projects.

The company is organized into departments. Each department has a unique number, a unique name, number of employees and a particular employee who manages the department. It keeps track of the start date when that employee began managing the department. A department may have several locations. A department controls a number of projects, each of which has a unique number, a unique name, and a single location.

It store each employee's name, social security number, address, salary, gender, and birth date. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. It keeps track of the current number of hours that an employee works on each project. It also keeps track of the direct supervisor of each employee who is another employee. It wants to keep track of the dependents of each employee for insurance purposes. It keeps each dependent's first name, sex, birth of date, and relationship to the employee.



Fname
Minit
Lname
Name
Address
Sex
Salary
Ssn
Bdate
EMPLOYEE
N
WORKS_FOR
1
Number
Name
Locations
DEPARTMENT
NumberOfEmployees
StartDate
1
MANAGES
1
1
CONTROLS
N
Hours
M
WORKS_ON
N
PROJECT
Name
Location
Number
supervisor
supervisee
1
SUPERVISION
N
1
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
BirthDate
Relationship

Mapping Strong Entities

- Create a relation for each strong (regular) entity type in the ER diagram.
- Include all the simple attributes of the entity.
- Underline the chosen key attribute as the primary key.
- Any composite attributes must be represented as simple attributes.
- Omit multi valued and derived attributes.

Eg: Department (<u>DeptNo</u>, DeptName)

Mapping of Weak Entity Types

- Create a new relation for each weak entity.
- Include all the attributes of the weak entity.
- Add the primary key of the strong entity (owner entity) as a foreign key.
- The primary of the weak entity will be a combination of the foreign key (primary key of owner entity) and the partial key of the weak entity (Composite PK).

Eg: Dependent (<u>SSN, DependantName</u>, Sex, BirthDate, Relationship)

Mapping of Multivalued Attributes

- Create a new relation for each multivalued attribute.
- Include the primary key as a foreign key in the new relation multi valued attribute.
- The primary key of the new relation is the combination of the foreign key and the multi valued attribute (Composite PK).
- If the multivalued attribute is composite, include its simple attributes.

Eg: Dept_Location (<u>DeptNo, Location</u>)

Mapping Composite Attributes

- Any composite attributes must be represented with their simple attributes.

Employee (Ssn, Fname, Minit, Lname, Bdate, Sex, Address, Salary)

Mapping Derived Attributes

- Do not represent in relations.
- Those values can be taken (derived) with the support of another attribute.
- Helps to reduce the capacity of the DB.

Mapping of Binary 1:1 Relationships

- If one side is with total participation
  - Select the total participation side relation.
  - Add the PK of the other side relation as FK to the selected relation.
  - Include all the simple attributes of the 1:1 relationship type to the selected relation.

Eg: Department (<u>DeptNo</u>, DeptName, ManagerSSN, StartDate)

Mapping of Binary 1:1 Relationships

- If both sides are with partial participation
  - Can select any side and follow the above steps.

- If both sides are with total participation
  - Must merge the 2 entities into one relation with all attributes of both entities.
  - Choose one PK from the two original PKs. Make the other one the AK.

Mapping of Binary 1:M Relationships

- Select the MANY side of the relation.
- Include the primary key of the ONE side relation as the foreign key in the selected relation.
- Include any simple attributes of the 1:M relation type as attributes of the selected relation.

Eg: Project (<u>ProjNo</u>, ProjName, Location, DeptNo)

Mapping of Binary M:N Relationships

- Create a new relation.
- Include the primary keys of both relations as foreign keys in the new relation.
- The combination of both foreign keys will form the primary key of the new relation (Composite PK).
- Include any simple attributes of the M:N relationship type.
- A partial key taken from the new relation can also be joined with two foreign keys to create the composite primary key if needed.

Eg: Works_On (<u>EmpNo, ProjNo</u>, Hours)

Mapping of N-ary Relationship Types

- For each n-ary relationship (where n>2), create a new relation.
- Include the primary keys of the relations that represent the participating entity types as foreign keys.
- Include any simple attributes of the n-ary relationship type.
- The combination of all foreign keys will form the primary key of the new relation (Composite PK).

Mapping Recursive Relationships

- Check the cardinality of the relationship and apply the correct mapping rule accordingly.

- For a m:n relationship
  - Create a new relation and apply m:n mapping rule.

- For 1:1 or 1:m relationships
  - Add the primary key to the same relation again with a different name.

Eg: Employee (<u>SSN</u>, Fname, Minit, LName, Sex, Address, Salary, Super_SSN)

Thank you

Contact: dileeka.a@iit.ac.lk