VIT | Vidyalankar Institute of Technology
Accredited A+ by NAAC

Relational Model and Relational Algebra

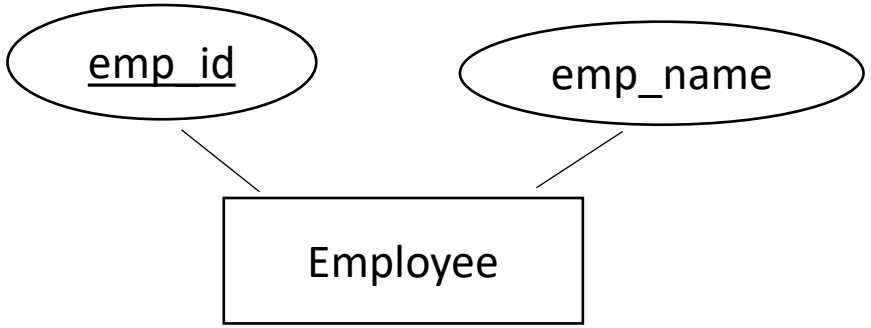
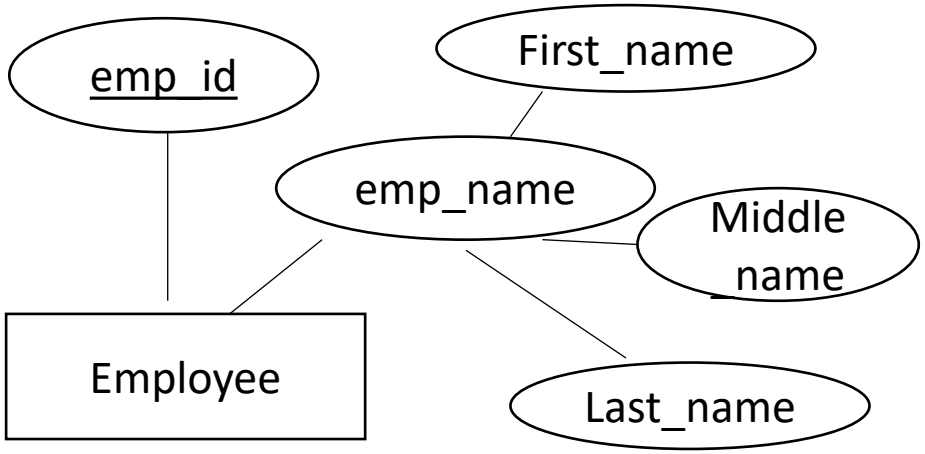
Prof. Suja Jayachandran

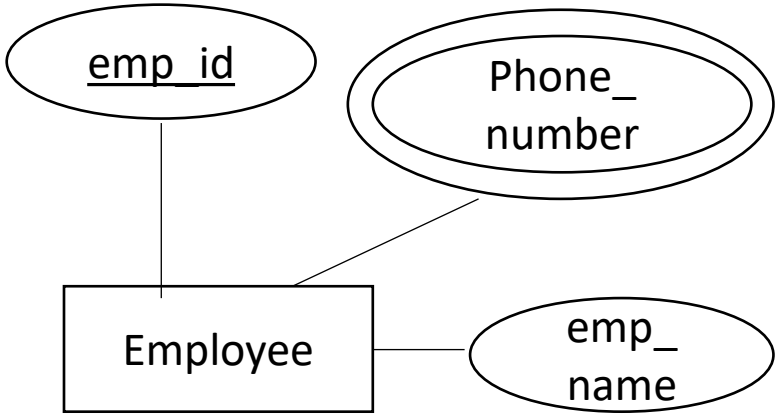
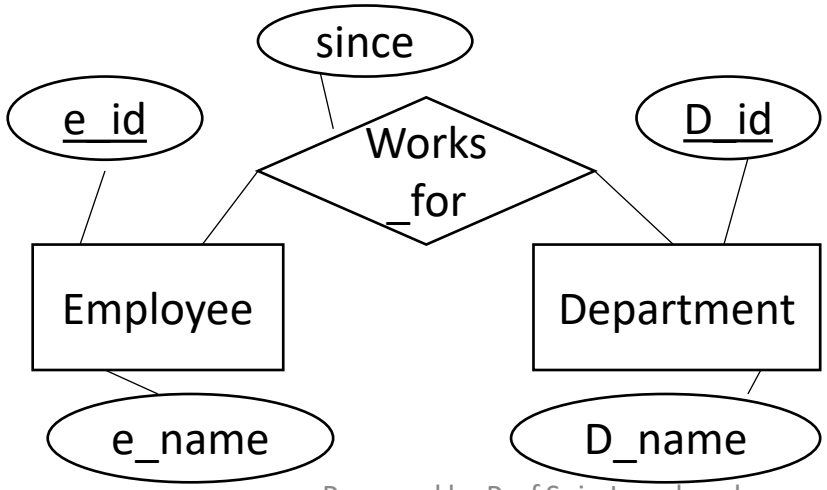
Assistant Professor
Computer Engineering Department
Vidyalankar Institute of Technology

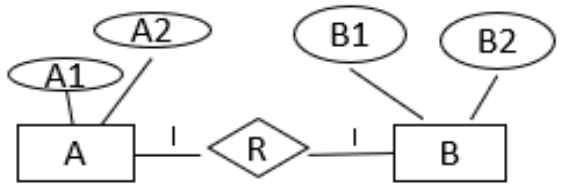
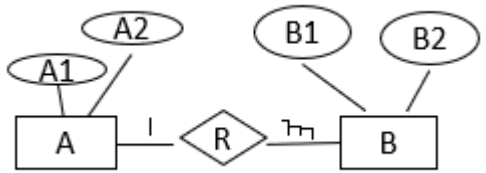
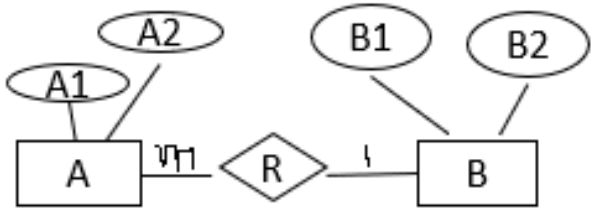
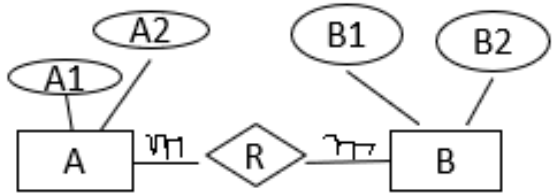
- Relational Model was proposed by E.F. Codd to model data in the form of relations or tables.
- Relational Model represents how data is stored in Relational Databases. A relational database stores data in the form of relations (tables).

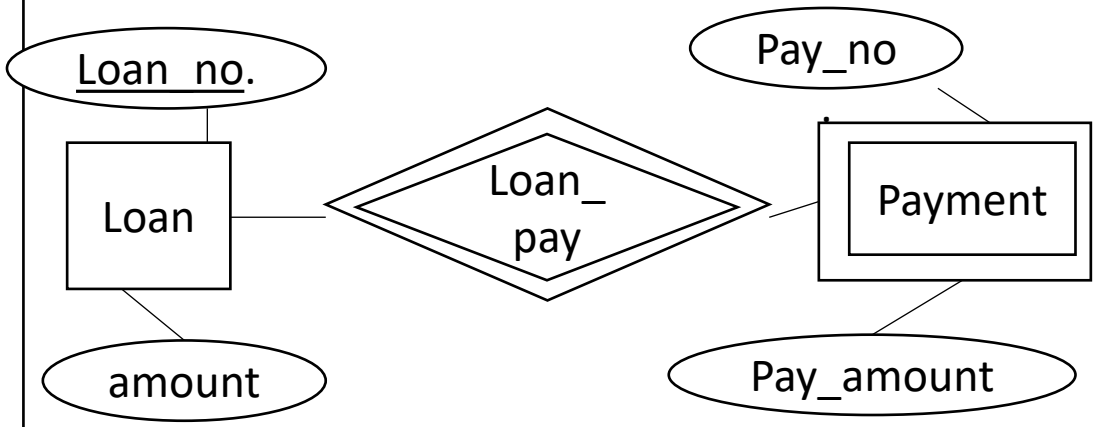
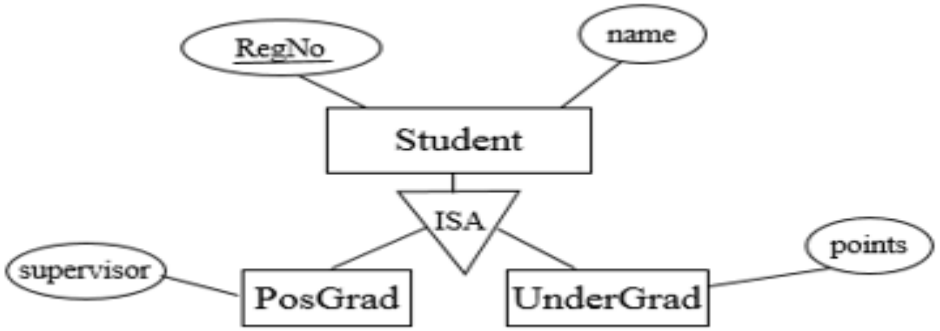
ER Model	Relational Model
ER model is the high level or conceptual model.	It is the representational or implementation model.
It is used by people who don't know how database is implemented.	It is used by programmers.
It represents collection of entities and describes relationship between them.	It represent data in the form of tables and describes relationship between them.
It consists of components like Entity, Entity Type, Entity Set.	It consists of components like domain, attributes, tuples.
It is easy to understand the relationship between entities.	It is less easy to derive the relationship between different tables.

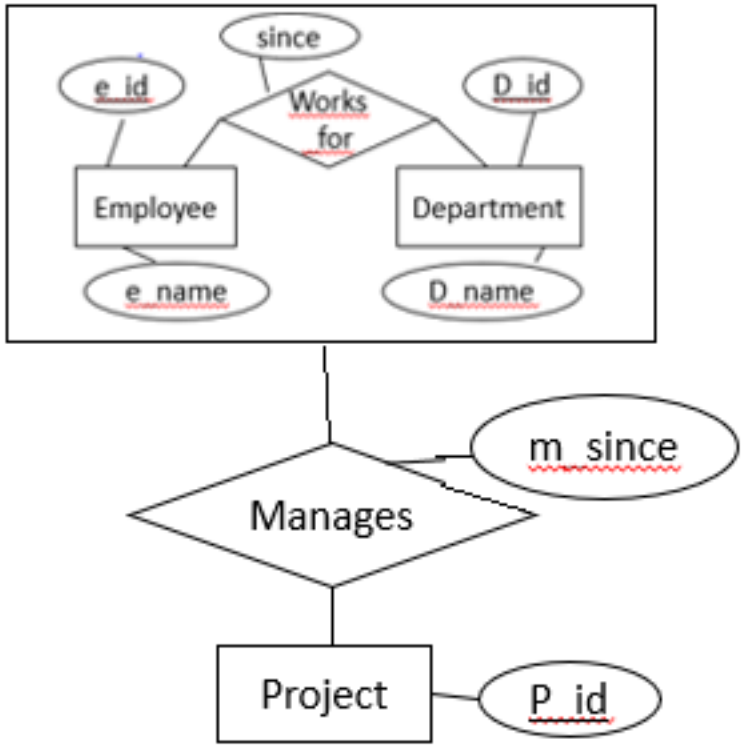
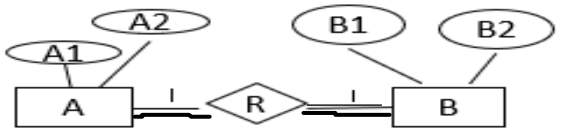
- Mapping the ER and EER Model to the Relational Model

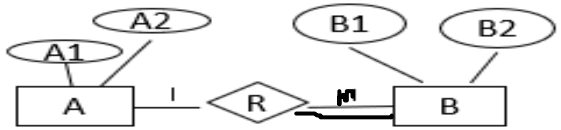
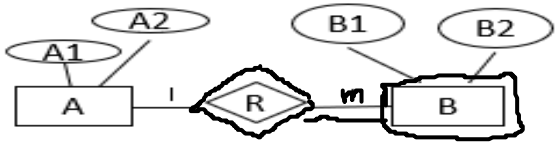
<table>
<tr><th>Sr.No.</th><th>Description</th><th>ER Model</th><th>Relational Model</th></tr>
<tr><td>1</td><td>Entity Set Employee with emp_id as primary key and emp_name simple attribute</td><td> 
</td><td>Employee (Table/Relation Name)
<table><tr><td>emp_id</td><td>emp_name</td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table></td></tr>
<tr><td>2</td><td>Entity Set Employee with emp_id attribute as primary key and emp_name as composite attribute</td><td> 
</td><td>Employee (Table/Relation Name)
<table><tr><td>emp_id</td><td>First_ name</td><td>Middle_ name</td><td>Last_ name</td></tr><tr><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td></tr></table></td></tr>
</table>

<table>
<tr><th>Sr.No.</th><th>Description</th><th>ER Model</th><th>Relational Model</th></tr>
<tr><td>3</td><td>Entity set Employee with emp_id as primary key ,emp_name and phone number as multivalued attribute</td><td> 
</td><td>Employee
<table><tr><td>emp_id</td><td>emp_name</td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table>
<table><tr><td>emp_id</td><td>Phone_number</td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table></td></tr>
<tr><td>4</td><td>Translating a relationship set with descriptive attribute into table
If no descriptive attribute then based on cardinality create the table.</td><td> 
</td><td>Employee
<table><tr><td>e_id</td><td>e_name</td></tr><tr><td></td><td></td></tr></table>
Department
<table><tr><td>D_id</td><td>D_name</td></tr><tr><td></td><td></td></tr></table>
Works_for
<table><tr><td>e_id</td><td>D_id</td><td>since</td></tr><tr><td></td><td></td><td></td></tr></table></td></tr>
</table>

Sr.No.	Description	ER Model	Relational Model
5	**One to one** cardinality relationship A(A1,A2) B(B1,B2)		Relations/Tables: A(A1,A2) and B(B1,B2,A1) OR A(A1,A2,B1) and B(B1,B2)
	One to Many A(A1,A2) B(B1,B2)		Relations/Tables: A(A1,A2) and B(B1,B2,A1)
	Many to One A(A1,A2) B(B1,B2)		Relations/Tables: A(A1,A2,B1) and B(B1,B2)
	Many to Many A(A1,A2) B(B1,B2)		Relations/Tables: A(A1,A2) B (B1,B2) R(A1,B1)

Sr.No.	Description	ER/EER Model	Relational Model
6	To represent **Weak Entity Set**		Relations/Tables Loan(Loan_no.,amount) Payment(Pay_no.,Pay_ amount,Loan_no.)
7	To represent **Generalization /Specialization**		Relations/Tables Student(RegNo., name) PosGrad(RegNo. , supervisor) UnderGrad(RegNo.,points)

Sr.No.	Description	ER Model	Relational Model
8	To represent **Aggregation**		Relations/Tables Employee(e_id,e_name) Department(D_id,D_name) Works_for(e_id,D_id,since) Project(P_id) Manages(e_id,D_id,P_id,m_since)
9	Total PArticipation		A(A1,A2),B(B1,B2)R(A1,A2,B1,B2)

Sr.No.	Description	ER Model	Relational Model
9	Partial PArticipation		A(A1,A2),B(B1,B2,A1)
9	Partial Participation by Weak entity		A(A1,A2) B(B1,B2,A1)

Relational Algebra

- Relational algebra is a procedural query language, which takes instances of relations as input and yields instances of relations as output. It uses operators to perform queries.
- **Unary Relational Operations**

➢ SELECT (symbol: σ)

➢ PROJECT (symbol: π)

➢ RENAME (symbol: ρ)

- **Relational Algebra Operations From Set Theory**

➢ UNION (symbol: U)

➢ INTERSECTION (symbol: ∩),

➢ DIFFERENCE (symbol: -)

➢ CARTESIAN PRODUCT (symbol: x)

- **Binary Relational Operations**

➢ JOIN(symbol: ⋈)

➢ DIVISION(symbol: ÷)

<table>
<tr><th colspan="3">Unary Relational Operations</th></tr>
<tr><td>1</td><td>SELECT (symbol: σ)
Selection operator only selects the required tuples according to the selection condition.
Selection operator always selects the entire tuple. It can not select a section or part of a tuple
Allows duplicate
It is commutative</td><td>Select tuples from a relation “Books” where subject is “database” and price is “450”

σsubject = “database” ∧ price = “450” (Books)</td></tr>
<tr><td>2</td><td>PROJECT (symbol: π)
Projection operator automatically removes all the duplicates while projecting the output relation.So, cardinality of the original relation and output relation may or may not be same.
It is not commutative</td><td>Student Table
<table><tr><th>ID</th><th>Name</th><th>Subject</th><th>Age</th></tr><tr><td>100</td><td>Ajay</td><td>DBMS</td><td>18</td></tr><tr><td>200</td><td>Raj</td><td>DWM</td><td>20</td></tr></table>
πName, Age(Student)
<table><tr><th>Name</th><th>Age</th></tr><tr><td>Ajay</td><td>18</td></tr><tr><td>Raj</td><td>20</td></tr></table></td></tr>
<tr><td>3</td><td>RENAME (symbol: ρ)
Rename operator is used to give another name to a relation</td><td>To create a relation STUDENT_NAMES with ID and Name from STUDENT, it can be done using rename operator as:
ρ(STUDENT_NAMES, ∏(ID, Name)(STUDENT))</td></tr>
</table>

Relational Algebra Operations From Set Theory

1 UNION (symbol: U)

Let R and S be two relations.
Then-

- R ∪ S is the set of all tuples belonging to either R or S or both.
- In R ∪ S, duplicates are automatically removed.
- It is both commutative and associative

R

ID	Name	Subject
100	Ajay	SPCC
300	Raj	SE

S

ID	Name	Subject
100	Ajay	SPCC
200	Vijay	DWM

RUS

ID	Name	Subject
100	Ajay	SPCC
300	Raj	SE
200	Vijay	DWM

2 INTERSECTION(symbol:∩)

Let R and S be two relations.
Then-

- R ∩ S is the set of all tuples belonging to both R and S.
- In R ∩ S, duplicates are automatically removed.
- It is both commutative and associative

R

ID	Name	Subject
100	Ajay	SPCC
300	Raj	SE

S

ID	Name	Subject
100	Ajay	SPCC
200	Vijay	DWM

R∩S

ID	Name	Subject
100	Ajay	SPCC

Relational Algebra Operations From Set Theory

4 **DIFFERENCE (symbol: -)**

Let R and S be two relations.

Then-

- R – S is the set of all tuples belonging to R and not to S.
- In R – S, duplicates are automatically removed.
- Difference operation is associative but not commutative.

R

ID	Name	Subject
100	Ajay	SPCC
300	Raj	SE
200	Vijay	DWM

S

ID	Name	Subject
200	Vijay	DWM

R-S

ID	Name	Subject
100	Ajay	SPCC
300	Raj	SE

5 **CARTESIAN PRODUCT (symbol: X)**

The **Cartesian Product** is also an operator which works on two sets. It is sometimes called the **CROSS PRODUCT** or **CROSS JOIN**. It combines the tuples of one relation with all the tuples of the other relation.

R X S

R

A	1
B	2
D	3
F	4
E	5

S

A	1
C	2
D	3
E	4

R CROSS S

A	1	A	1
A	1	C	2
A	1	D	3
A	1	E	4
B	2	A	1
B	2	C	2
B	2	D	3
B	2	E	4
D	3	A	1
D	3	C	2
D	3	D	3
D	3	E	4

F	4	A	1
F	4	C	2
F	4	D	3
F	4	E	4
E	5	A	1
E	5	C	2
E	5	D	3
E	5	E	4

Binary Relational Operations

1	**JOIN(symbol: ⋈) -** Join operation is essentially a cartesian product followed by a selection criterion.
1.1	**INNER JOIN:**In an inner join, only those tuples that satisfy the matching criteria are included, while the rest are excluded.
1.1.1	**Theta Join:** The general case of JOIN operation is called a Theta join. It is denoted by symbol **θ** Example-A $\bowtie_{\theta}$ B
1.1.2	**EQUI join:** When a theta join uses only equivalence condition, it becomes a equi join.

A

ID	Name	Salary
1	Ajay	2000
2	Vijay	3000

B

ID	Name	Salary
3	Ram	4000
4	Ramesh	2000

A ⋈ A.Salary > B.salary (B)

ID	Name	Salary
2	Vijay	3000

A

ID	Name	Salary
1	Ajay	2000
2	Vijay	3000

B

ID	Name	Salary
3	Ram	4000
4	Ramesh	2000

A ⋈ A.Salary =B.salary (B)

ID	Name	Salary
1	Ajay	2000
4	Ramesh	2000

1.1.3

NATURAL JOIN (⋈)

Natural join can only be performed if there is a common attribute (column) between the relations. The name and type of the attribute must be same.

A

Number	Square
2	4
3	9

B

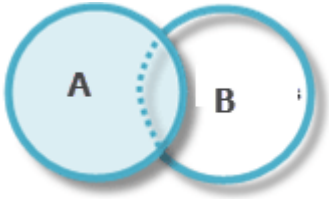
Number	Cube
2	8
3	27

A⋈B

Number	Square	Cube
2	4	8
3	9	27

1.2 **Outer Join**: In an outer join, along with tuples that satisfy the matching criteria, we also include some or all tuples that do not match the criteria.

1.2.1 **Left Outer Join(A⟕B):**In the left outer join, operation allows keeping all tuple in the left relation. However, if there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values.



A

Number	Square
2	4
3	9
4	16

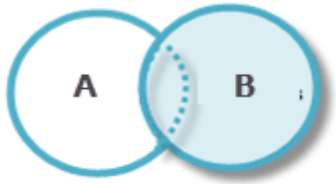
B

Number	Cube
2	8
3	27
5	125

A⟕B

Number	Square	Cube
2	4	8
3	9	1
4	16	Null

1.2.2 **Right Outer Join: (A ⟖ B)-** It allows keeping all tuple in the right relation. However, if there is no matching tuple is found in the left relation, then the attributes of the left relation in the join result are filled with null values.



A

Number	Square
2	4
3	9
4	16

B

Number	Cube
2	8
3	27
5	125

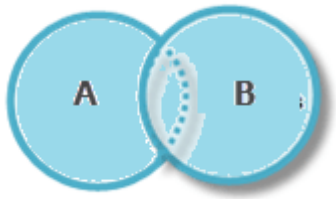
A ⟖ B

Number	Cube	Square
2	8	4
3	27	9
5	125	Null

1.2.3

Full Outer Join: (A ⟗B)

In a full outer join, all tuples from both relations are included in the result, irrespective of the matching condition.



A

Number	Square
2	4
3	9
4	16

B

Number	Cube
2	8
3	27
5	125

A ⟗B

Number	Square	Cube
2	4	8
3	9	27
4	16	Null
5	Null	125

2

DIVISION(symbol: ÷)

Division operator A÷B can be applied if and only if:

Attributes of B is proper subset of Attributes of A.

The relation returned by division operator will have attributes = (All attributes of A – All Attributes of B)

Which employees work on **all** the critical projects?

Works(enum,pnum) Critical(pnum)

Works

enum	pnum
E35	P10
E45	P15
E35	P12
E52	P15
E52	P17
E45	P10
E35	P15

Critical

pnum
P15
P10

Works ÷ Critical

enum
E45
E35

(Works ÷ Critical) × Critical

enum	pnum
E45	P15
E45	P10
E35	P15
E35	P10