

Information Management & Retrieval



Chamali Ranasinghe
21.1

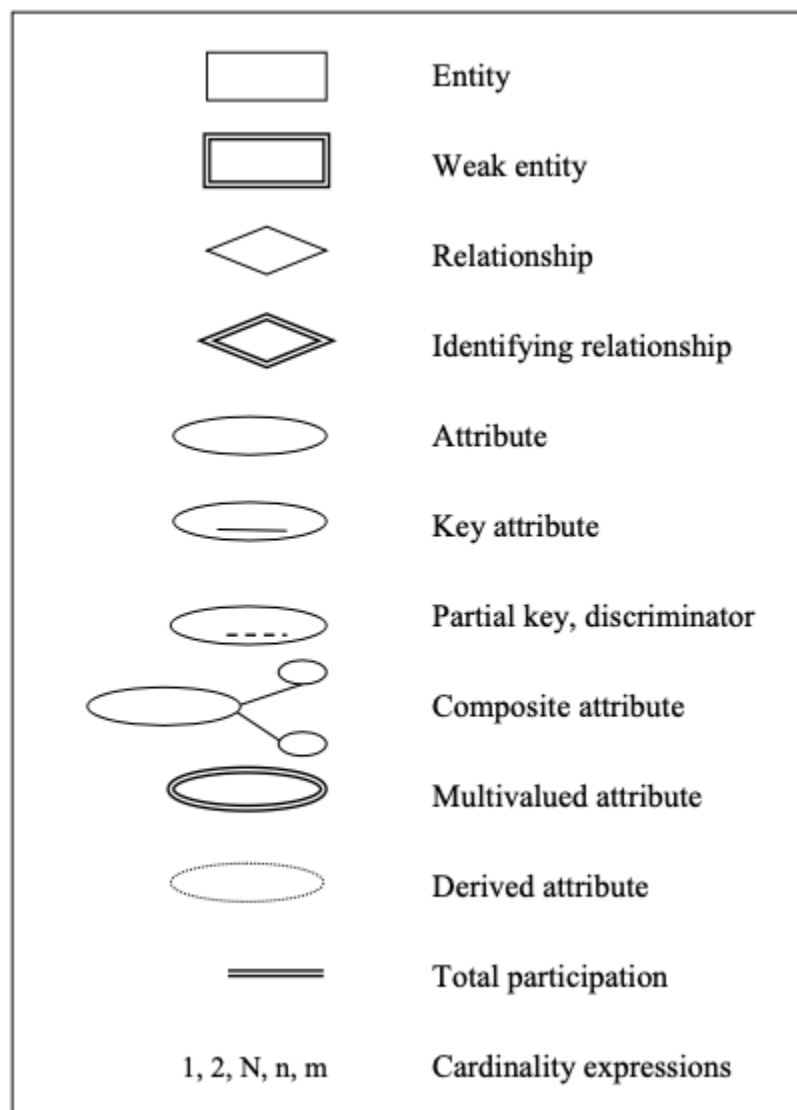
Revision

1. *What is the difference between data and information?*
Data is unorganized facts. You can make information by organizing data. Data can be access, process and manipulate.
2. *What is a database?*
A very large collection of logically related data that is organized to manipulate easily.
3. *Why do we need a database?*
To store data
4. *What is DBMS?*
A software package designed to create databases and manipulate data.
5. *What are the elements of a database?*
Entities
Attributes
Relation
Field
Record
6. *What are the database relationships?*
One to one
One to many
Many to many
7. *What are defined as Primary key, foreign key, composite primary key and referential integrity?*
Primary key: key in a relational database that is unique for each other.
Foreign key: a column that connect with PK in the original table.
Composite PK: A combination of two or more fields will help to identify a record uniquely.
Referential integrity: A relational database concept, which states that table relationships must always be consistent.
8. *What are the advantages of a database?*
Program data independency
Minimal data redundancy
Improved data sharing
Reduced program maintenance

1. What is network model?
 - All connected
2. What is relational model?
 - Has relationship with each table
3. What is hierarchical model?
 - Same level doesn't have connectivity.
 - One parent table can have more than one child.
 - Each child table can have only one parent data table.

Entity – relationship diagram

Symbols in ER Diagram



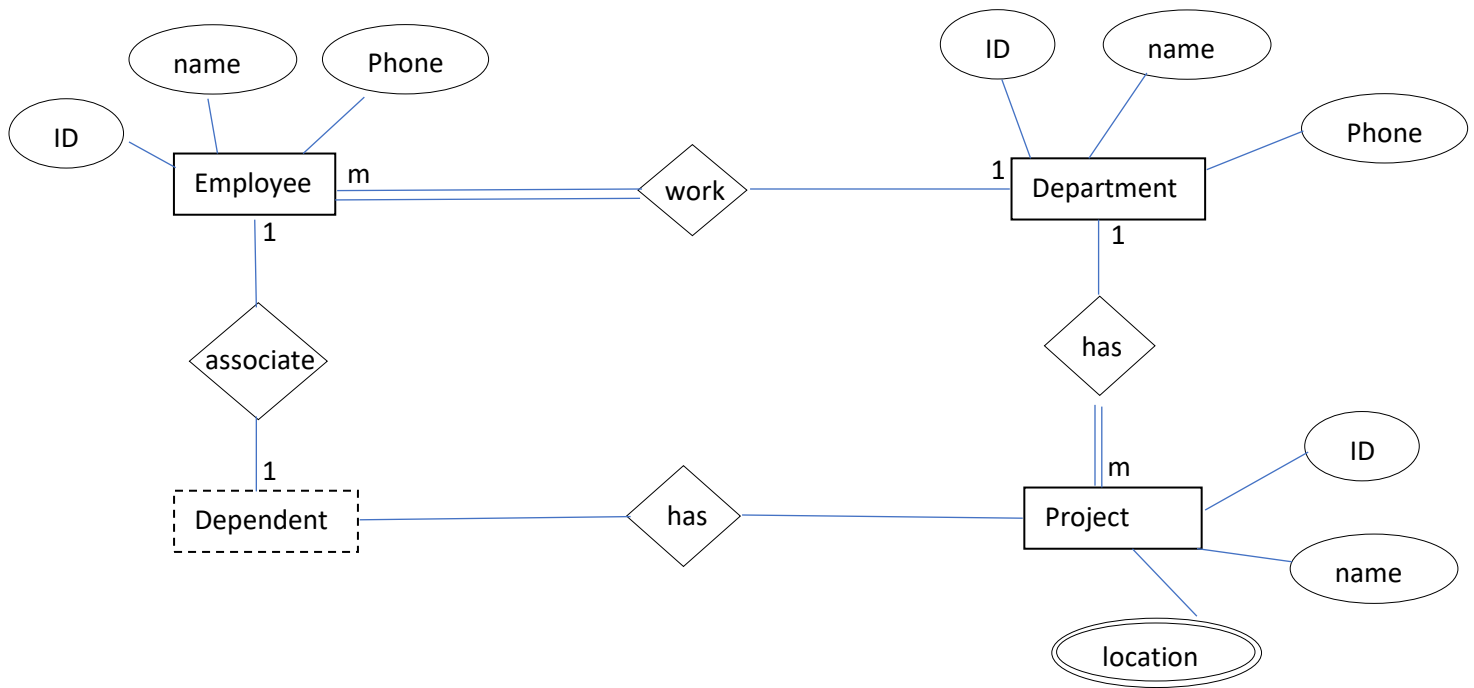
1. what are the three levels in database architecture?
 - External
 - Conceptual
 - Internal
2. What is data modeling?
 - A collection of tools and techniques for describing data and their relationships when designing databases.
3. What are the three types of data models?
 - Conceptual model
 - Logical model
 - Physical model
4. What is database schema and instance?
 - Schema – data type (design of the DB)
 - Instance – variable (moment of the DB)
5. What is data dictionary?
 - Data above the data. Consists of the basic definitions or the organization of a database and its tables.
6. What is data independence?
 - The capacity to change the schema at one level of the database system without affecting to the schema at the next higher level.
7. What are the two levels of data independence?
 - Logical data independence
 - Physical data independence

Question 01

Employees are working for a department. Each department has number of projects. The employees are allocated for their department's projects. Each project has number of locations. Employees are associated with dependents. The following attributes are stored In the DB.

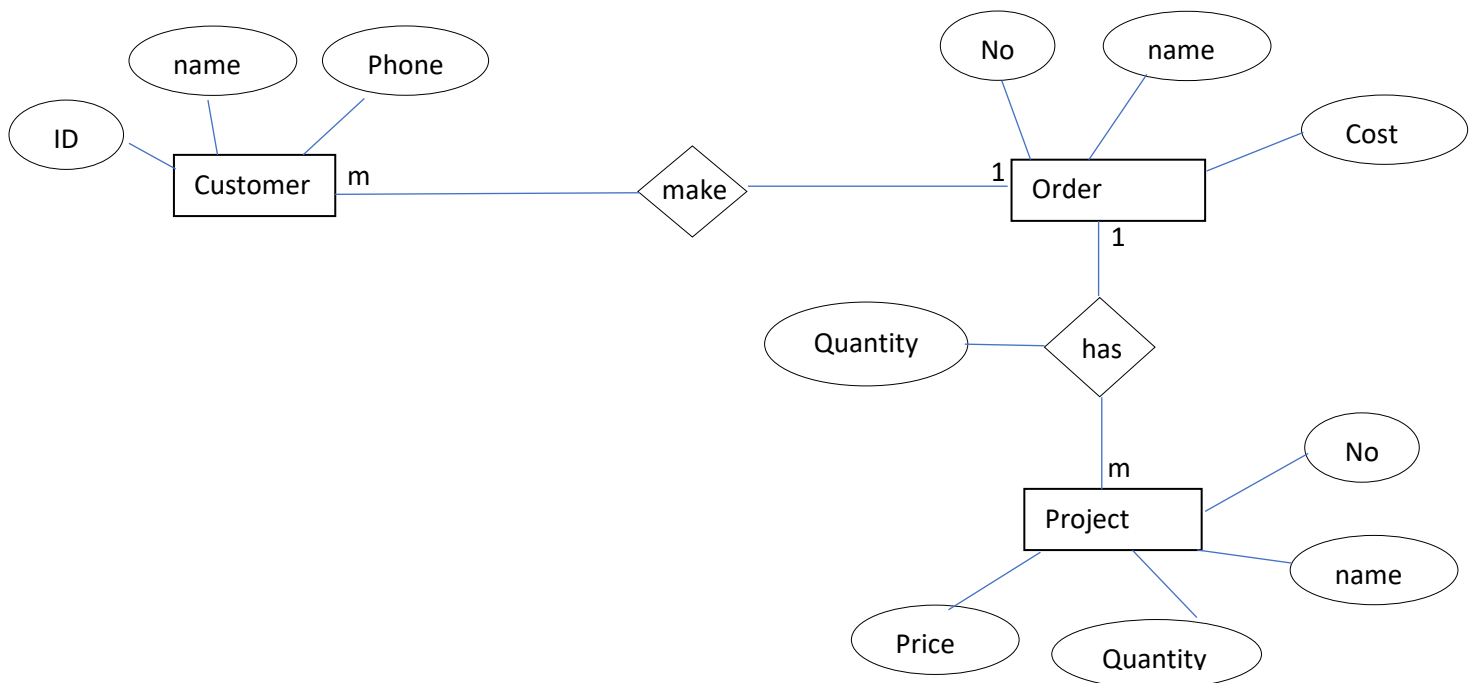
- a) Department ID, Name, Phone no
- b) Project ID, Name
- c) Employee ID, Name, Phone no

Create the ER diagram and mention the assumptions.



Question 02

Customer makes number of orders. Each order is stored with order No, order date & cost. Each order has number of products. Product No, name and unit price are stored. When the order is placed, the stock will be managed by the system. For this order quantity will be store in the DB.



SQL

1. Create database

- `CREATE DATABASE Database_Name`

2. Create table

- `CREATE TABLE table_name`

3. Insert data into table

- `INSERT INTO table_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);`
- `INSERT INTO table_name VALUES (value1, value2, value3, ...);`

4. Delete all the data

- `DELETE FROM table_name;`

5. Delete the whole table

- `DROP TABLE table_name;`

The DELETE command deletes one or more existing records from the table in the database. The DROP Command drops the complete table from the database.

6. Primary key and foreign key

- `CREATE TABLE table_name (
 OrderID int NOT NULL,
 OrderNumber int NOT NULL,
 PersonID int,
 PRIMARY KEY (OrderID),
 FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);`

Q] how to convert ERD to schema?

To convert ERD to schema, have to follow the steps that mentioned below.

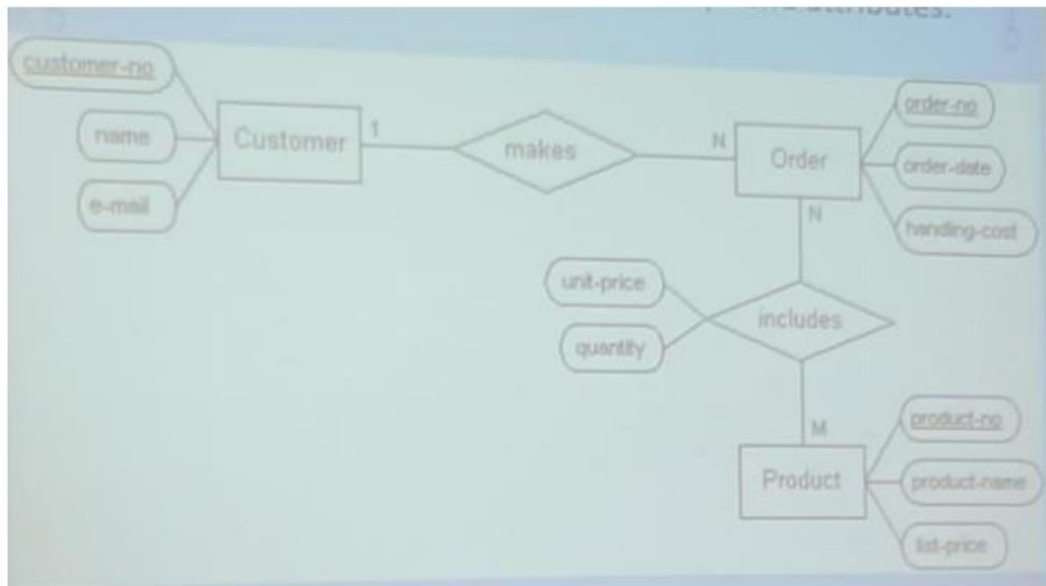
1. If there is a composite attribute in the ERD, have to make separate columns in the main table and if there is multivalued attribute don't store in the table.
2. To convert weak entity, have to make a new/ separate table and take the primary key of strong entity as the foreign key.
3. If there is a 1:1 relationship and both sides are partial of both sides are total, take the primary key from either side but if there is one side partial and one side total, take the primary key from the partial side as the foreign key.
4. If there is a 1:M relationship, take the primary key from the 1 side as the foreign key.
5. If there is a M: N relationship take the primary key from either side.
6. If there is a n-ary or ternary relationship, create a separate table and take the primary keys of every entity in the relationship to the separate table.
7. To map multivalued attributes, create a separate table and take the primary key of the attached entity.

SQL 2

Select statement

1. `SELECT column1, column2, ...
FROM table_name;`
2. `SELECT * FROM table_name;`

Convert ERD to schema.



Customer(customer_no,name,email)

Order(order_no,order_date,handling_cost,customer_no)

Product(product_no,product_name,list_price)

Orderproducts(order_no,product_no,unit_price,quantity)

Functional Dependency

- Because of one column the entire table is affected that is functional dependency.
- How one person's action affects the other one.

Eg: a-> b

'a' decides 'b'. but 'b' is not deciding 'a'.

StudentNo	Marks
1	80
2	70
3	90

In this case, Student 1's mark is 80. Then if 1 comes the marks should be 80. But if the mark value is 80, then we can't exactly say that the StudentNo should be 1. It can be any other StudentNo whose having 80 marks.

Anomalies

ENO	Name	Position	Salary
1	A	Driver	3 000
2	B	Driver	3 000
3	C	Manager	10 000
4	D	Supervisor	8 000

Duplication Anomaly

Here the Position and the Salary is duplicated again and again.

Update Anomaly

When we need to add more 100 Drivers, this will create an Update Anomaly.

Insertion Anomaly

When we need to add an assistant manager, we can't do that as it will occur INSERTION ANOMALY. That means we can't add such position because that data wasn't there before.

Potential Deletion Anomaly

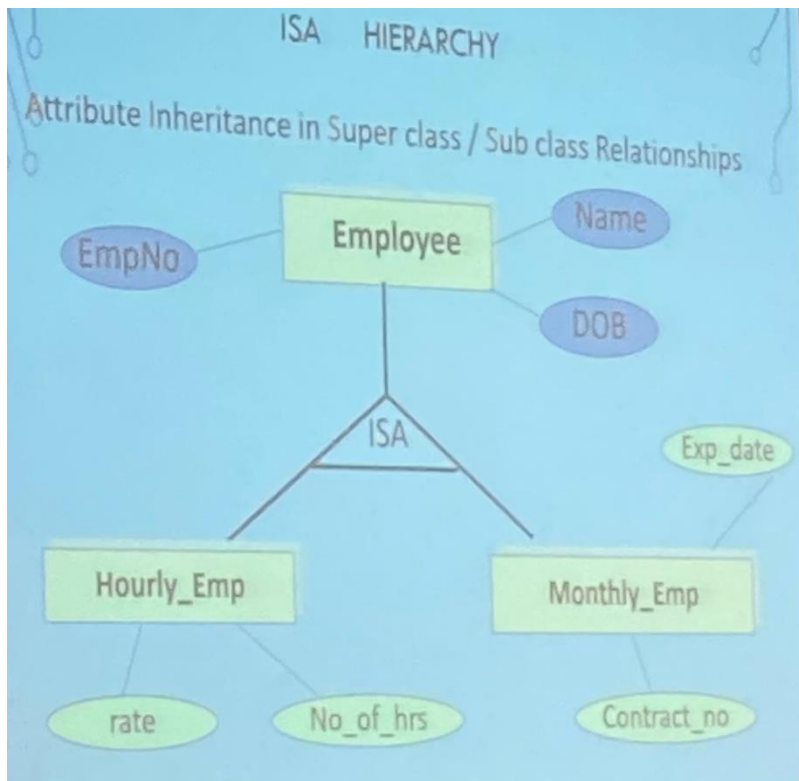
When ENO 4 is leaving, we need to delete its data and info. But there's only one supervisor. There's only one supervisor for the company. So, the position will be empty!!! Those data are called Potential Records!!! Deleting a Potential record will occur Potential Deletion Anomaly.

Normalization

The process of structuring a relational database in accordance with a series of so-called normal forms in order to reduce data redundancy and improve data integrity.

IS A HIRERARCHY

Attribute inheritance in Super class/ sub class relationships.



Overlap

- An entity occurrence may be a member of more than one subclass.

Union

- The result of this operation is a relation that includes all tuples that are either in A or in B or in both.
- Duplicate tuples are eliminated.
- Used to combine data from one or more tables into new rows.
 - Columns are not combined to create results, rows are combined.
 - The rows are in the same result.
 - Data from the first table is in one set of rows, and the data from the second table in another set.
- Typically used where you have two results whose rows you want to include in the same result set.

Requirements to apply union

1. The number of columns must be the same for both select statements.
2. The columns, in order, must be of the same data type.

Use the “All” keyword to keep all rows from both select statement’s results.

Intersection

- The result of this operation is a relation that includes all tuples that are in both A and B
- Duplicate tuples are eliminated.

Difference

- The result of this operation is a relation that includes all tuples that are in R but not in S (R-S)

SQL

1. SELECT * FROM A UNION SELECT * FROM B
2. SELECT * FROM A INTERSECT SELECT * FROM B
3. SELECT * FROM A MINUS SELECT * FROM B
4. SELECT * FROM B MINUS SELECT * FROM A

Join

Combines rows from two or more tables when common fields satisfy a condition.

Common columns are usually in a primary key- foreign key relationship (PK-FK).

10/11/2022

EmpID	Name	Date joined	Salary	Dept No	Designation
E1	JOHN	10/10/2013	75 000	D1	EXECUTIVE
E2	PETER	01/04/2014	90 000	D2	MANAGER
E3	ANN	15/06/2014	85 000	D1	MANAGER
E4	TOM	28/01/2015	38 000	D3	CASHIER
E5	JIMMY	06/12/2016	60 000	D1	EXECUTIVE

1. Display the different positions available in the company

→ `SELECT DISTINCT designation from Emp;`

2. Display the name and EmpID of the employees who works in Dept 01

→ `SELECT Name, EmpID from Emp WHERE DeptNo = "D1";`

3. Display the EmpID, name according to their salary (ascending)

→ `SELECT EmpID, Name from Emp ORDER BY Salary ASC;`

**When using order by, we can't use "WHERE". Instead of that we use "HAVING".*

EXERCISE 02

1. QUERY A: retrieve EmpID and Name of all employees who are working in Department of 1 from Emp table.

→ `CREATE TABLE A AS SELECT EmpID, Name FROM Emp WHERE DeptNo = "D1";`

2. QUERY B: retrieve EmpID and name of all managers from Emp table.

→ `CREATE TABLE B AS SELECT EmpID, Name FROM Emp WHERE Designation = "Manager";`

3. Find the following.

a) $A \cup B$ = `SELECT * FROM A UNION SELECT * FROM B;`


b) $A \cap B$ = `SELECT * FROM A INTERSECT SELECT * FROM B;`

c) $A - B$ = `SELECT * FROM A MINUS SELECT * FROM B;`

d) $B - A$ = `SELECT * FROM B MINUS SELECT * FROM A;`

JOINS

SQL Equijoin

Employee  Dno=Dnumber Department

```
SELECT *  
FROM Employee  
JOIN Department  
ON Dno = Dnumber
```

SQL Natural Join

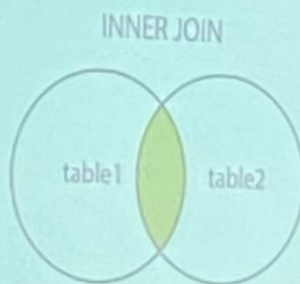
- Both fields should have the same name and data type.

Employee  Department

```
SELECT *  
FROM Employee  
NATURAL JOIN Department
```

INNER JOIN

- Returns all rows from participating tables where the join condition is met.
- Displays only the rows that have a match in both the joined tables.



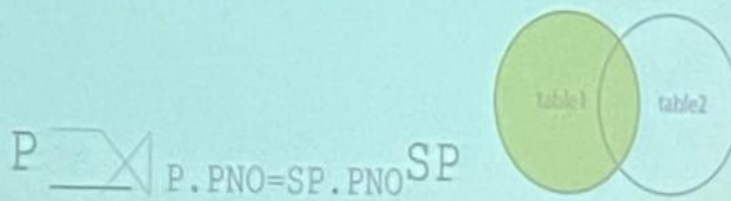
OUTER JOIN

- Returns all rows from both the participating tables which satisfy the join condition along with rows which do not satisfy the join condition.
- Where join condition value is not present in the second table, results table padded out with null values.

- LEFT JOIN
- RIGHT JOIN
- FULL JOIN

LEFT OUTER JOIN

- Return all rows from the left table, and the matched rows from the right table.
- The result is NULL in the right side when there is no match.



Example:

Customers

CustomerId	Name
1	Steven
2	Neena
3	Lex Deane

Orders

OrderId	CustomerId	OrderDate
100	1	2014-02-26 22:46:02.700
200	4	2014-03-20 22:46:07.700
300	3	2014-03-21 22:46:07.700

INNER JOIN on CustomerId Column

RESULT:

CustomerId	Name	OrderId	CustomerId	OrderDate
1	Steven	100	1	2014-02-26 22:46:02.700
3	Lex Deane	300	3	2014-03-21 22:46:07.700

```
SELECT *
FROM Customers AS C
INNER JOIN Orders AS O
ON C.CustomerId = O.CustomerId
```

```
SELECT *
FROM Customers AS C
JOIN Orders AS O
ON C.CustomerId = O.CustomerId
```

Product

- Returns all possible combinations of rows from two tables.
- Produces Cartesian product of the tables that are involved in the join.
- The size of a cartesian product is the number of the rows in the first table multiplied by the number of rows in the second table.

$$a \times b$$

cross join

- `SELECT * FROM customers CROSS JOIN Orders;`

Self-join

- A table joined to itself (Unary relationships), especially when the table has a foreign key which references its own primary key.
- Can be view as a join of two copies of the same table.
- Self-join is used to retrieve the records having some relation or similarity with other records in the same table.
- Used where the same table needs to be visited twice.

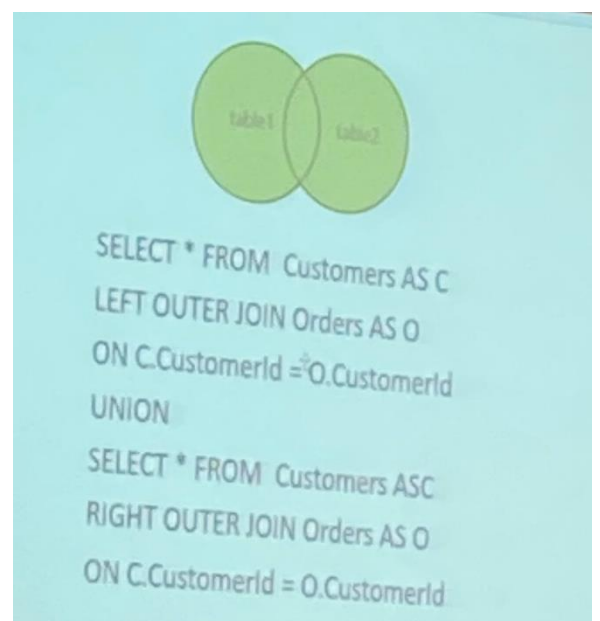
```
SELECT e.employeeID, e.name AS 'employee name',
```

```
       M.name AS 'manager name'
```

```
FROM employee AS E
```

```
JOIN employee AS M
```

```
ON e.ManagerID = m.EmployeeID
```



View

- CREATE VIEW abc AS SELECT eno, name FROM emp;
- SELECT * FROM abc;
- DROP VIEW abc;

Commit

- Confirming the changes happened.

Rollback

- Cancel the changes.

For more than one record as the output → joins

For only one output → subqueries

EMP			Dep	
ENO	Name	Dno	Dno	Dname
7	p	1	1	A
8	q	1	2	B
9	r	2		

select Dname from dep where dno=(select dno from emp where name=p)

P dep ?

select dno from emp where name=p

select Dname from dep where dno=1

Write the SQL sub queries.

Stud

Rno	Name
-----	------

Mark

Rno	Marks
-----	-------

1. Find Rno 1's mark.

→ `SELECT marks FROM mark WHERE Rno = 1;`

2. Find student chanaka's mark.

→ `SELECT marks FROM mark WHERE Rno = (SELECT Rno FROM stud WHERE name = "Chanaka");`

3. Find student's name whose mark is 20.

→ `SELECT name FROM stud WHERE Rno = (SELECT Rno FROM mark WHERE marks = 20);`

4. Find the students name who got the maximum mark.

→ `SELECT name FROM stud WHERE Rno = (SELECT Rno FROM mark WHERE marks = (SELECT MAX (marks) FROM mark));`

5. Display the students' names who passed the exam.

→ `SELECT name FROM stud WHERE marks in (<0,50>);`

PSQL

SQL triggers

- Used to specify automatic actions that the database system will perform when certain events and conditions occur.
- Provide an alternative way to check the integrity of data and also to run scheduled tasks.
- It is a stored program that **executes automatically** in response to a specific event associated with a table.
 - Insert
 - Update
 - Delete

CREATE TRIGGER

CREATE TRIGGER *trigger_name* ON *table_name* AFTER INSERT AS

SELECT * FROM *table_name*;