CS619-CS519

PROJECT+INTERNSHIP

ABSTRACT

Hy! Here is Junaid Malik, and I have completed BSCS. I have three+ years Experiences in this field. I work in these language PHP, LARAVEL, PYTHON (DJANGUE+FLASK). Here is Final Viva Preparation Notes. I Will this PDF cover all the aspect of Final Viva

JUNAID MALIK

0304-1659294

What is MySQL?

MySQL is a relational database management system (RDBMS) developed By Oracle that is based on Structured Query Language (SQL).

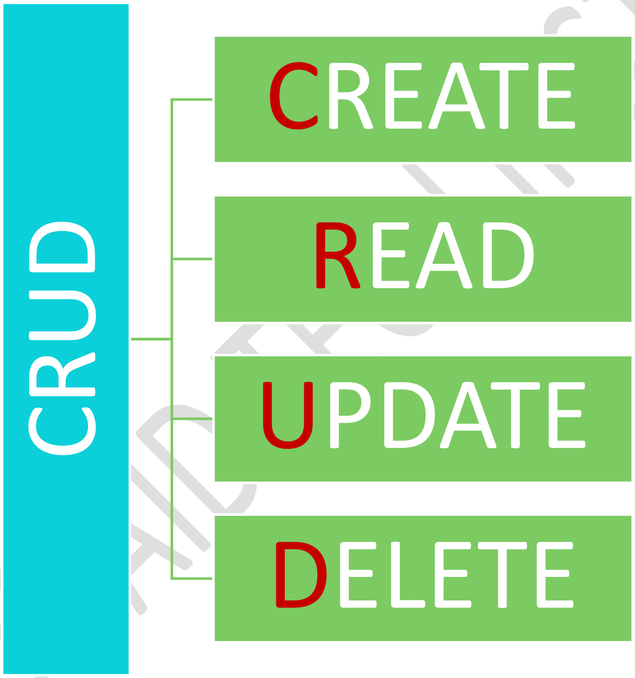
Database:

A Database is an organized collection of structured information or Data.

Like as Register or any product when it scene it show whole details this is database.

WHAT is DBMS:

Database Management System (DBMS) are software system used to store, retrieve and run queries (code) on Data.



CRUD

CREATE

READ

UPDATE

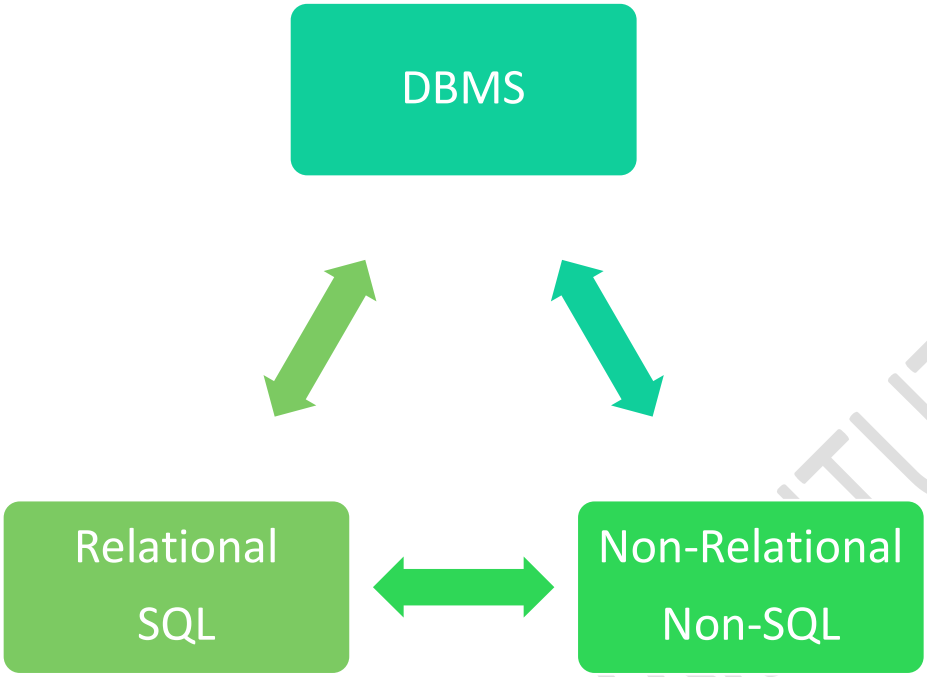
DELETE

RDBMS (Relational Database Management System):

- **Large-scale Projects:** RDBMS is commonly used in large-scale projects where data integrity, consistency, and relational structure are crucial.
- **Structured Data:** If your project deals with structured data that can be organized into tables with defined relationships, RDBMS is often the preferred choice.
- **Complex Queries:** RDBMS systems offer powerful querying capabilities using SQL (Structured Query Language), which can efficiently handle complex data retrieval and analysis tasks.
- **Data Integrity and ACID Compliance:** RDBMS systems ensure data integrity and transactional consistency through features like foreign key constraints, transactions, and ACID properties.
- **Scalability:** RDBMS systems can scale to handle large volumes of data and high transaction rates, provided proper database design and optimization strategies are implemented.

DBMS (Database Management System):

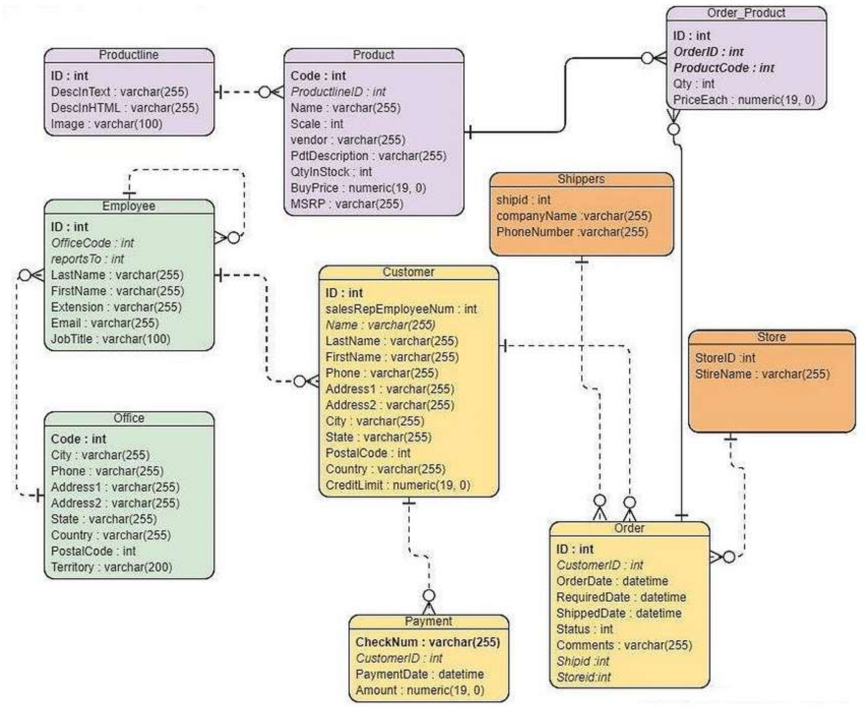
- **Small to Medium Projects:** DBMS systems are suitable for small to medium-sized projects where data volume and complexity are relatively low.
- **Less Structured Data:** If your project deals with less structured data or does not require complex relational structures, DBMS may suffice.
- **Lightweight and Easy to Manage:** DBMS systems are often lightweight and easier to set up and manage compared to RDBMS, making them suitable for projects with limited resources.
- **Non-Relational Data Models:** DBMS systems can accommodate non-relational data models such as hierarchical or network databases, which may be more appropriate for certain types of projects.



Relational SQL: It works on Structured Query Language. It is a collection of Data items with pre-defined relationships between them, stored in the form of table, rows and columns

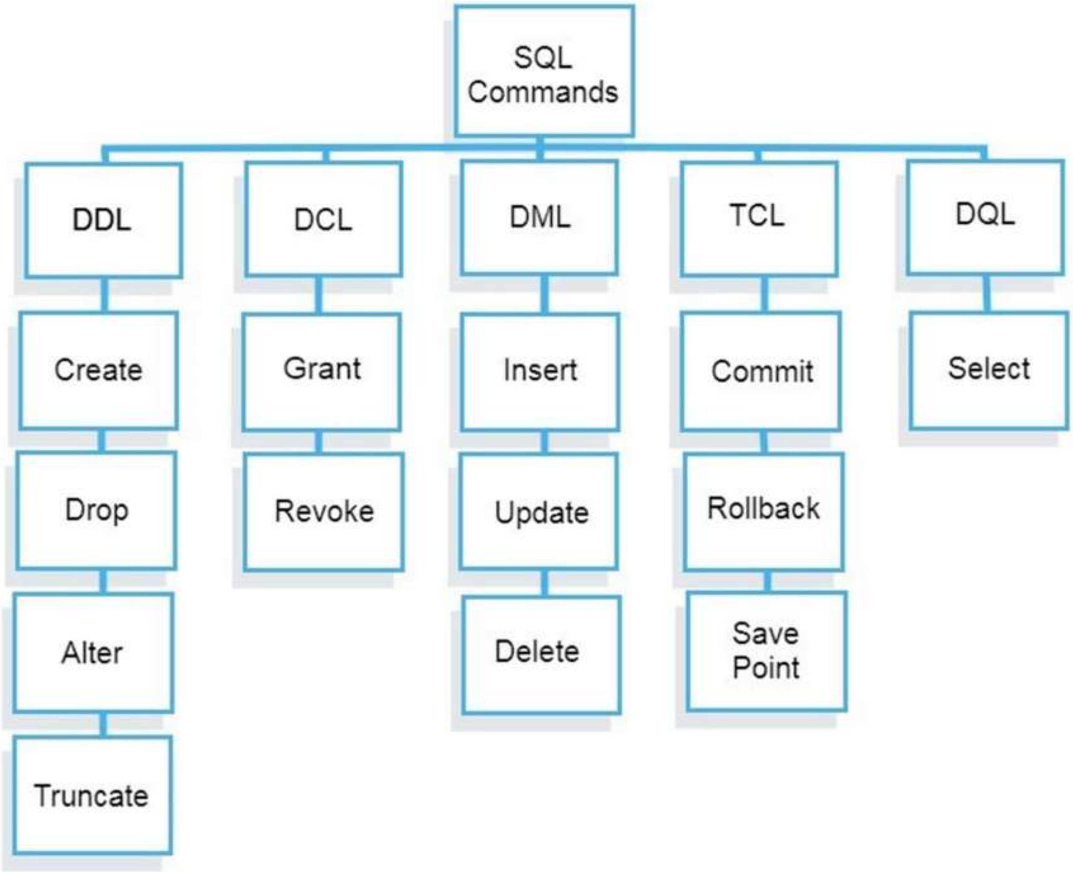
List of SQL:

- MySQL
- MariaDB
- Oracle
- PostgreSQL
- MSSQL



Non-Relational SQL: It’s not work in Structured Query Language.

SQL commands classified into four subgroups:


SQL Commands
DDL
Create
Drop
Alter
Truncate
DCL
Grant
Revoke
DML
Insert
Update
Delete
TCL
Commit
Rollback
Save Point
DQL
Select

DDL:

DDL is an abbreviation for Data Definition Language. It is concerned with database schemas and descriptions of how data should be stored in the database. DDL statements are auto-committed, meaning the changes are immediately made to the database and cannot be rolled back. These commands enable database administrators and developers to manage and optimize MySQL databases effectively.

DDL Command	Description
CREATE DATABASE	Creates a new database.
DROP DATABASE	Deletes a database.
CREATE TABLE	Creates a new table in a database.
ALTER TABLE	Alters the structure of an existing table.
DROP TABLE	Removes a table from a database.
CREATE INDEX	Creates an index on a table to improve a specific query performance.
CREATE VIEW	Creates a view, a virtual table based on one or more existing tables.
CREATE PROCEDURE	Creates a stored procedure, a precompiled SQL statement that can be run multiple times with different parameters.
CREATE FUNCTION	Creates a custom user-defined function that can be utilized in SQL statements.
CREATE TRIGGER	Creates a trigger, a type of stored procedure that is automatically executed when certain events occur, such as inserting, updating, or deleting data in a table.

MYSQL CONSTRAINTS:

- NOT NULL
- UNIQUE
- DEFAULT
- CHECK
- FORIGEN KEY
- PRIMARY KEY

Create Database:

```
Create Database db_name;
```

Create Table:

```
CREATE TABLE users (

   id int(11) NOT NULL PRIMARY KEY AUTO_INCREMENT,

   username varchar(255) NOT NULL,
   email varchar(255) NOT NULL,

   password varchar(255) NOT NULL

);
```

If we want to check Logic structure: `DESCRIBE table_name;`

SQL KEYS:

- Primary Key
- Supper Key
- Candidate Key
- Alternate or Secondary Key
- Composite Key
- Unique Key
- Foreign Key

SUPPER KEY:

Attributes of a Tables. Like as (id, name, email etc..). And also, a Uniquely Identifier.

ID	Name	Email	Gender	City
1	Junaid	junaid@gmail.com	M	Multan
2	Junaid	john@gmail.com	M	Multan
3	Ali	ali@gmail.com	M	JPPW
4	Alisha	alisha@gmail.com	F	LHR
5	Junaid	junaid@gmail.com	M	Multan

Candidate Key:

- A table contain Multiple Keys
- This key cannot store as a Null
- It must contain unique Values

First_name	Last_Name	Age
Junaid	Malik	20
John	Malik	21
Junaid	Malik	20

We cannot Find the Unique Row.

ID	regNo	lisenNo	contactNO	Name	Email	Gender	City
1	899	6775	567	Junaid	junaid@gmail.com	M	Multan
2	999	888	5	John	john@gmail.com	M	Multan
3	555	777	666	Ali	ali@gmail.com	M	JPPW
4	444	333	222	Alisha	alisha@gmail.com	F	LHR
5	333	222	111	Taiba	taiba@gmail.com	F	SLKT

PRIMARY KEY:

- A table cannot have More than One Primary Key
- It cannot be null
- Its value can never be Modified

ID	regNo	lisenNo	contactNO	Name	Email	Gender	City
1	899	6775	567	Junaid	junaid@gmail.com	M	Multan
2	999	888	777	John	john@gmail.com	M	Multan
3	555	777	666	Ali	ali@gmail.com	M	JPPW
4	444	333	222	Alisha	alisha@gmail.com	F	LHR
5	333	222	111	Taiba	taiba@gmail.com	F	SLKT

regNo, LisenNo, ContactNo are called Alternate Candidate Key

UNIQUE KEY:

- A Table can have More than one Unique Key
- The Unique key can allow only one Null value in a column

ID	regNo	lisenNo	contactNO	Name	Email	Gender	City
1	899	6775	567	Junaid	junaid@gmail.com	M	Multan
2	999	888	777	John	john@gmail.com	M	Multan
3	555	777	666	Ali	ali@gmail.com	M	JPPW
4	444	333	Null	Alisha	alisha@gmail.com	F	LHR
5	333	Null	Null	Taiba	taiba@gmail.com	F	SLKT

COMPOSITE KEY:

- A composite key in a relational database is a combination of two or more columns that uniquely identify each row in a table.

ID	Name	Email	Gender	City	forigenKey
1	Junaid	junaid@gmail.com	M	Multan	2
2	John	john@gmail.com	M	Multan	4
3	Ali	ali@gmail.com	M	JPPW	2
4	Alisha	alisha@gmail.com	F	LHR	3
5	Taiba	taiba@gmail.com	F	SLKT	3

FOREIGN KEY:

- It can be used to create a relationship between two table
- A primary key can take as a reference from one table and create a column in another table as a Foreign Key

ID	Name	Email	Gender	City	forigenKey
1	Junaid	junaid@gmail.com	M	Multan	2
2	John	john@gmail.com	M	Multan	4
3	Ali	ali@gmail.com	M	JPPW	2
4	Alisha	alisha@gmail.com	F	LHR	3
5	Taiba	taiba@gmail.com	F	SLKT	3

ALTER TABLE:

```
ALTER TABLE users ADD password varchar(255) NOT NULL;
```

```
ALTER TABLE users DROP COLUMS contact;
```

Create Index:

```
CREATE INDEX idx_name

ON users (name, email);
```

DML:

DML stands for Data Manipulation Language. It deals with data manipulation and includes the most common SQL statements such as SELECT, INSERT, UPDATE, DELETE, etc. DML statements are not auto-committed, meaning the changes can be rolled back if necessary. By mastering these DML commands, you can efficiently manipulate data in MySQL databases.

DML Command	Description
SELECT	Retrieves data from a table.
INSERT	Inserts new data into a table.
UPDATE	Updates existing data in a table.
DELETE	Deletes data from a table.
REPLACE	Updates or inserts a record into a table.
MERGE	Performs a UPSERT operation (insert or update) on a table.
CALL	Calls a stored procedure or Java subprogram.
EXPLAIN	Displays the execution plan for a given query.
LOCK TABLE	Locks a table to prevent other users from modifying it while a transaction progresses.

MERGE:

```
ALTER TABLE product

ADD COLUMN user_name VARCHAR(255),

ADD COLUMN user_email VARCHAR(255);
```

MERGE:

```
UPDATE product p JOIN users u ON p.id = u.id SET p.user_name = u.name,
p.user_email = u.email;
```

SELECT EVEN NUMBER:

```
SELECT * FROM users WHERE id % 2 = 0;
```

SELCT OR OPERATOR:

```
SELECT * FROM `users` WHERE name='xyz' OR fee=50;
```

SELCT AND OPERATOR:

```
SELECT * FROM `users` WHERE name='ali' AND fee=100;
```

SELCT BETWEEN OPERATOR:

```
SELECT * FROM users WHERE age BETWEEN 19 AND 25;
```

SELECT IN OPERATOR:

```
SELECT * FROM users WHERE age IN (19, 20, 21, 22, 23, 24, 25, 26);
```

SELECT NOT OPERATOR:

```
SELECT *FROM users WHERE NOT age=19 AND NOT name='john';
```

UPDATE QUERY:

```
UPDATE table_name SET colum_name='junaid' WHERE id=value;
```

UPDATE QUERY:

```
DELETE FROM table_name WHERE id=value;
```

MYSQL LIKE:

The LIKE operator is used in WHERE clause to search a specific Pattern in Column.

These are Two Wild Card Words:

- The Percent Sign (%) represent Zero, One OR Multiple Character
- The Underscore (_) represent One OR Single Character

PATTERN:

LIKE Operator	Description
LIKE a%	Start With a
LIKE %a	End With a
LIKE %or% 99.99%	Have a or in any position
LIKE _r%	Have r in the Second Position
Like a_%	Start With a and are at least 2 Character in Length
Like a__%	Start With a and are at least 3 Character in Length

LIKE a%o	Start with a and End with o

SYNTAX:

```
SELECT * FROM users WHERE name LIKE 'j%n';
```

ORDER and DISTINCT:

- ASC
- DSC

```
SELECT * FROM `users` ORDER BY name ASC;
```

DISTINCT:

- Only Unique value Find

```
SELECT DISTINCT name FROM users;
```

```
SELECT DISTINCT name FROM users ORDER BY name DESC;
```

LIMIT and OFFSET:

LIMIT:

If we want to LIMIT, the Number of results that are returned you can simply use the LIMIT command with several Rows to LIMIT by:

SYNTAX:

SELECT*FROM table_name LIMIT NUMBER_TO_LIMIT_BY

```
SELECT*FROM users LIMIT 3;
```

```
SELECT*FROM users ORDER BY name DESC LIMIT 3;
```

OFFSET:

- WHERE WE GIVE LIMIT
- It is Commonly used in PAGINATION:

```
SELECT*FROM users ORDER BY name DESC LIMIT 3,3;
```

AGGREGATE FUNCTIONS:

- Count (): Return the Number of Rows in a Database Table.
  ```
  SELECT COUNT(id) FROM users WHERE age>20;
  ```
- SUM (): Return the total Sum of a numeric Column.
  ```
  SELECT SUM(fee) FROM users;
  ```
- AVG (): Calculate the average of a set of values.
  ```
  SELECT AVG(fee) FROM users;
  ```
- MIN (): Return the Lowest Value (minimum) in a set of non-Null values.
  ```
  SELECT MIN(age) FROM users;
  ```

- MAX (): Return the greatest value (maximum) in a set of non-Null values.

```
SELECT MAX(age) FROM users;
```

FORIGEN KEY:

```
CREATE TABLE `order`(
   id int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,
   fkuserID int(11) NOT NULL,
   FOREIGN KEY (fkuserID) REFERENCES users(id)
  )
```

NORMALIZATION AND DENORMALIZATION:

NORMALIZATION:

It is a technique to reduce or remove the redundancy of a table.

Spread out a design smaller table into a Multiple Design.

- Reduce Data Redundancy and inconsistency
- Makes Write Faster
- Speed up insert, Update, Delete
- Data Integrity is maintained
- Good For OLTP (Online Transaction Processing)

uID	Uname	prID	Prname	orID	Orname	traID	trType		

user

Uid	Uname

Product

prID	prname

order

orID	Ordetail	FkuID	fkprID

Transection

trID	Type	fkorID

- More of joins

We have two level of Duplicasy.

- Row Level
- Column Level

Row Level:

stdID (not Null and Unique)	**sname**	**Age**
1	Junaid	22
2	John	23
1	Junaid	22
4	Ali	21
5	Taiba	30

Here in this table, we make stdID as a Primary Key. In this condition it cannot be Null, it must be unique.

stdID (pK)	**sname**	**Age**
1	Junaid	22
2	John	23
3	Junaid	22
4	Ali	21
5	Taiba	30

Colum Level:

stdID	sName	cID	Cname	fID	Fname	Salary
1	Junaid	C1	HTML	F1	Usama	300
2	John	C2	PHP	F2	Jubair	400
3	Ali	C1	HTML	F1	Usama	300
4	Alisha	C1	HTML	F1	Usama	300
5	#	C1	HTML	F3	Usamn	500

Now you can see there is same Data Colum Level So Problems Occurs.

- Insertion Anomaly
- Updation Anomaly
- Deletion Anomaly

stdID	sName	cID	Cname	fID	Fname	Salary
1	Junaid	C1	Java	F1	Usama	300
2	John	C2	PHP	F2	Jubair	400
3	Ali	C1	HTML	F1	Usama	300
4	Alisha	C1	HTML	F1	Usama	300
5	Taiba	C2	PHP	F2	Jubaid	400
6	John	C1	PHP	F1	Usama	30

Insertion Anomaly

If university Launch a New Course

C7 and Laravel: So we cannot Insert Data

stdID	sName	cID	Cname	fID	Fname	Salary
1	Junaid	C1	Java	F1	Usama	300
2	John	C2	PHP	F2	Jubair	400
3	Ali	C1	HTML	F1	Usama	300
4	Alisha	C1	HTML	F1	Usama	300
&*$	//	C7	Laravel	F4	Taiba	5000

It should be Huge Problem So we cannot add.

Deletion Anomaly

DELETE FROM table_name WHERE stdID=1

stdID	sName	cID	Cname	fID	Fname	Salary
~~1~~	~~Junaid~~	~~C1~~	~~Java~~	~~F1~~	~~Usama~~	~~300~~
2	John	C2	PHP	F2	Jubair	400
3	Ali	C1	HTML	F1	Usama	300
4	Alisha	C1	HTML	F1	Usama	300

Now I want To Delete Course:

stdID	sName	cID	Cname	fID	Fname	Salary
1	Junaid	C1	HTML	F1	Usama	300
2	John	C2	PHP	F2	Jubair	400
3	Ali	C1	HTML	F1	Usama	300
4	Alisha	C1	HTML	F1	Usama	300

But Student and Teacher can never be Delete

Updation Anomaly

UPDATE table_name SET name=Junaid WHERE stdID=4;

UPDATE table_name SET contact=98765 WHERE stdID=4;

UPDATE table_name SET c1=98765 WHERE stdID=4;

stdID	sName	cID	Cname	fID	Fname	Salary	contact
1	Junaid	C1	HTML	F1	Usama	300	123456789
3	Ali	C1	HTML	F1	Usama	300	123456789
4	~~Alisha~~ Junaid	C1	Laravel	F1	Usama	~~300~~ 5000	~~123456789~~ 98765

This is Called Data Inconsistency.

So, the Solution is:

stdID	**sname**	**Age**
1	**Junaid**	**21**
2	**John**	**22**

cID	Cname
1	PHP
2	Laravel
3	MYSQL

Fid	fname	Salary
1	Junaid	2000
2	John	9000000

Normal Form:

- **INF:** A relation is a 1NF if all attributes must have an atomic value.
- **2NF:** A relation is said to be 2NF if it is in 1NF and all Non-Key Attributes (Non-Prime Attribute) are Fully Functional Dependent on the Prime attribute (No Partial Dependency)
- **3NF:** If already 2NF and No Transitive Dependency.
- **BCNF:** Already In 3NF and for X-> y X must be Super Key or Candidate Key
- **4NF:** It already BCNF and No have Multiple Dependency.
- **5NF:** It already in 4NF and there is no further non loss decomposed

1NF:

A relation is a 1NF if all attributes must have an atomic value.

Uid	Uname	phone
1	Junaid	0304-16599294/ 0304-1659294
2	John MALIK	0304-1659294

This is Non-Atomic Value

Uid	Uname	phone
1	Junaid	0304-16599294
1	Junaid	0304-16599294
2	John	0304-1659294

This is Atomic Value.

2NF:

A relation is said to be 2NF if it is in 1NF and all Non-Key Attributes (Non-Prime Attribute) are Fully Functional Dependent on the Prime attribute (No Partial Dependency)

stID	Name	prID(fk)	prName
1	saba	1	Libraries
2	Farah	1	Libraries

ID and depID make a Prime Attributes.

~~stdID-> Name~~

~~prID-> prName~~

This is Partial Dependency So this is Not in a 2NF

ID	Name	prName
1	Zenia	Libraries
2	Farah	Libraries

prID	prName
1	Libraries

3NF:

If already 2NF and No Transitive Dependency.

sID	Name	city	zipcode
1	Junaid	Multan	4192
2	John	Lahore	6100

This is 2NF cause Every attribute Depends on Prime Attributes.

~~Sid-> zipcode~~

~~Zipcode->city~~

Transitive Dependency

sID	Name	zipcode
1	Junaid	4192
2	John	6100

city	zipcode
Multan	4192
Lahore	6100

Now It is Not-Transitive Dependency.

BCNF:

Already In 3NF and for X-> y X must be Super Key or Candidate Key

sID	Name	zipcode
1	Junaid	4192
2	John	6100

city	zipcode
A	4192
B	6100

This is BCNF

JOINING and Its Type:

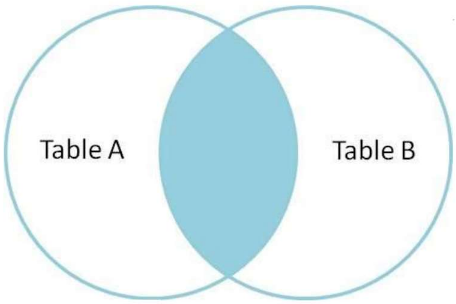
A "join" in the context of databases is an operation that combines rows from two or more tables based on a related column between them. The purpose of a join is to retrieve data from multiple tables simultaneously by establishing relationships between them.

For example, consider a database with two tables: "Employees" and "Departments". The "Employees" table might contain information about individual employees, such as their names, IDs, and department IDs. The "Departments" table might contain information about different departments, such as their names and IDs.

- INNER JOIN
- LEFT JOIN (or LEFT OUTER JOIN):
- RIGHT JOIN (or RIGHT OUTER JOIN):
- FULL JOIN (or FULL OUTER JOIN):
- CROSS JOIN:
- SELF JOIN:

INNER JOIN:

- The MySQL Inner Join is used to returns Only Those results From the Tables that match the specified condition and hides other rows and columns.
- MySQL assumes it as a default Join, so it is optional to use the Inner Join Keyword with the query.



Syntax:

```
SELECT columns
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name;
```

User:

Uid PK (int=255)	Uname	Uemail
1	John	Somth
2	Ali	//
3	Usama	//
4	Xyz	//
5	ABC	//

Order:

Uid	Fkuid (int=255)	quantity
1	2	23
2	3	56
3	2	89
4	6	23

```
SELECT * FROM order
INNER JOIN user ON order.fKuid=user.uid
```

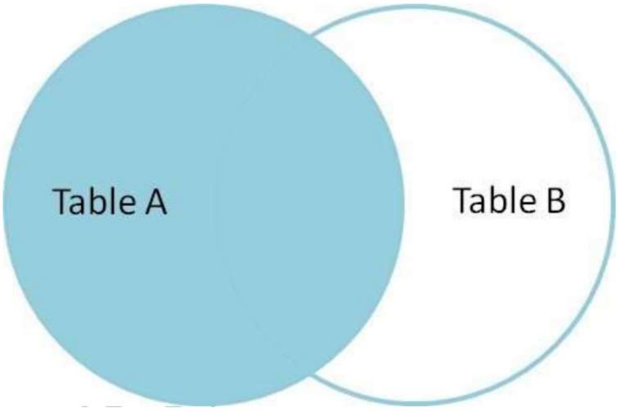
1== Ali and 23

2== Ali and 89

3== Usama and 56

LEFT JOIN:

- The MySQL JOIN keyword returns all record from the left table (table1) and the match records from the right table (table2)



Syntax:

```
 SELECT columns
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name;
```

User:

Uid PK (int=255)	Uname	Uemail
1	John	Somth
2	Ali	//
3	Usama	//
4	Xyz	//
5	ABC	//
6	xyzAbx	//

Order:

Uid	Fkuid (int=255)	quantity
1	2	23
2	3	56
3	2	89
4	6	23

1== John and Null

2== Ali and 23

3== Ali and 89

4== Usama and 56

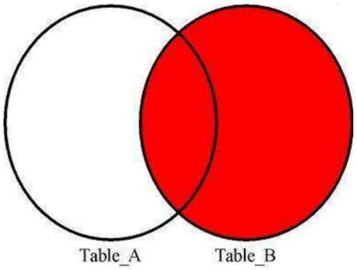
5== XYZ and Null

6== ABC and Null

7== XYZabx and 23

RIGHT JOIN:

- The Right Join Keyword return all records from the right table (table2) and the match record from the left table (table1)



Syntax:

```
SELECT columns
FROM table1
RIGHT JOIN table2
ON table1.column_name = table2.column_name;
```

FULL JOIN:

- This will select those record those have matching in both table and also left table and also Right table

Syntax:

```
SELECT columns
FROM table1
FULL JOIN table2
ON table1.column_name = table2.column_name;
```

```
$a=0;
For ($n=0; $n<=10; $n++){
echo
}
```

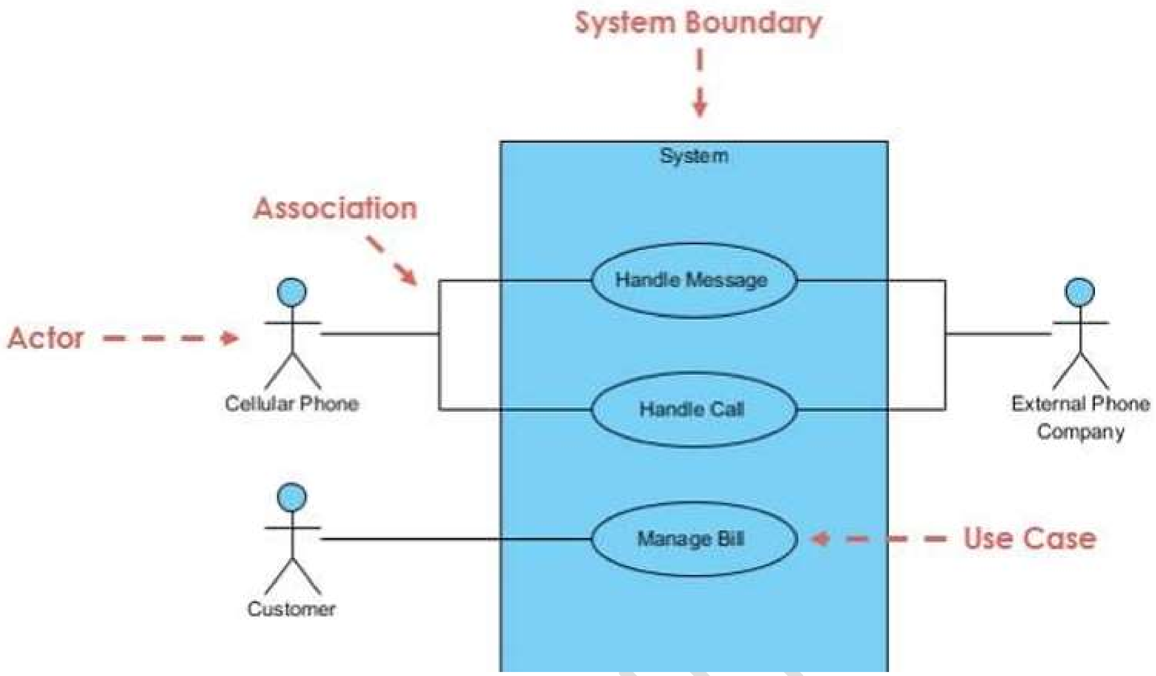
Data Integrity in a Database:

Data integrity refers to the overall accuracy, completeness, and reliability of data. It can be specified by the lack of variation between two instances or consecutive updates of a record, indicating that your information is error-free. It also corresponds to the security of data pertaining to regulatory compliance.

TYPES OF DATA INTEGRITY:

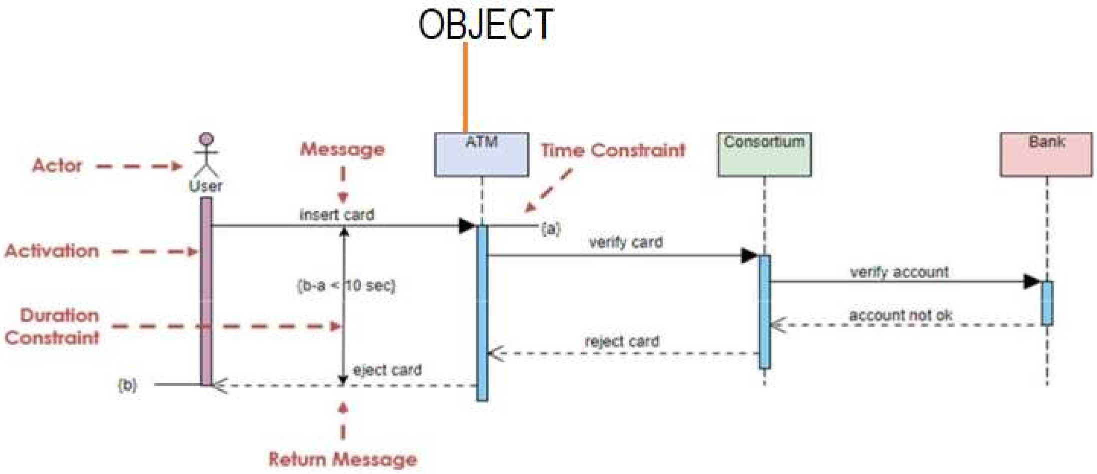
- Physical Integrity
- User Define Integrity
- Logical Integrity
- Entity Integrity
- Referential Integrity

USECASE DIAGARAM:



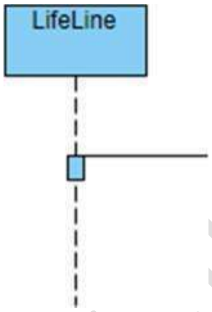
Icon	Name
⬬	Use Case
—	Association
웃	Actor
▯	System
<I>	Include
<E>	Extend
···>	Dependency
◁—	Generalization
R	Realization
⬬	Collaboration

SEQUENCE DIAGRAM:



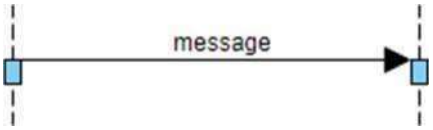
ACTIVATION

An activation is represented by a thin rectangle on a lifeline) represents the period during which an elementis performing an operation. The top and the bottom of the of the rectangle are aligned with the initiation and the completion time respectively



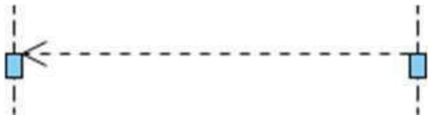
Calls Message:

A call message defines a particular communication between lifelines of an interaction, which represents an invocation of operation of target lifeline.



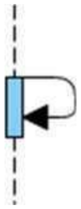
Return Message:

A return message defines a particular communication between lifelines of an interaction, which representsthe pass of information back to the caller of a corresponded former message.



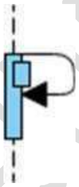
Self-Message:

A self-message defines a particular communication between lifelines of an interaction, which represents the invocation of message of the same lifeline.



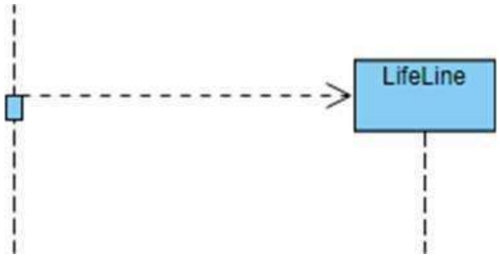
Recursive message

A recursive message defines a particular communication between lifelines of an interaction, which represents the invocation of message of the same lifeline. It's target points to an activation on top of the activation where the message was invoked from.



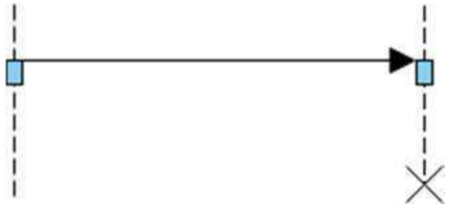
Create Message:

A create message defines a particular communication between lifelines of an interaction, which representsthe instantiation of (target) lifeline.



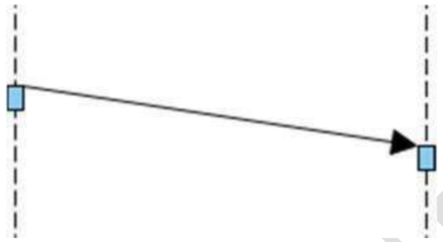
Destroy message

A destroy message defines a particular communication between lifelines of an interaction, whichrepresents the request of destroying the lifecycle of target lifeline.



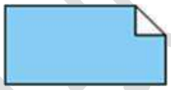
Duration message:

A duration message defines a particular communication between lifelines of an interaction, which showsthe distance between two-time instants for a message invocation.



NOTE:

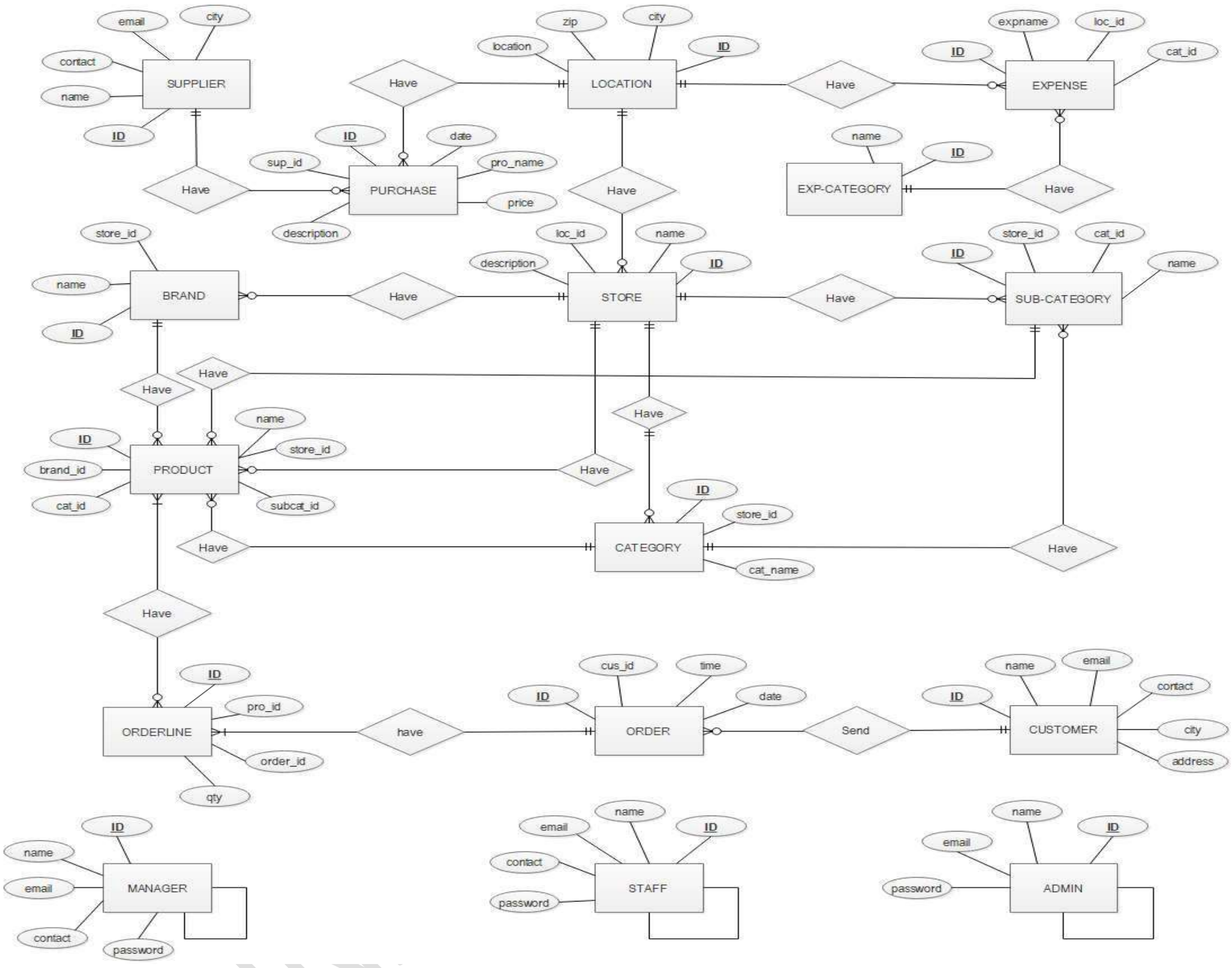
A note (comment) gives the ability to attach various remarks to elements. A comment carries nosemantic force, but may contain information that is useful to a modeler.



AL-JUNAID TECH INSTITUTE

ERD DIAGARM:

An Entity-Relationship Diagram (ERD) is a visual representation of the entities (real-world things like people, places, or concepts) and their relationships within a database system. It's like a blueprint that shows how data is organized and connected.

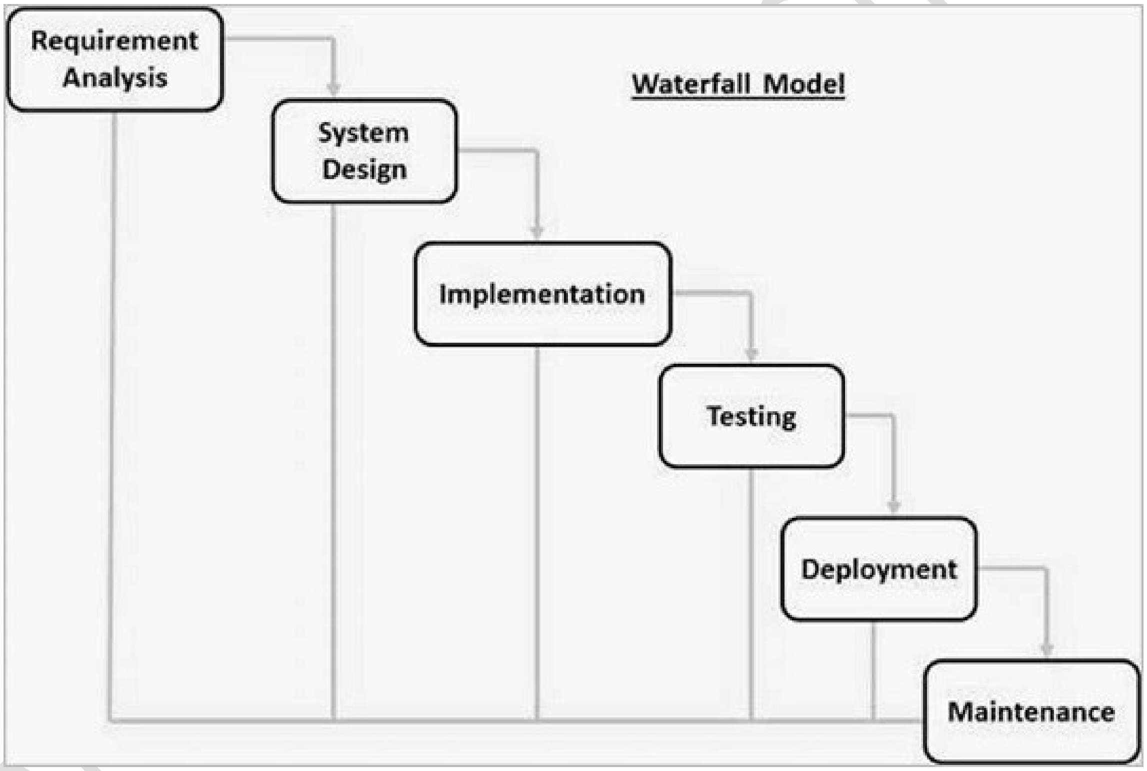


No	Symbol	Name
1	▭	Entity
2	⬭	Attributes
3	—	Association
4	◇	Relationship

WATERFALL MODEL:

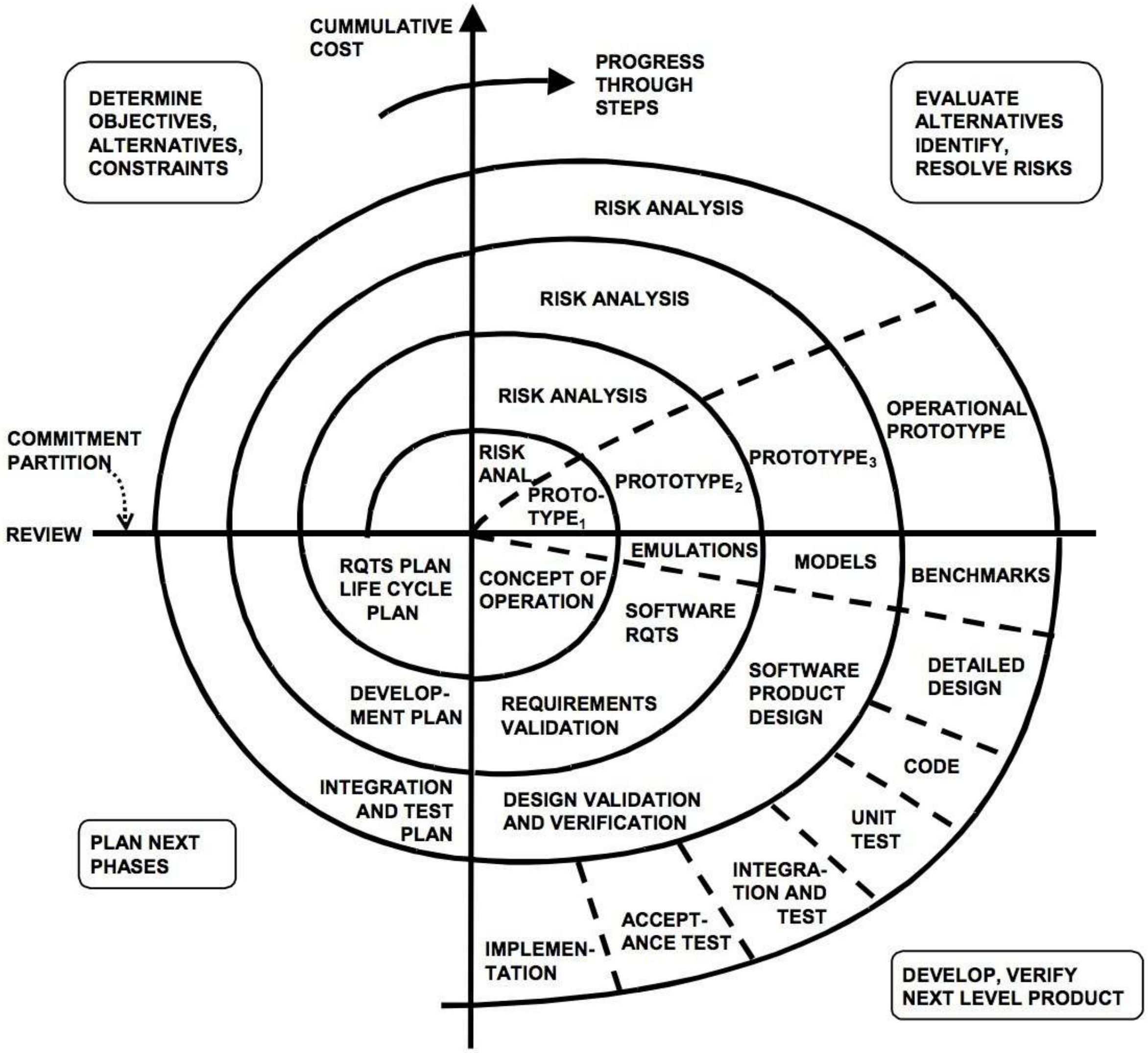
The Waterfall Model was the first Process Model to be introduced. It is also referredto as a **linear-sequential life cycle model**. It is very simple to understand and use.In a waterfall model, each phase must be completed before the next phase can beginand there is no overlapping in the phases

- **Used in Small Projects:** The Waterfall model is particularly effective for smaller projects with well-defined and stable requirements. Its straightforward approach is best suited for projects where changes are minimal.
- **Works Step by Step:** The model follows a linear and sequential process, moving through each phase methodically. This ensures that each aspect of the project is thoroughly addressed before progressing to the next stage.
- **Completes First Step Then Moves to the Next Step:** In the Waterfall model, each phase must be completed before the next one begins. This ensures a disciplined approach where all requirements are met in order, reducing the risk of overlooking important details.



SPIRAL MODEL:

The Spiral Model is a software development methodology that combines iterative development (prototyping) with the systematic aspects of the Waterfall model. It allows for incremental refinement of the system through each iteration or "spiral."



Phases of the Spiral Model:

- Planning Phase: Determine objectives, alternatives, and constraints.
- Risk Analysis Phase: Identify and resolve risks by developing prototypes and simulations.
- Engineering Phase: Develop and verify the next level of the product.
- Evaluation Phase: Assess the results of the engineering phase and plan the next iteration

Advantages:

- **Used in Large Projects:** The Spiral Model is particularly suited for large, complex, and high-risk projects. Its iterative nature allows for managing these complexities effectively by breaking the project into manageable parts.
- **User Can Move to Any Phase at Any Time**: This model offers flexibility, allowing the project team to revisit and refine previous phases based on new information or changes in requirements. This iterative approach ensures continuous improvement and adaptation throughout the project lifecycle.
- **Risk Analysis:** A fundamental aspect of the Spiral Model is its emphasis on risk analysis. Each iteration (or spiral) begins with identifying and analyzing potential risks, which helps in early detection and mitigation of issues. This ongoing risk assessment ensures that the project remains on track and minimizes the chances of encountering significant problems later on.

VU PROECSS MODEL:

It is the combination of Waterfall and Spiral Model.

For Example:

CS619 projects:

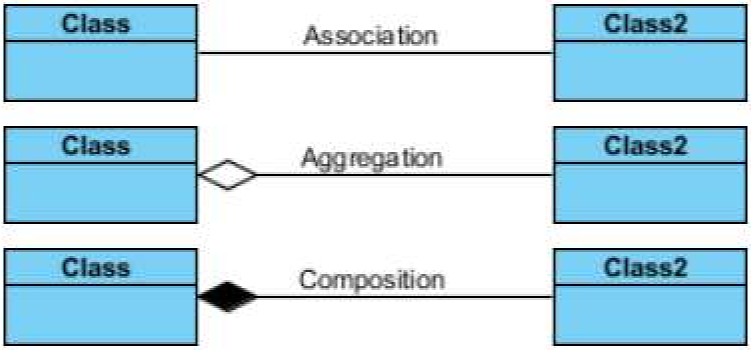
- SRS
- DD
- Prototype
- Final

Now, first we submit SRS, Then DD, Then Prototype and then Final Deliverable it shows Waterfall model means we work Step by Step.

When We submit Final Deliverable, we refine our SRS and DD by instructor Comments so its full fill the Spiral model. Why it shows Spiral model because we move back any phase at anytime

CLASS DIAGARM:

A class diagram is a type of static structure diagram in the Unified Modeling Language (UML) that describes the structure of a system by showing its classes, attributes, operations, and the relationships among the objects. It is widely used in software engineering for modeling object-oriented systems.



Association

If two classes in a model need to communicate with each other, there must be a link between them, and that can be represented by an association.

Association can be represented by a line between these classes **with an arrow indicating the navigation direction**. In case an arrow is on both sides, the association is known as a bidirectional association.

Composition:

A child (or part) is fully dependent on the parent (or whole)

For Example:

House (parent) and Room (child). Rooms will neverseparate into a House.

Aggregation:

"child" (or part) can exist independently of the "parent" (or whole). Aggregation represents a relationship where the child does not depend on the parent for its existence.

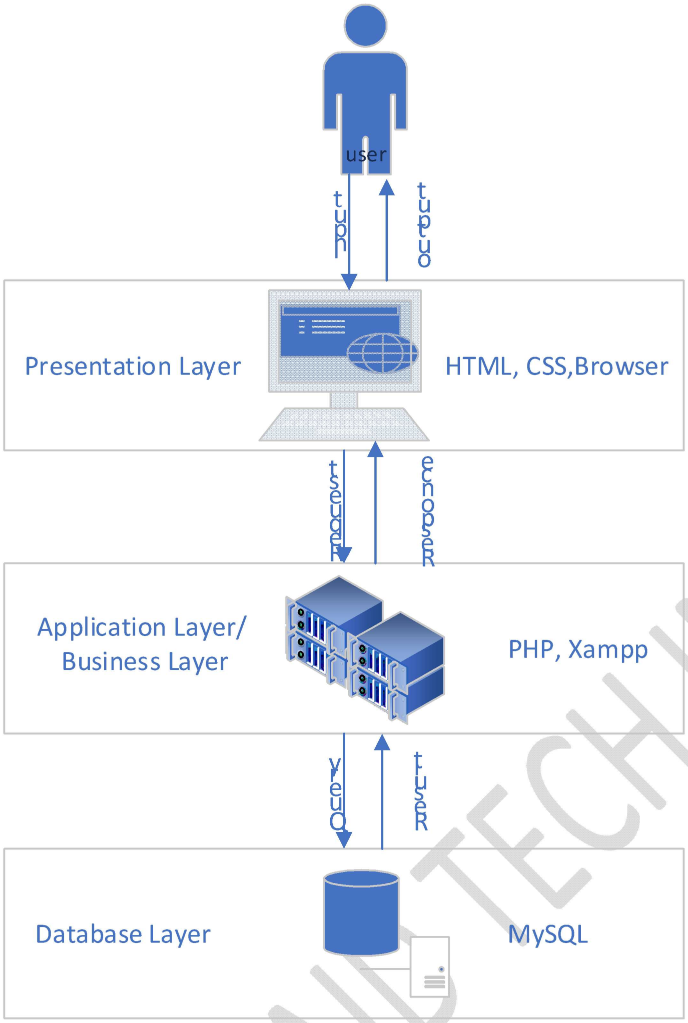
For Example:

Automobile (Parent) and Car (Child).So, if you delete the Automobile, the child Car still exist.

Database Diagram:

A database diagram is a visual representation of the structure of a database. It shows the tables, columns, and relationships between them.

Architectural Design:



3 Layer Architecture Diagram

<u>Presentation Layer</u>

- **<u>Components:</u>** HTML, CSS, Browser
- **<u>Role:</u>** This is the topmost layer that interacts with the user. It is responsible for presenting the user interface and handling user inputs and outputs.
- **<u>Technologies:</u>** Typically, this layer uses HTML for structure, CSS for styling, and is rendered in a web browser.
- **<u>Interactions:</u>** The user provides input (such as clicks, form submissions) which is sent to the application layer. The output from the application layer is displayed back to the user through the browser.

Application Layer (Business Layer)

- **Components:** PHP, XAMPP
- **Role:** This middle layer processes business logic and handles the flow of data between the presentation layer and the database layer. It processes requests from the presentation layer and sends appropriate responses back.
- **Technologies:** PHP is a common scripting language used here, and XAMPP is a popular software stack that includes Apache, MySQL, PHP, and Perl.
- **Interactions:** Receives requests from the presentation layer, processes them (e.g., executing scripts, applying business rules), and sends queries to the database layer. After processing the database response, it sends the results back to the presentation layer.

Database Layer

- **Components:** MySQL
- **Role:** This bottom layer is responsible for data storage, retrieval, and management. It handles queries from the application layer and returns the results.
- **Technologies:** MySQL is a widely-used relational database management system.
- **Interactions:** Receives queries from the application layer, processes these queries to fetch or update data, and returns the results to the application layer.