* **Module-1(Fundamental)**

1. **What is SDLC**

* SDLC is structure imposed on the development of a software product that defines the process for planning, implementation, testing, documentation, deployment and ongoing maintenance and support.
* A software Development Life Cycle is essentially a series of steps, or phases, that provides a model for development and lifecycle management of an application or piece of software.
* The methodology within the SDLC process can very across industries and organization, but standard such as ISO/IEC 12207 represent process that establish a lifecycle for software, and provide a mode for the development, acquisition, and configuration of software systems.

1. **What is agile methodology?**

* Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by repaid delivery of working software product.
* Agile method break the product into small internal builds.
* These builds are provide in iteration.
* Each iteration typically lasts from about one to three weeks.
* Each iteration involves cross functional teams working simultaneously on various areas like planning, requirement analysis, design, coding, unit testing and acceptable testing.
* Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In agile the tasks are divided to time boxes to deliever specific features for a release.

1. **What is SRS?**

* A software requirement specification(SRS) is a complete description of the behavior of the system to be developed.
* includes a set of use cases that describe all of the interactions that the users will have with the software.
* Use cases are also known as functional requirements. In addition to use cases, the SRS also contains nonfunctional requirements.
* Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance requirements, quality standard, or design constraints).
* Recommended approaches for the specification of software requirements are describe by **IEEE 830-1998.**
* This standard describe possible structure, describe contents, and qualities of the software requirements specification.

1. **What is OOPS.**

* Identify the objects and assigning responsibilities to these objects.
* Objects communicate to other objects by sending messages.
* Messages are received by methods of an object.
* An object is like a black box.
* Objects is derived from abstract data types.
* Object-object oriented programming has a web interacting objects, each house-keeping is own state.
* Objects of a programs interact by sending messages to each other.

1. **Write the basic concepts of oops.**

* OOPS concepts:

1. Class
2. Objects
3. Encapsulations
4. Abstractions
5. Inheritance
6. Polymorphism

**7.What is object?**

* An object represents an individual, identifiable item, unit, or entity, either real or abstract, with a well-defined role in the problem domain.
* An "object" is anything to which a concept applies.
* This is the basic unit of object-oriented programming (OOP).
* That is both data and function that operate on data are bundled as a unit called as object.

**8. What is class?**

* A class represents an abstraction of the object and abstracts the properties and behavior of that object.
* When you define a class, you define a blueprint for an object.
* This doesn’t actually define any data, but it does define what the class name means, that is, what an object of the class will consist of and what operation can be performed on such an object.
* Class can be considered as the blueprint or definition or a template for an object and describe the properties and behavior of that object, but without any actual existence.

1. **What is encapsulation?**

* Encapsulation is the practice of including in an object everything it needs hidden from other objects. The internal state is not accessible by other objects.
* Encapsulation is placing the data and the function that work on the data in the same place. while working with procedural language, it is not always clear which functions work on which variable but object-oriented programming provides you framework to place the data and the relevant data functions together in the same objects.
* Encapsulation in java is the process of wrapping up of data(properties) and the behavior (methods) of an object into single unit; and the unit here is the class (or interface).
* Encapsulate in plain English means to enclose or be enclosed in or as if in a Capsule (or unit).
* Encapsulations enables data hiding, hiding irrelevant information from users of a class and exposing only the relevant details required by the user.
* We can expose our operations hiding the details of what is needed to perform that operation.
* We can protect the internal state of an object by hiding its attributes from the outside the world (by making it private), and the exposing them through setter and getter method.

**10.What is inheritance?**

* Inheritance means that one class inherit the characteristic of another class. This is also called “is a” relationship
* One of the most useful aspects of object-oriented programming is code reusability. As the name the suggest inheritance is the process of forming a new class from an existing class called as base class and then add unique features helps to reduce the code size.
* Inheritance Describe the relationship between two classes. A class can get some of its characteristics from a parent class and then add unique features of its own.
* In general java supports single-parent, multiple-children inheritance and multilevel inheritance (Grandparent->Parent->Child) for classes and interfaces. Java supports multiple inheritances (multiple parents, single child) only through interfaces.
* In a class context, inheritance is referred to as implementation inheritance, and in an Interfaces context, it is also referred to as interface inheritance.

Grandparent

Parent

Child

* For Example consider a vehicle parent class and its child car.
* Vehicle class will have all common properties and functionalities for all vehicle inherit those properties which are specific to a car.
* Here, vehicle is known as base class, parent class or super class.
* Car is known as derived class, child class or subclass.

A car is vehicle

A dog is an animal

A teacher is a person

1. **What is polymorphism?**

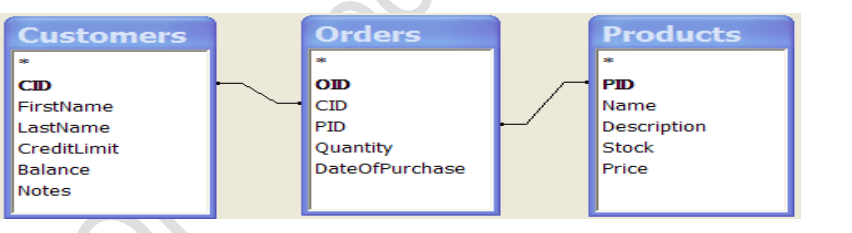
* Polymorphism means “having many forms”.
* It allows different object to respond to the same message in different way, the response specific to the type of the object.
* The most important aspect of an object is its behaviour (the things it can do). A behaviour is initiated by sending a message to the object (usually by calling a method).
* The ability to use an operator or function in different ways in other words giving different meaning or functions to the operators or functions is called polymorphism.
* Poly refers too many. That is a single function or an operator functioning in many ways different upon the usage is called polymorphism.
* E.g. the message dislaydetails() of the Person class should give different results when send to a Student object (e.g. the enrolment number).
* The ability to change form is known as polymorphism.
* There is two types of polymorphism in Java
* Compile time polymorphism (Overloading)
* Runtime polymorphism(overriding)

**12.what is RDBMS?**

* RDBMS stands for Relational Database Management System. RDBMS is the basis for SQL, and for all modern database systems like MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access.
* A Relational database management system (RDBMS) is a database management system (DBMS) that is based on the relational model as introduced by E. F. Codd.
* Most of today's databases are relational:
* Database contains 1 or more tables.
* Table contains 1 or more records
* Record contains 1 or more fields
* Fields contain the data

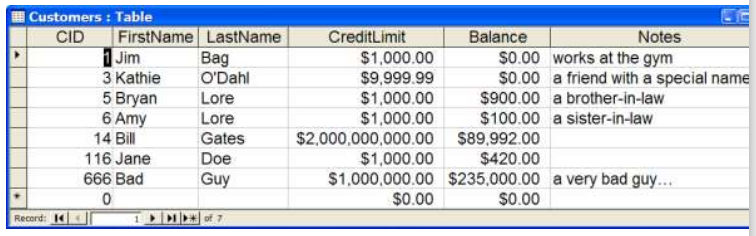
**EXAMPLE**

* Here's a simple database schema for tracking sales
* 3 tables, related by primary keys (CID, OID, PID)
* primary keys (in boldface) are unique record identifiers



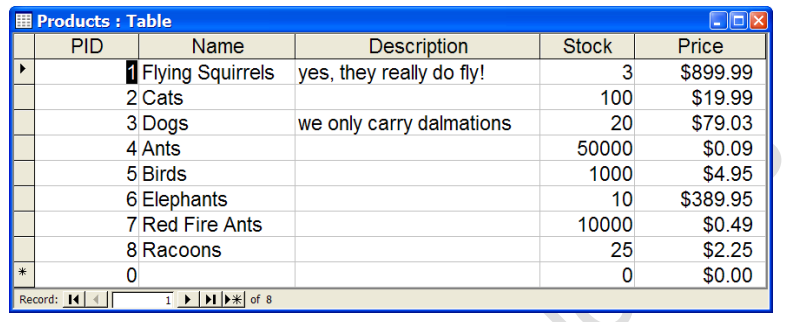
**Customers…..**

* Here's some data for the Customers table
* Ignore last row, it's a MS Access mechanism for adding rows…



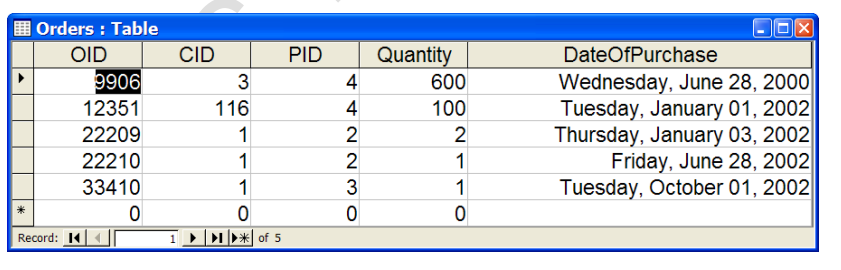
**Products…**

* Here's some data for the Products table



**Orders….**

* Here's some data for the Orders table
* How do you read this?
* e.g. order #9906 states Kathie O'Dahl purchased 600 Ants
* Must join tables together to figure that out..



**13.What is SQL?**

* SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in relational database.
* SQL is the standard language for Relation Database System. All relational database management systems like MySQL, MS Access, and Oracle, Sybase, Informix, Postgres and SQL Server use SQL as standard database language.
* Also, they are using different dialects, such as:
* MS SQL Server using T-SQL, ANSI SQL
* Oracle using PL/SQL,
* MS Access version of SQL is called JET SQL (native format) etc.
* SQL is a language of database, it includes database creation, deletion, fetching rows and modifying rows etc.
* SQL is an ANSI (American National Standards Institute) standard but there are many different versions of the SQL language.
* SQL is the standard programming language of relational DBs
* SQL is a standard computer language for accessing and manipulating databases.
* SQL is a great example of a declarative programming language
* Your declare what you want, DB engine figures out how…

**14. write SQL commands**

* DDL – Data Definition Language
* DML – Data Manipulation Language
* DCL – Data Control Language
* DQL – Data Query Language

**DQL- Data Query Language**

|  |  |
| --- | --- |
| **Command** | **Description** |
| **Select** | Retrieves from records from one or more tables |

|  |  |
| --- | --- |
| **Commands** | **Description** |
| **Insert** | Creates a record |
| **Update** | Modifies records |
| **Delete** | Deleted records |

**DML- Data Manipulation Language**

**DCL- Data Control Language**

|  |  |
| --- | --- |
| **Commands** | **Description** |
| **GRANT** | Gives a privilege to users |
| **REVOKE** | Takes back privileges granted from users |

**15.Draw Usecase on online book shopping**

**<<include>>**

**User**

**(Online book store system)**

**16.Draw Usecase on online bill payment system(Paytm)**

user Admin

**17.Write SDLC phases with basic introduction**

|  |  |
| --- | --- |
| Requirements Collection/Gathering | Establish Customer Needs |
| Analysis | Model And Specify the requirements- “What” |
| Design | Model And Specify a Solution – “Why” |
| Implementation | Construct a Solution In Software |
| Testing | Validate the solution against the requirements |
| Maintenance | Repair defects and adapt the solution to the new requirement |

**Requirement Gathering**

* Features
* Usage scenarios
* Although requirements may be documented in written form, they may be incomplete, unambiguous, or even incorrect.
* Requirements will Change!
* Inadequately captured or expressed in the first place
* User and business needs change during the project •
* Validation is needed throughout the software lifecycle, not only when the “final system” is delivered.
* Build constant feedback into the project plan
* Plan for change

* **There are two types of requirements:**
* **Functional Requirements**: describe system services or functions.
* Compute sales tax on a purchase
* Update the database on the server
* **Non-Functional Requirements**: are constraints on the system or the development process.
* Non-functional requirements may be more critical than functional requirements.
* If these are not met, the system is useless!

**Analysis phase**

* The analysis phase defines the requirements of the system, independent of how these requirements will be accomplished.
* This phase defines the problem that the customer is trying to solve.
* This analysis represents the “what” phase.
* The requirement documentaries to capture the requirements from the customer's perspective by defining goals.
* This phase starts with the requirement document delivered by the requirement phase and maps the requirements into architecture.
* This phase represents the “how” phase.
* Details on computer programming languages and environments, machines, packages, application architecture, distributed architecture layering, memory size, platform, algorithms, data structures, global type definitions, interfaces, and many other engineering details are established.
* The design may include the usage of existing components.

**Design phase**

* Design Architecture Document
* Implementation Plan
* Critical Priority Analysis
* Performance Analysis
* Test Plan
* The Design team can now expand upon the information established in the requirement document.
* The requirement document must guide this decision process.
* Analyzing the trade-offs of necessary complexity allows for many things to remain simple which, in turn, will eventually lead to a higher quality product. The architecture team also converts the typical scenarios into a test plan.

**Implementation Phase**

* In the implementation phase, the team builds the components either from scratch or by composition.
* Given the architecture document from the design phase and the requirement document from the analysis phase, the team should build exactly what has been requested, though there is still room for innovation and flexibility.
* For example, a component may be narrowly designed for this particular system, or the component may be made more general to satisfy a reusability guideline.
* Implementation - Code
* Critical Error Removal
* The implementation phase deals with issues of quality, performance, TOPS Technologies Pvt.Ltd Version Jan\_2018 Page 5 baselines, libraries, and debugging. The end deliverable is the product itself. There are already many established techniques associated with implementation.

**Testing phase**

* Simply stated, quality is very important. Many companies have not learned that quality is important and deliver more claimed functionality but at a lower quality level.
* It is much easier to explain to a customer why there is a missing feature than to explain to a customer why the product lacks quality.
* A customer satisfied with the quality of a product will remain loyal and wait for new functionality in the next version.
* Quality is a distinguishing attribute of a system indicating the degree of excellence.
* Regression Testing
* Internal Testing
* Unit Testing
* Application Testing
* Stress Testing

**Maintenance phase**

* Maintenance is the process of changing a system after it has been deployed.
* Corrective maintenance: identifying and repairing defects
* Adaptive maintenance: adapting the existing solution to the new platforms. Perfective Maintenance: implementing the new requirements
* The maintenance phase is the phase which comes after deployment of the software into the field.
* Software maintenance is one of the activities in software engineering, and is the process of enhancing and optimizing deployed software (software release), as well as fixing defects.
* Software maintenance is also one of the phases in the System Development Life Cycle (SDLC), as it applies to software development.

**18.Explain phases of waterfall model**

* **REQUIREMENT COLLECTION:**
* All possible requirements of the system to be developed are captured in this phase and documented in requirement specification document.
* Requirement gathering is also known as election.

**Analysis:**

* All the information and requirement for the product development is gathered from the customer and processes for analysis.
* The primary role of part is so incompleteness related to software product development.

**Design phase**

* This phase aims to transform the requirements gathered in the SRS into suitable from which permits further coding language.
* It defines the overall software architecture together with high level and detailed design.

**Implementation**

* In this phase the source code is written as per requirements.
* The physical design specification are turned into a working code.
* The system is developed in small programs called unit, after which these units are integrated.

**Testing**

* The testing phase is a separate phase which is performed by a different team after the implementation is completed.
* Unfortunately, delegating (alternate) testing to another team leads to as lack (dull) attitude regarding quality by the implementation team.
* Testers check the program for all possible defects, by running test cases.
* All flaws and bugs detected during this phase are fixed to Quality Assurance.

**Maintenance**

* After the deployment phase, the next step is to provide support and maintenance for the software, making sure it runs smoothly.
* configuration and version management
* reengineering (redesigning and refactoring)
* updating all analysis, design and user documentation

**19.Write phases of spiral model**

* **Planning:** determination of objectives, alternatives and constraints
* **Risk analysis:** Analysis of alternatives and identification/resolution of risks
* **Customer evaluation:** Assessment of the results of engineering
* **Engineering:** Development of the next level product.

**20.Write agile manifesto principle.**

* Individuals and interactions - in agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.
* Working software - Demo working software is considered the best means of communication with the customer to understand their requirement, instead of just depending on documentation.
* Customer collaboration - As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.
* Responding to change - agile development is focused on quick responses to change and continuous development.

**21.What is join.**

* A join is an SQL operation performed to establish a connection between two or more database tables based on matching columns, thereby creating relationship between the tables.

**22.Write types of joins.**

* **INNER JOIN**: returns rows when there is a match in both tables.
* **LEFT JOIN**: returns all rows from the left table, even if there are no matches in the right table.
* **RIGHT JOIN**: returns all rows from the right table, even if there are no matches in the left table.
* **FULL JOIN**: returns rows when there is a match in one of the tables. DDL - Data Definition Language.

**23. Explain working methodology of agile model and also pros and cons.**

* **Agile model methodology:**
* Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product.
* Agile Methods break the product into small incremental builds.
* These builds are provided in iterations.
* Each iteration typically lasts from about one to three weeks.
* Every iteration involves cross functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing, and acceptance testing.
* Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In agile the tasks are divided to time boxes (small time frames) to deliver specific features for a release.
* Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer.
* Agile thought process had started early in the software development and started becoming popular with time due to its flexibility and adaptability.

**24.Draw usecase on online shopping product using COD**

ADMIN

USER

**25.Draw usecase on online shopping product using payment gateway.**

**Customer**