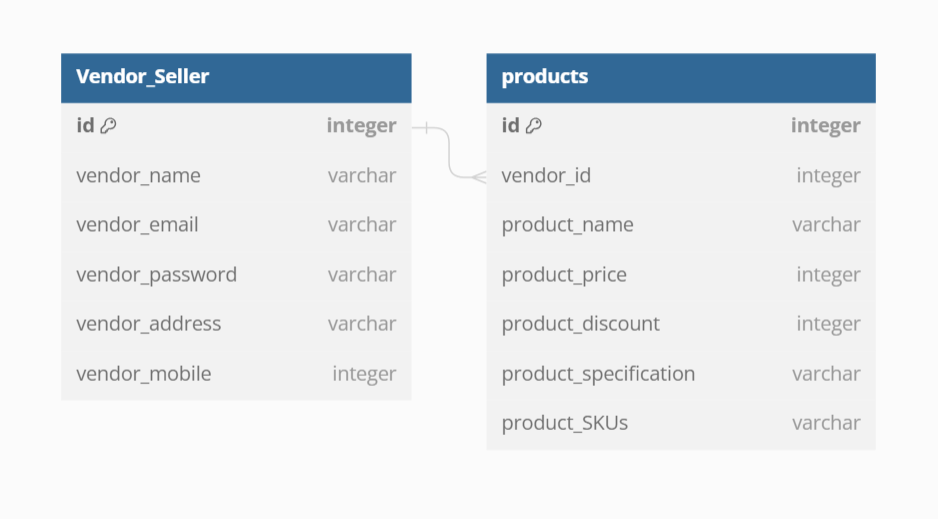
**Database Design Assignment**

**E- Commerce Application**

****

1. Physical Entity Relationship diagram of database

**Entity**

* An entity is a thing that can have data stored about it. It can be a physical object (e.g. car, person), a concept (e.g. address) or an event (e.g. student enrolment in a course). They represent nouns.
* They are usually represented as rectangles on an ERD with the entity name inside the rectangle.
* An entity can also be a strong entity or a weak entity. What’s the difference?
* A strong entity has an identifier (a primary key) and does not depend on any other entities for it to exist. For example, a student may be a strong entity, as it can have a primary key and does not depend on any other entities for it to exist.
* A weak entity is one that depends on a strong entity for existence. This means it has a foreign key to another entity. For example, an enrolment of a student may be a weak entity, as an enrolment cannot exist without a student.

**Relationship**

A relationship in an ERD defines how two entities are related to each other. They can be derived from verbs when speaking about a database or a set of entities.

Relationships in ERDs are represented as lines between two entities, and often have a label on the line to further describe the relationship (such as “enrols”, “registers”, “completes”).

There are several types of relationships that are represented on an ERD:

* One to one: One record of an entity is directly related to another record of an entity
* One to many: One record of an entity is related to one or more records of another entity.
* Many to many: Many records of one entity can be related to many records of another entity.

**Attribute**

An attribute is a property of an entity or something that can be used to describe an entity. They are often represented as ovals, or as entries inside an entity.

There are several different types of attributes represented on an ERD:

* Simple: an attribute that cannot be split into other attributes, such as a first name.
* Composite: an attribute that can be split into other attributes, such as name being split into first, middle, and last name.
* Derived: an attribute that is calculated or determined from another attribute, such as the age of record being calculated from the created date.

1. Explain about searching performance. How will you handle replication in SQL for searching & Reporting?

* **Data Replication** is the process of storing data in more than one site or node. It is useful in **improving the availability of data**.
* It is simply copying data from a database from one server to another server so that all the users can share the same data without any inconsistency.
* The result is a **distributed database** in which users can access data relevant to their tasks without interfering with the work of others.
* Data replication encompasses duplication of transactions on an ongoing basis, so that the **replicate is in a consistently updated state** and synchronized with the source.
* However, in data replication data is available at different locations, but a particular relation must reside at only one location.
* There can be full replication, in which the whole database is stored at every site. There can also be partial replication, in which some frequently used fragments of the database are replicated, and others are not replicated.

### Features of data replication are:

**Increased Availability:**Data replication can improve availability by providing multiple copies of the same data in different locations, which reduces the risk of data unavailability due to network or hardware failures.

**Improved Performance:**Replicated data can be accessed more quickly since it is available in multiple locations, which can help to reduce network latency and improve query performance.

**Enhanced Scalability:** Replication can improve scalability by distributing data across multiple nodes, which allows for increased processing power and improved performance.

**Improved Fault Tolerance:** By storing data redundantly in multiple locations, replication can improve fault tolerance by ensuring that data remains available even if a node or network fails.

**Improved Data Locality:**Replication can improve data locality by storing data close to the applications or users that need it, which can help to reduce network traffic and improve performance.

**Simplified Backup and Recovery:**Replication can simplify backup and recovery processes by providing multiple copies of the same data in different locations, which reduces the risk of data loss due to hardware or software failures.

**Enhanced Disaster Recovery:**Replication can improve disaster recovery capabilities by providing redundant copies of data in different geographic locations, which reduces the risk of data loss due to natural disasters or other events.

**Advantages of full replication –**

* High Availability of Data.
* Improves the performance for retrieval of global queries as the result can be obtained locally from any of the local site.
* Faster execution of Queries.

1. Mention about Indexing, Normalization and Denormalization.

**Normalization:** Normalization is the method used in a database to reduce the data redundancy and data inconsistency from the table. It is the technique in which non-redundancy and consistency data are stored in the set schema. By using normalization, the number of tables is increased instead of decreased.

**Denormalization: Denormalization**is also the method which is used in a database. It is used to add the redundancy to execute the query quickly. It is a technique in which data are combined to execute the query quickly. By using denormalization the number of tables is decreased which oppose to the normalization.

**Indexing:** An index is a schema object. It is used by the server to speed up the retrieval of rows by using a pointer. It can reduce disk I/O(input/output) by using a rapid path access method to locate data quickly. An index helps to speed up select queries and where clauses, but it slows down data input, with the update and the insert statements. Indexes can be created or dropped with no effect on the data.

1. How will you handle scaling, if required at any point of time.

## **Scaling:**

* It can be defined as a process to expand the existing configuration (servers/computers) to handle many user requests or to manage the amount of load on the server.
* This process is called scalability. This can be done either by increasing the current system configuration (increasing RAM, number of servers) or adding more power to the configuration.
* Scalability plays a vital role in the designing of a system as it helps in responding to many user requests more effectively and quickly.

There are two ways to do this:

1. Vertical Scaling
2. Horizontal Scaling

**Vertical Scaling:**

* It is defined as the process of increasing the capacity of a single machine by adding more resources such as memory, storage, etc. to increase the throughput of the system.
* No new resource is added, rather the capability of the existing resources is made more efficient.
* This is called Vertical scaling.
* Vertical Scaling is also called the Scale-up approach.

Example: MySQL

### Horizontal Scaling:

* It is defined as the process of adding more instances of the same type to the existing pool of resources and not increasing the capacity of existing resources like in vertical scaling.
* This kind of scaling also helps in decreasing the load on the server. This is called **Horizontal Scaling**.  
  Horizontal Scaling is also called the Scale-out approach.
* In this process, the number of servers is increased and not the individual capacity of the server.
* This is done with the help of a **Load Balancer** which basically routes the user requests to different servers according to the availability of the server. Thereby, increasing the overall performance of the system.
* In this way, the entire process is distributed among all servers rather than just depending on a single server.   
  Example: **NoSQL, Cassandra, and MongoDB**

1. Mention all the assumptions you are taking for solutions.

Assumptions definition and review build on the situation analysis, project objectives, constraints, and measurements standards. For planning purposes, the assumption defines the key operating characteristics, variables and economies of current and alternative systems. Assumptions generally fall into three classes:

**Business assumptions** – They define the characteristics of the general environment including relevant market, consumer, and product trends and competitive actions, within which an alternative logistics plan must operate. They are generally outside the ability of the firm to change.

**Management assumptions**define the physical and economic characteristics of the current or alternative logistics environment and are generally within the firm’s ability to change or refine. Typical assumptions include a definition of alternative distribution facilities, transport modes, logistics processes and fixed and variable costs.

**Analysis assumption** defines the constraints and limitations that must be included to fit the problem to the analysis technique. These assumptions frequently concern problem size, degree of analysis detail and solution methodology.