A data model is a conceptual illustration of data that explains the organization, constraints, and connections of the data. It offers a structure for data management and organization that is methodical. Hierarchical, network, relational, object-oriented, and NoSQL models are just a handful of the several types of data models.

The first category of data models, hierarchical data models, have data organized in a tree-like structure. This style is useful for displaying data that has a clear hierarchy, but it can be difficult to use and maintain. Data is arranged as a graph with nodes and edges in network data models, which were created as an enhancement over hierarchical models. This paradigm allows for more flexible interactions between data elements, but organizing and navigating it can be difficult..

The most often used data models right now are relational ones, which arrange data in tables with rows and columns. It is feasible to handle and organize data in an intelligible manner with this method. Relational databases make use of SQL as a query language. The object-oriented programming paradigm, in which data is organized as objects with properties and methods, serves as the foundation for object-oriented data models. Despite being difficult to construct and manage, this paradigm is suitable for displaying complex data with extensive relationships. Massive amounts of unstructured data that are organized as key-value pairs, documents, or graphs can be handled by NoSQL data models. Although it can be difficult, this paradigm offers high performance and scalability.

Newer data models are replacing older ones for a number of reasons, including:

* **Scalability**: In the data-driven world of today, it is crucial since out-of-date data models would not be able to handle large amounts of data.
* **Flexibility:** In order to support more complicated interactions and data structures, more recent data models offer more adaptable methods of storing and organizing data.
* **Performance:** More recent data models now have better performance and scalability, allowing for quicker and more effective data analysis.
* **Complexity:** Newer data models offer easier and more intuitive ways to organize and handle data than older ones, which can be complex and challenging to manage.
* **Rising technologies:** Newer data models are being created in response to developing technologies like big data and cloud computing, which call for specific approaches to data management and processing.

Data models, to put it simply, offer a conceptual framework for handling and arranging data. Newer data models are still being developed to solve the problems of existing data models and to manage developing technologies, despite the fact that there are many different types of data models, each with its own advantages and disadvantages.

There are several approaches to database design, including:

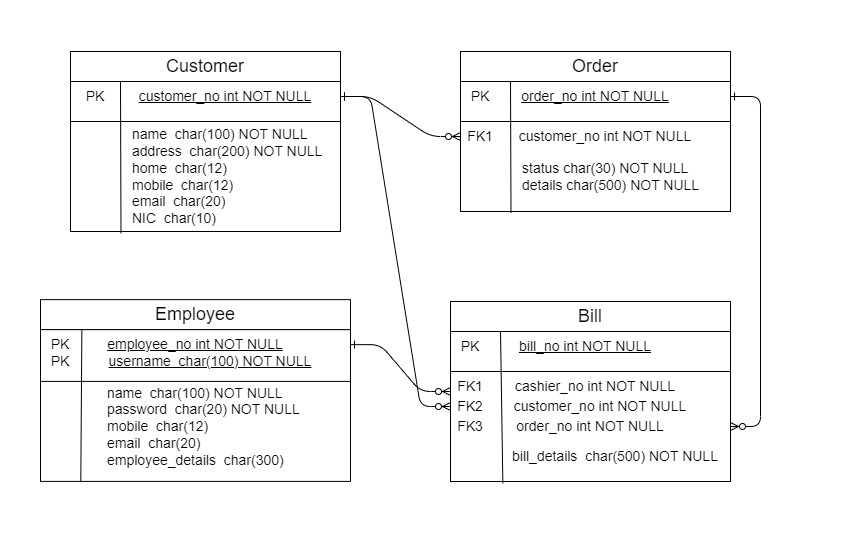
* **Top-down approach:** In this method, the database design is produced from the overall system architecture. The main objective in developing the database schema was to meet the system's functional needs. This method works well for small to medium-sized systems with clearly specified system designs.
* **Bottom-up approach**: The database design is created utilizing the data fragments and connections between them in this method. Supporting the system's data requirements is the main goal of creating the database structure. This method works well for complicated, massive systems with ill-defined data requirements.
* **Entity-relationship (ER) modeling**: Entities, attributes, and relationships are used to define the system's data requirements in the entity-relationship (ER) modeling method of database design. The components of the system, as well as the connections between them, are represented by entities, characteristics, and relationships. Relational databases are frequently created using ER modeling.
* **Object-oriented modeling**: In order to express the system's data requirements, object-oriented concepts like inheritance, encapsulation, and polymorphism are used in object-oriented modeling, a method for creating databases. This strategy is appropriate for complicated systems that call for an adaptable and extensible data model.
* **Data normalization**: Data normalization is a relational database design method that lowers data duplication and enhances data integrity. By organizing the data into tables that adhere to particular criteria, data normalization aims to eliminate data abnormalities such as update anomalies, insertion anomalies, and deletion anomalies.
* **Dimensional modeling**: Data warehouses and business intelligence tools are made using the database approach known as dimensional modeling. The data is divided into facts and dimensions in dimensional modeling, where facts are the business KPIs and dimensions are the properties that describe the facts.

1. To sum up, there are a variety of database design strategies, each with its own advantages and disadvantages. The choice of strategy is influenced by the precise needs of the system as well as the expertise and experience of the database designer.

Diagram

Description automatically generated





The conceptual data model is transformed into a relational schema using a mapping algorithm, which entails the following steps:

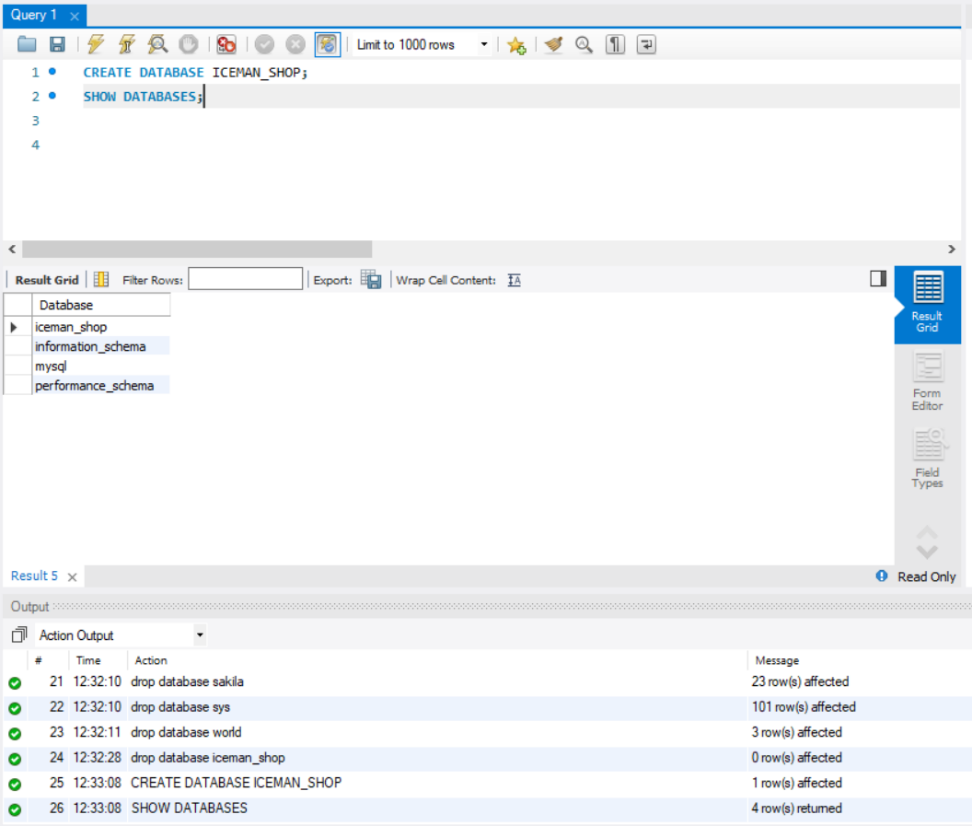
* Compile a list of all entities and their attributes.
* Describe each link and its underlying principles.
* Create tables for each entity and assign primary keys to them.
* For each many-to-many relationship, make a separate table with a primary key, and add foreign keys to each table to describe the relationships.

Explanation of the schemas:

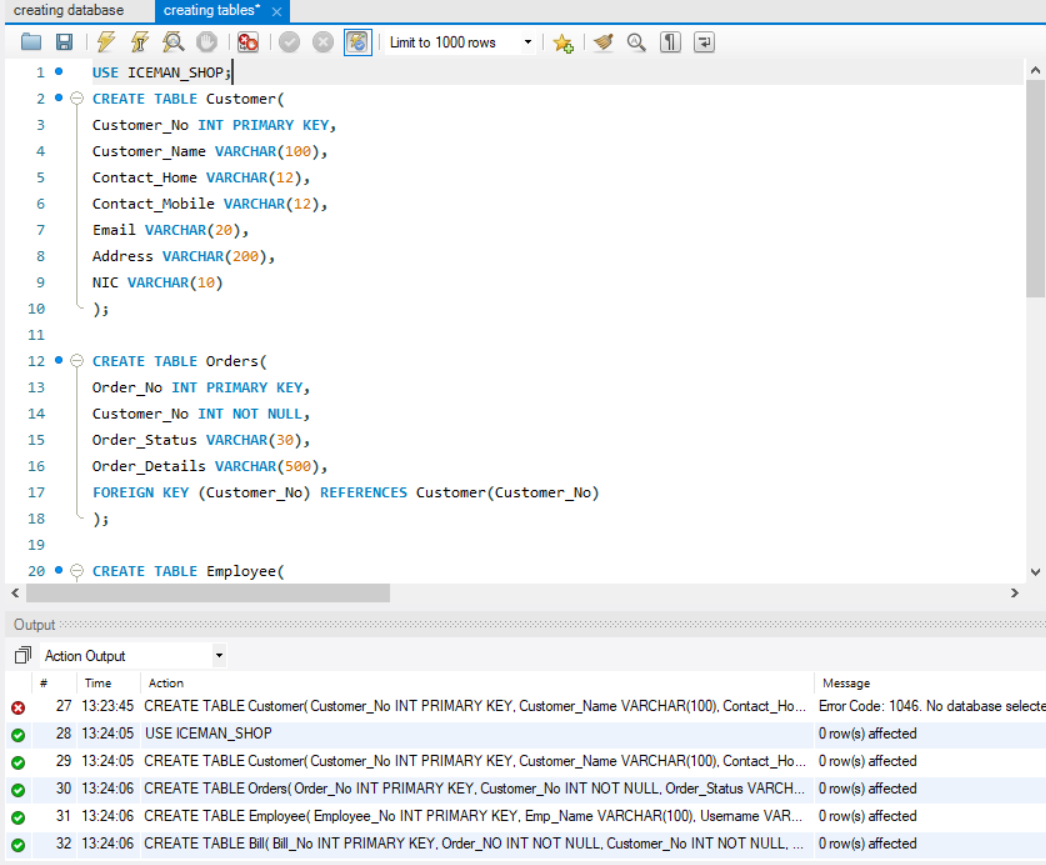
* In the Customers database, which holds customer data, the customer\_no acts as the primary key. The extra properties include address, NIC, home\_contact, mobile\_contact, email\_contact, and customer\_name.
* The Orders database, which stores order details, uses Order\_no as its primary key. Flavor, additives, quantity, order\_details, order\_status, and customer\_no, a foreign key that refers to the Customers table, are additional properties.
* The Bills table, which utilizes bill\_no as its primary key, houses the details of each bill. Additional properties that act as foreign keys to the Orders and Customers tables are order\_no and customer\_no. The table also includes cashier\_no and other\_details.
* The Employees table, which utilizes cashier\_no as its primary key, holds data on each employee. The extra properties are cashier\_name, username, and password. The table contains information on the cashier's login.



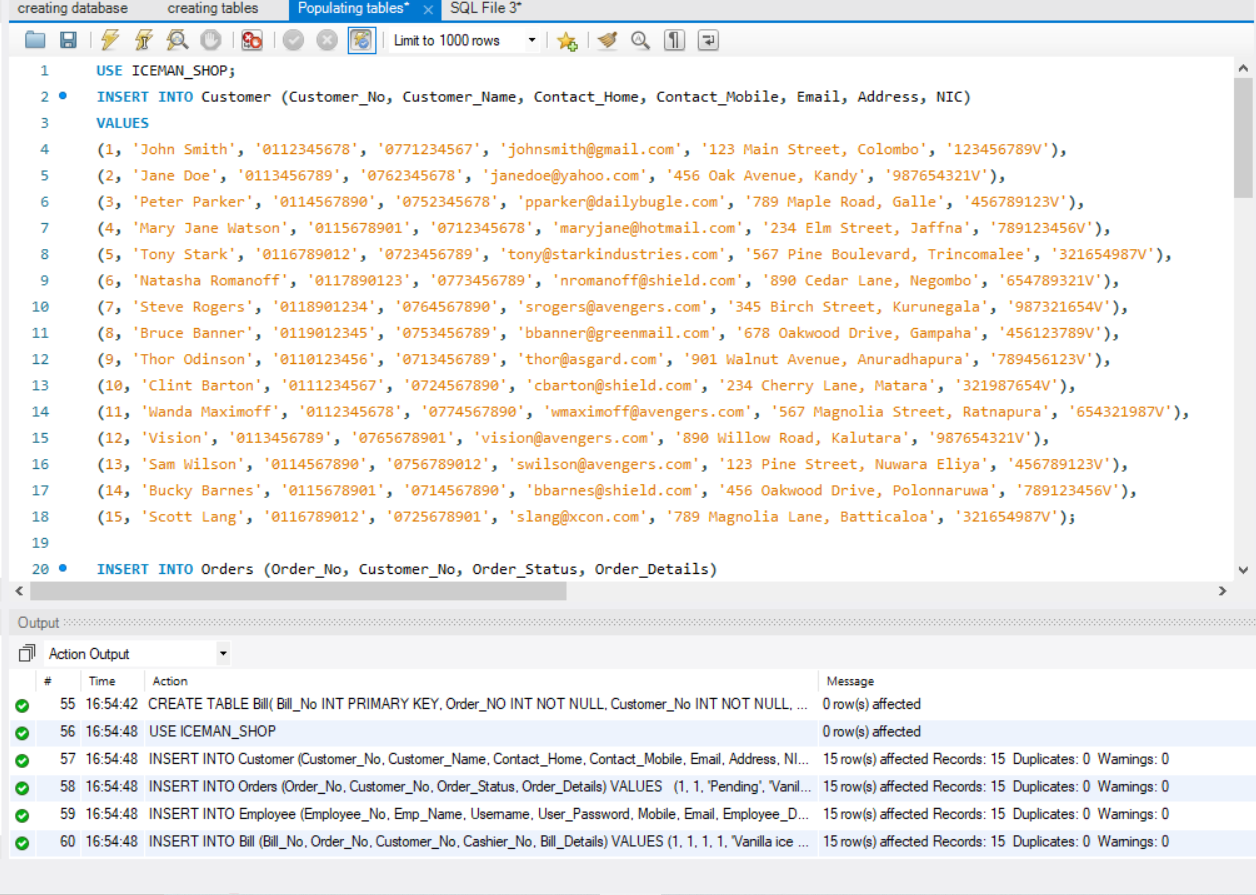
* Creating the database



* Create tables.



* Populating tables



* Extracting data from tables

Graphical user interface

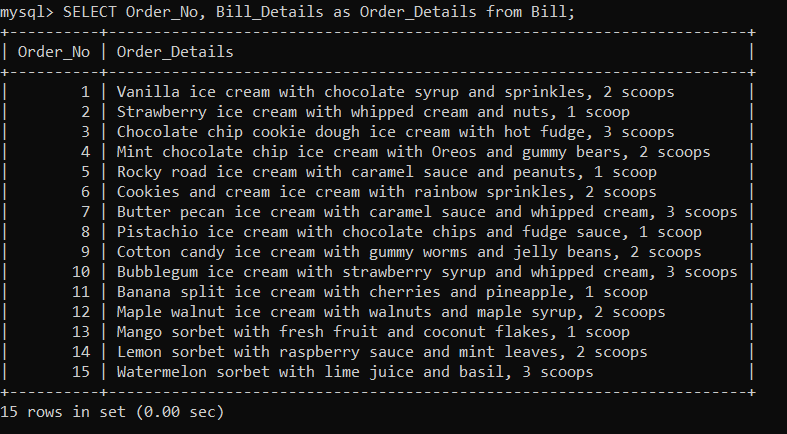
Description automatically generated with low confidence

Text

Description automatically generated



List of items of a particular order



No of ongoing orders made by a particular customer

Text

Description automatically generated with low confidence

Graphical user interface, text, application

Description automatically generated

List of customers and their bills

A picture containing text

Description automatically generated



Here is the test plan for this system.

* Introduction:

The purpose of this test plan is to validate the precision and efficiency of the Iceman Shop database system. The test plan contains the test approach, test cases, test objectives, and test scope.

* Scope:

The database system for the Iceman Shop is included in the scope of this test plan. The following aspects of the system will be looked at:

1. Evaluate how well the system works

2. Performance testing the system

* Objectives:

The following are the goals of this test strategy:

1. To guarantee that the system operates properly and effectively

2. To confirm that the system satisfies the user's expectations

3. To find any problems or flaws in the system

* Test Strategy:

The following steps will be part of the test strategy for the database system for the Iceman Shop:

1. Unit testing: To ensure that each database table is running correctly, unit testing will be applied to each table.

2. Integration Testing: Integration testing will be performed to ensure that all of the database's tables are interacting with one another without any issues.

3. System testing will be carried out to ensure that the system is generally functioning correctly.

4. Performance Testing: To ensure sure the system is responsive and able to handle the anticipated load, performance testing will be carried out.

5. Security Testing: Security testing will be carried out to ensure that the system and user data are secure.

* Test Cases:

The following test cases will be performed on the Iceman Shop database system:

1. Test case name: Customer Information Test

Test goal: Verify that customer data is accurately recorded in the database.

Steps for the test:

Adding new client information to the database, retrieving it from the database, and comparing it to the newly added information are the three steps involved.

1. Test case name: Order Information Test

Test goal: Verify that order data is accurately saved in the database.

Test procedures: Update the database with fresh order details

Check that the order information in the database matches the information that was added Retrieve the order information from the database

1. Test case name: Employee Information Test

Test goal: Confirm that employee data is accurately recorded in the database.

Steps for the test include Adding new employee data to the database.

Check that the employee information in the database matches the information that was added Retrieve the employee information from the database

1. Test case name: Bill Information Test

The purpose of the test is to confirm that the database's bill information is properly stored.

Steps for the test include: Adding fresh bill data to the database.

Check that the bill information in the database matches the newly entered information by retrieving the bill information from the database.

1. Test case name: Performance Test

Test goal: Confirm that the system is capable of handling the anticipated load and that it is responding promptly.

Test procedures include simulating a heavy load on the system, tracking its response time, and ensuring that it can handle the load.



Verification and validation can be used to guarantee the dependability and quality of a database. The following are the methods used in our database to handle verification and validation:

* Data entry validation: We have implemented a number of data entry validation procedures to make sure that the data input into the database is accurate and consistent. For instance, we have set limitations on the length, format, and data type of specific database fields.
* Referential integrity: In order to ensure that the database's data is accurate and consistent, referential integrity requirements have been put in place. As a result, information that is referred to by another database record is shielded from deletion or modification.
* Testing: We have developed a thorough test plan to ensure that the database functions as expected. We've employed a variety of testing techniques, including functional testing, performance testing, and integration testing.
* User access rights: Access rights have been assigned to different types of users in order to ensure that only authorized users can access the database. As an example, we've given certain users full access while giving others read-only access. We have additionally established password rules to ensure that only authorized users can access the database.

Our database uses testing, access permissions, referential integrity constraints, and data entry validation methodologies to address verification and validation. The consistency, accuracy, and security of the database are ensured by these procedures.

2. Navathe, S.B. and Elmasri, R., 2016. database systems' foundational concepts. Education by Pearson.
3. (2008) Halpin, T., and Morgan, T. Relational databases and information modeling. Publishers Morgan Kaufmann.
4. W3Schools. (n.d.). SQL SELECT Statement. Retrieved April 23, 2023, from <https://www.w3schools.com/sql/sql_select.asp>
5. Microsoft. (n.d.). SQL Server - T-SQL. Retrieved April 23, 2023, from <https://docs.microsoft.com/en-us/sql/t-sql/language-reference?view=sql-server-ver15>
6. PostgreSQL. (n.d.). SQL Query Language. Retrieved April 23, 2023, from <https://www.postgresql.org/docs/current/sql.html>
7. "Database Testing - A Complete Guide (Updated 2022)." Guru99, 14 Feb. 2022, <https://www.guru99.com/database-testing.html>.
8. Lucidchart. (n.d.). ER Diagrams. Retrieved from <https://www.lucidchart.com/pages/er-diagrams>
9. The first one is "Validation and Verification Techniques for Relational Database Applications." The eighth issue of the International Journal of Advanced Research in Computer Science, published in September-October 2017, was 1569-1573. 23rd April 2023 access date. <http://ijarcs.info/index.php/Ijarcs/article/view/4369>.
10. Visual Paradigm. (n.d.). ER Diagram Tutorial. Retrieved from <https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/>
11. MySQL Tutorial. (n.d.). Creating a Relational Schema. Retrieved from <https://www.mysqltutorial.org/mysql-create-table/>