**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **10-July-2020** | **Name:** | **Bhuvanesh M** |
| **Course:** | **Coursera** | **USN:** | **4AL16EC015** |
| **Topic:** | **SQL data types** | **Semester & Section:** | **8th sem & ‘A’ section** |
| **Github Repository:** | **Bhuvan** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report –**  In my first session today I have studied about - **SQL data types**  The data type of a column defines what value the column can hold: integer, character, money, date and time, binary, and so on. SQL Data Types Each column in a database table is required to have a name and a data type.  An SQL developer must decide what type of data that will be stored inside each column when creating a table. The data type is a guideline for SQL to understand what type of data is expected inside of each column, and it also identifies how SQL will interact with the stored data.  Data types might have different names in different database. And even if the name is the same, the size and other details may be different! **Always check the documentation!** MySQL Data Types (Version 8.0) In MySQL there are three main data types: string, numeric, and date and time. String data types:  |  |  | | --- | --- | | **Data type** | **Description** | | CHAR(size) | A FIXED length string (can contain letters, numbers, and special characters). The size parameter specifies the column length in characters - can be from 0 to 255. Default is 1 | | VARCHAR(size) | A VARIABLE length string (can contain letters, numbers, and special characters). The size parameter specifies the maximum column length in characters - can be from 0 to 65535 | | BINARY(size) | Equal to CHAR(), but stores binary byte strings. The size parameter specifies the column length in bytes. Default is 1 | | VARBINARY(size) | Equal to VARCHAR(), but stores binary byte strings. The size parameter specifies the maximum column length in bytes. | | TINYBLOB | For BLOBs (Binary Large OBjects). Max length: 255 bytes | | TINYTEXT | Holds a string with a maximum length of 255 characters | | TEXT(size) | Holds a string with a maximum length of 65,535 bytes | | BLOB(size) | For BLOBs (Binary Large OBjects). Holds up to 65,535 bytes of data | | MEDIUMTEXT | Holds a string with a maximum length of 16,777,215 characters | | MEDIUMBLOB | For BLOBs (Binary Large OBjects). Holds up to 16,777,215 bytes of data | | LONGTEXT | Holds a string with a maximum length of 4,294,967,295 characters | | LONGBLOB | For BLOBs (Binary Large OBjects). Holds up to 4,294,967,295 bytes of data | | ENUM(val1, val2, val3, ...) | A string object that can have only one value, chosen from a list of possible values. You can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted. The values are sorted in the order you enter them | | SET(val1, val2, val3, ...) | A string object that can have 0 or more values, chosen from a list of possible values. You can list up to 64 values in a SET list |   Okay. Now, let's wrap up our discussion about data models by getting  back to talking about the type of data models we'll be working with in this class. After this video, you should be able to define and  describe both relational and transactional database models, define entities, attributes and relationship,  describe and explain the difference between one-to-one, one-to-many and many-to-many relationships,  describe the use of a primary key in a database and explain how ER diagram is used to document and illustrate relationships. I want to kick off this video by talking a little bit about relational versus transactional database models. A relational model is a database design that shows the relationships between the different tables and this is really used to optimize querying data, making it easy and intuitive to access the data. Transactional, on the other hand, you can think of as a more operational database.  If you are in healthcare, for example, you may have a transactional database that is used to store all the claims information and then this information may not be stored in a great way for querying and using it for analysis. In fact, you may need to take and extract that transactional information from the database and move it into a relational model. Most of what we will be working with in this class is the relational model.  The building blocks for this relational model are really three simple things. We have entities which are a person, place, thing or event. These are very distinguishable. They are unique. They are distinct. For example, I could be an entity, Sadie St. Lawrence, and then we have attributes, which are characteristics of this entity. As an entity, it would be myself and then an attribute about me would be that I am female. Then, the third building block of the model is the relationship. This describes the association among different entities. There are a few types of relationships in a database and the ones I want to cover are the three shown here: one-to-many, many-to-many and one-to-one. If you think of a one-to-many relationship, this could be one customer that has many invoices. When you think of a many-to-many relationship, this could be an example of many students to many classes. You may have one student who belongs to lots of different classes or you may have a class who has many different students. Then, if you think of a one-to-one relationship, this is a manager to a store. Let's say you have a sporting goods store and each of those stores has only one manager. That would be one example of a one-to-one relationship.  To understand these relationships between the tables a lot better, what's often used to depict this are ER diagrams. An ER model then is composed of the entity types and the specific relationships that can exist between these entities. These are usually displayed in a visual format and a relate represents a relationship between the tables. It often helps you to understand and represent a business process and it will show the links between these tables. The links are really important because in a later lesson,  we're going to learn how to join these tables together and combine the data. Being able to look at this diagram and see how they relate to each other is really important. What we will use to join these tables together are two things. |